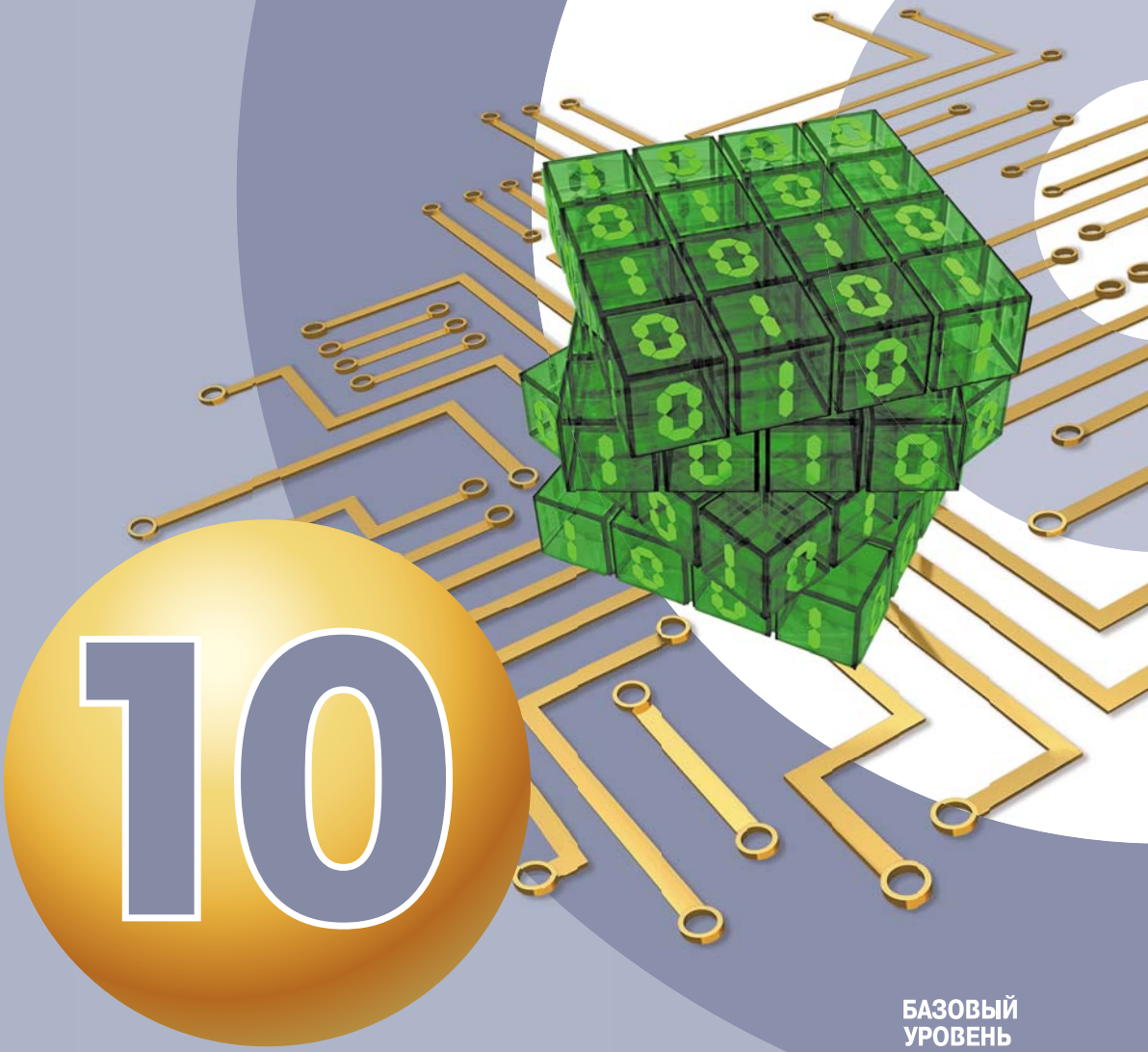
СФЕРЫ

А.Г. Гейн
Н.А. Юнерман

ИНФОРМАТИКА

ИНФОРМАТИКА



10

БАЗОВЫЙ УРОВЕНЬ

А. Г. Гейн
Н. А. Юнерман

ИНФОРМАТИКА

10 класс

БАЗОВЫЙ УРОВЕНЬ

Учебник

Допущено
Министерством просвещения
Российской Федерации

4-е издание, стереотипное

Москва
«Просвещение»
2022

УДК 373.167.1:004+004(075.3)
ББК 32.81я721
Г29

Серия «Сферы» основана в 2003 году

На учебник получены **положительные** заключения
научной (заключение РАО № 1196 от 19.11.2016 г.),
педагогической (заключение РАО № 1087 от 21.11.2016 г.)
и **общественной** (заключение РКС № 550-ОЭ от 19.12.2016 г.) экспертиз

Издание выходит в pdf-формате.

Гейн, Александр Георгиевич.

Г29 Информатика. 10 класс : базовый уровень : учебник : издание в pdf-формате / А. Г. Гейн, Н. А. Юнерман. — 4-е изд., стер. — Москва : Просвещение, 2022. — 126, [2] с. : ил. — (Сферы).

ISBN 978-5-09-101596-6 (электр. изд.). — Текст : электронный.

ISBN 978-5-09-088838-7 (печ. изд.).

Учебник входит в учебно-методический комплект «Информатика. 10—11 классы» авторского коллектива под руководством А. Г. Гейна. Представленный курс реализует установку на формирование у учащихся системы базовых понятий информатики и представлений об информационных технологиях, а также вырабатывает умения применять их для решения жизненных задач. В объяснительных текстах и заданиях учебника действует разветвлённая система методических приёмов, направленных на создание мотивации к изучению материала, реализующих движение от простого к сложному, дифференциацию заданий по степени креативности и т.д. Знания проверяются системой вопросов, покрывающих всю систему базовых понятий курса, а в разделе «Теоретический практикум» обширной системой заданий закрепляются умения применять знания. Навыки в использовании информационных технологий для решения различных исследовательских и практических задач учащиеся получают при проведении лабораторных работ в компьютерном классе. В учебнике приведено подробное описание лабораторного практикума, включающего работы по всем разделам.

УДК 373.167.1:004+004(075.3)
ББК 32.81я721

ISBN 978-5-09-101596-6 (электр. изд.)
ISBN 978-5-09-088838-7 (печ. изд.)

© АО «Издательство «Просвещение», 2019
© Художественное оформление.
АО «Издательство «Просвещение», 2019
Все права защищены

Предисловие

Уважаемые старшеклассники!

Вы вступили в тот важный период школьного образования, когда вам предстоит определить приоритеты дальнейшего профессионального пути. Но что бы вы ни выбрали, владение базовыми информационными технологиями, понимание информационных процессов, протекающих в производственной сфере и жизни общества, создают надёжный фундамент вашей успешной самореализации. Жизнь в современном обществе, насыщенном быстро меняющимися информационными средствами, требует не просто знания и владения теми инструментами, которые сегодня доступны, но и понимания тенденций их развития и готовности к освоению новинок информационной среды.

В учебнике для 10 класса вам предложены три главы: «Информация и информационные процессы», «Основные информационные объекты. Их создание и компьютерная обработка» и «Алгоритмы и исполнители. Основы программирования». Конечно, с этими темами вы в той или иной степени знакомились в курсе информатики в предыдущие годы, но здесь они получат дальнейшее развитие.

Использование компьютера — неотъемлемая черта современного курса информатики. Именно с его помощью вы будете выполнять большинство заданий, предусмотренных учебником. Но и компьютер будет вашим помощником в получении новых знаний, освоении новых технологий, в том числе и с помощью разнообразных источников в Интернете. Ведь важно не только усвоить то, что мы рассказываем вам в учебнике, но и научиться самостоятельно добывать знания, соотносить их с уже имеющимся опытом и подниматься всё выше и выше в своих компетенциях. Помогать вам в этой деятельности будет компьютерный практикум, помещённый в конце каждой главы.

Желаем вам успехов!

Информационный блокнот

ИССЛЕДОВАНИЯ

ВНИМАНИЕ!

РАБОТАЕМ С УЧЕБНИКОМ

Материал учебника включает три главы. Перед каждой главой приводится информация, которая отражает суть её содержания.

Каждая глава делится на темы. Тема начинается с вводных рубрик «Вы узнаете» и «Вспомните». Рубрика «Вы узнаете» знакомит вас с основными вопросами, которые обсуждаются в теме. Рубрика «Вспомните» подскажет, что необходимо освежить в памяти из ранее изученного материала, для того чтобы успешно осваивать новый.

Основной текст сопровождают рубрики, которые помогут вам глубже понять материал и расширить кругозор.

В рубрике «Информационный блокнот» содержатся интересные сведения и дополнительная информация к основному тексту. Там же вы найдёте разнообразные примеры использования информатики и информационных технологий в жизни и технике.

Рубрика «Исследования» поможет вам провести экспериментальную проверку различных гипотез и утверждений, как сформулированных в объяснительном тексте, так и выдвигаемых вами самостоятельно.

Важные понятия выделены в рубрике «Внимание!».

Графический элемент в виде лупы акцентирует ваше внимание на внутреннем содержании рисунка.

В конце каждой темы приведены вопросы, ответы на которые помогут вам сориентироваться в том, насколько хорошо вы усвоили материал. А в конце каждой главы приведён теоретический практикум, выполняя задания которого вы сможете проявить умения применять полученные знания.

Там же, в конце главы, вы найдёте рубрику «Подведём итоги», позволяющую в лаконичной форме представить материал главы в единстве и взаимосвязи.

В рубрике «Вопросы для обсуждения» приведены вопросы, которые носят проблемный характер и могут стать интересной темой для дискуссии.

Ссылки на интернет-ресурсы и литературу позволят привлечь дополнительные источники информации при изучении каждой темы.

Учебник дополняют также задачник-практикум и тетрадь-экзаменатор, составляющие единый учебный комплекс, который поможет вам качественно усвоить материал.

Глава 1

ИНФОРМАЦИЯ И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ

САМОЕ... САМОЕ...

Информация всегда играла важную роль в жизни человека. Уже наш пещерный предок, идя по следам своей будущей добычи, использовал информацию, поступающую к нему через органы чувств. Разыгрывая перед соплеменниками сцены охоты, он передавал им свой опыт и свои знания, т. е. информацию; в наскальных рисунках человек стремился сохранить информацию. Иными словами, информационные процессы сопровождают человека с первых минут его существования на Земле. Конечно, само слово «информация» появилось гораздо позже. Но сегодня оно встречается каждому из нас практически ежедневно. Что же такое информация? Это непростой вопрос, и с его обсуждения мы начинаем курс информатики.

1 ИНФОРМАЦИЯ И ЕЁ СВОЙСТВА

ВЫ УЗНАЕТЕ:

- Что понимают под термином «информация» в разных областях человеческой деятельности
- Чем знания отличаются от информации
- Какие свойства информации важно учитывать при её использовании

ВСПОМНИТЕ:

- Где вам встречалось слово «информация»

Живая природа от неживой отличается уже тем, что её существование невозможно без постоянного потока информации. Но что такое информация?

ИНФОРМАЦИЯ Говоря об информации, мы в первую очередь имеем в виду те сведения, которые предоставляют нам Интернет, телевидение, газеты, книги. Информацию несут нам карты местности и картины художников. Информация аккумулирована в тысячах научных трудов, составляющих основу нашей цивилизации. Информация составляет основу ваших школьных учебников, без которых трудно стать образованным человеком. Поэтому нередко говорят, что **информация — это сведения, знания об окружающем человека мире и о самом себе**. Так обычно разъясняется понятие информации в гуманитарных науках, где центральным объектом является человек как существо социальное.

Но есть и другая точка зрения на то, что такое информация. Поведение любого живого организма — муравья, птицы, собаки... — показывает, что они тоже воспринимают информацию об окружающем мире и в соответствии с этой информацией выстраивают своё поведение. Более того, они обмениваются информацией друг с другом. Да и вы, отдавая приказы домашним животным, уверены, что они воспринимают передаваемую вами информацию.

Информация нужна и различным техническим устройствам, работающим автоматически, т.е. без вмешательства человека. Какое бы такое устройство мы ни взяли, оно обязательно использует информацию, получаемую от человека и окружающей среды. Сформулируем ещё одну точку зрения на понятие информации. **Информация — это то, что позволяет живым организмам, их сообществам или техническим системам реагировать на окружающую среду, обеспечивая их целенаправленную деятельность.** Конечно, цели техническим системам определяют люди, которые эти системы используют.

Такое понимание термина «информация» называют кибернетическим. Кибернетика (от греческого κυβερνητικ — искусство управления) — наука об общих закономерностях получения, хранения, передачи и преобразования информации в сложных управляющих системах, будь то живые организмы, технические системы или общество.

Теперь представьте себя посередине пустыни. Вокруг вас совершенно ровная поверхность, не на чем глазу остановиться. Солнце в зените, даже тень расположилась прямо под вами... Прошло три часа. Ничего не изменилось, только тень вытянулась в определённую сторону, но вы уже способны определить стороны света. Что же привело к появлению информации? Наличие разнообразия в окружающей обстановке. Таким образом, **информация — это отражение разнообразия в существующем мире**. Отсутствие разнообразия, когда неотличимы никакие два объекта, явления или процесса, — это и есть отсутствие какой бы то ни было инфор-

Слово «информация» очень древнее и происходит от латинского *informatio* — разъяснения, сведения. Именно в этом значении его использовал знаменитый древнеримский оратор Марк Туллий Цицерон в своих речах ещё в 62 г. до н.э.

Информация — нематериальное отражение материального мира, социальных процессов и явлений, а также духовного мира человека.

Понятие «информация» относится к числу наиболее общих понятий, применяемых для описания картины мира.

мации. Такой весьма общий взгляд на понятие «информация» присущ многим естественным и философским наукам. Мы же в учебнике большей частью будем вести речь об информации, которая выступает как человеческое знание. Но и общий взгляд на понятие «информация» вам нередко окажется полезным.

ЗНАНИЯ Нередко люди не делают различия между информацией и знаниями. **Знания — это информация в обобщённом и систематизированном виде, готовая к применению в схожих ситуациях.** Отдельный факт становится знанием, если он является элементом некой системы понятий и представлений человека об окружающем мире, закономерностях и процессах, в нём происходящих. Знания выступают основой для принятия решений.

Знание может быть очень разным. Это и народные приметы (скажем, о погоде), и личный житейский опыт, и религиозное знание, и, наконец, научное знание. Их не следует противопоставлять друг другу, хотя временами они могут вступать в противоречие. Научное знание характеризуется логической обоснованностью, воспроизводимостью результатов, проверяемостью, стремлением к устранению противоречий.

СВОЙСТВА ИНФОРМАЦИИ Использование информации при формировании целенаправленной деятельности опирается на такие её важные свойства, как новизна, актуальность, достоверность, объективность, ценность, полнота и т.п. Разъясним смысл каждого из этих свойств.

Информация обладает **новизной**, если её смысловое содержание отличается от смыслового содержания ранее имевшейся информации.

Информация **актуальна** (иными словами, своевременна), если она оказывает влияние на формирование деятельности именно в данный момент времени.

Информация **достоверна**, если принимается, что она отражает реальное положение дел, в частности не вступает в противоречие с уже имеющейся информацией, также признаваемой в качестве достоверной. Одним из способов проверки достоверности является её подтверждение по крайней мере тремя независимыми источниками. Вовсе не исключается, что с поступлением новой информации данная информация уже перестанет быть достоверной.

Информация **объективна**, если она не зависит от свойств источника информации. Конечно, абсолютно не зависеть от источника информация не может, но при тех или иных условиях можно считать, что такое влияние пренебрежимо мало.

Информация обладает **ценностью (полезностью)**, если она повышает вероятность достижения цели в деятельности той системы, которая использует эту информацию.

Информация **полна**, если её достаточно для достижения цели. Информация может быть **избыточной**, если для достижения цели достаточно только части этой информации. Это, однако, не означает, что надо стремиться избегать избыточности — она часто используется для защиты информации от возможных искажений в процессе передачи. Оценить полноту информации можно, только указав цель, для достижения которой она будет использоваться.

ИССЛЕДОВАНИЯ

Возьмите литературу (художественную, научную из различных источников, предоставляемых средствами массовой информации).

- Найдите примеры употребления слова «информация».
- В каждом случае попытайтесь определить, в каком смысле из указанных в объяснительном тексте использовано это слово.
- Подсчитайте частоту каждого из этих смысловых употреблений.

При использовании информации особую важность имеют её достоверность, объективность и полнота.

ВОПРОСЫ:

- Почему имеются разные точки зрения на содержание понятия «информация»?
- В чём состоят различия между понятиями «информация» и «знания»?
- Всякая ли информация увеличивает знания? Ответ обоснуйте.
- Какие свойства информации нужно учитывать при её использовании?
- Может ли информация быть достоверной, но бесполезной? Актуальной, но недостоверной? Ответ обоснуйте.

2 ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ

ВЫ УЗНАЕТЕ:
- Каковы основные виды информации
- Каковы основные виды информационных процессов
- Что называют информационным объектом

ВСПОМНИТЕ:
- Какие органы чувств человека участвуют в получении информации
- Какими средствами вы пользуетесь для сохранения информации

Непрочитанная книга, конечно, содержит информацию, но она ничего не изменит ни в вас, ни в окружающем мире. Задача, которую никто не решает, тоже содержит информацию, но и она ни на что не повлияет. Другое дело, если книга прочитана, задача решена. Значит, не информация сама по себе, а те процессы, в которых она участвует и преобразуется, представляют интерес в первую очередь. Такие процессы называют информационными.

СПОСОБЫ ПРЕДСТАВЛЕНИЯ ИНФОРМАЦИИ Оторвитесь на минуту от учебника и выгляните в окно. Вы увидите дома и людей, солнце и небо, деревья и траву и автоматически переключитесь на анализ информации совсем другого сорта. Она очень сильно отличается от однообразных строчек с буквами, которые вы видите в книге.

Информация, получаемая из сообщений, записанных с помощью букв и цифр, называется **символьной**; информация, получаемая с помощью зрительных образов окружающего мира, называется **видеоинформацией**.

Конечно, кроме символьной и видеоинформации, есть и другие виды представления информации. Информацию несёт запах; осязание доставляет нам тактильную информацию. Через вкусовые рецепторы мы получаем информацию о качестве пищи и её готовности к употреблению.

Но чтобы сохранить информацию и сделать её удобной для передачи другим людям, человек использует прежде всего символьную, визуальную и звуковую формы представления информации. Информация, представленная одновременно этими тремя формами, называется **мультимедийной**.

Обратите внимание, что информация обязательно должна быть зафиксирована. Результаты исследований облекаются в формулы, таблицы, графики. Генетическая информация фиксируется в генетическом коде, представляющем собой последовательность аминокислот. Информация о ландшафте местности фиксируется на планах и картах.

Информацию, зафиксированную каким-либо способом, будем называть **информационным объектом**.

ИНФОРМАЦИОННЫЙ ПРОЦЕСС Как вы знаете, к основным видам информационных процессов относятся получение, сохранение, передача и обработка информации. При этом фиксация информации в виде информационного объекта может осуществляться разными способами. Например, информация о тепловом состоянии какого-либо механизма может отображаться термометром в числовом виде (т.е. символьно), а может изображаться самописцем в графическом виде. Нередко приходится преобразовывать одну форму представления информации в другую. Процесс, в котором преобразуется форма представления информации, тоже относят к информационным. Таким образом, **информационный процесс** — это процесс, в ходе которого изменяется содержание информации или форма её представления.

Информация, участвующая в любом информационном процессе, должна быть зафиксирована в форме информационного объекта. Такая фиксация может осуществляться в символьном виде, в звуковом или как видеоинформация. Информация, зафиксированная в этих трёх видах одновременно, называется мультимедийной.

Открытия, сделанные в результате наблюдений и опытов

Наблюдая, как отплывающие корабли постепенно скрываются за линией горизонта, Аристотель в 330 г. до н.э. сделал вывод о шарообразности Земли.

В 1589 г. Г. Галилей, сбрасывая с Пизанской башни ядро массой 80 кг и пулю массой 200 г, установил, что время падения не зависит от массы.

В 1913 г. Э. Резерфорд, облучая тонкую пластину из золота потоком положительно заряженных частиц, открыл, что атом обладает положительно заряженным ядром.

ПОЛУЧЕНИЕ ИНФОРМАЦИИ **Получение информации** — это реализация способности живых организмов к отражению различных свойств объектов, явлений и процессов окружающего мира. У высших животных и человека для получения информации имеются органы чувств, специализированные для восприятия информации определённого вида.

Человек пошёл ещё дальше — он создал приборы, позволяющие извлекать ту информацию, которая недоступна ему в непосредственных ощущениях. Микроскоп и телескоп, термометры и шумомеры, различные индикаторы и дозиметры — вот далеко не полный перечень изобретений человечества в области средств получения информации об окружающем мире и самом себе.

ПЕРЕДАЧА ИНФОРМАЦИИ Получать информацию человек и другие живые организмы могут, не только наблюдая окружающий мир, но и общаясь друг с другом, а также обращаясь к тем или иным источникам информации. Такое получение информации происходит в результате информационного процесса другого вида — процесса **передачи информации**.

Передача информации всегда осуществляется по некоторому каналу связи от источника информации к её приёмнику. Канал связи можно уподобить транспортному средству, осуществляющему доставку информации от источника к приёмнику.

СОХРАНЕНИЕ ИНФОРМАЦИИ **Сохранение информации** всегда предполагает наличие физического носителя. Камень, на котором первобытным человеком высекались рисунки, глиняные таблички Вавилона, бумага, магнитная лента, оптический диск — чем глубже проникал человек в тайны материи, тем больше информации ему удавалось сохранять в единице объёма физического тела. Впрочем, борьба шла не только за увеличение ёмкости носителя информации, но и за надёжность её сохранения. Методы защиты информации возникли в глубокой древности, и сейчас это бурно развивающаяся область информатики, о которой мы подробно поговорим несколько позже.

ОБРАБОТКА ИНФОРМАЦИИ Обработка (преобразование) информации — достаточно широко понимаемый информационный процесс. Под **обработкой информации** понимают получение новой информации из уже имеющейся или преобразование формы представления информации.

Каким бы ни был способ обработки информации, существует нечто или некто, выполняющий эту обработку. Его обычно называют **исполнителем**. В информатике из всех исполнителей особое внимание уделяется компьютеру. Это техническое устройство было изобретено, чтобы в первую очередь облегчить человеку выполнение вычислительных работ.

Развитие информатики и совершенствование технической базы привело к пониманию, что компьютер вопреки своему названию — вычислитель — является универсальным инструментом обработки информации.

Наиболее информационно ёмкой является молекула ДНК: в ней хранится в 100 миллионов раз больше информации, чем на самой современной микросхеме с объёмом памяти тех же размеров.

ВОПРОСЫ:

- Каковы основные виды представления информации?
- Что такое информационный объект?
- Какой информационный объект называют мультимедийным?
- Что называют информационным процессом? Назовите основные виды информационных процессов.
- В каких видах информационных процессов вы участвуете, когда выполняете домашнее задание по русскому языку?

3 ЯЗЫК КАК СРЕДСТВО СОХРАНЕНИЯ И ПЕРЕДАЧИ ИНФОРМАЦИИ

ВЫ УЗНАЕТЕ:

- Какие виды языков использует человек в своей информационной деятельности
- Какой язык называют формализованным
- Какой смысл имеют термины «алфавит» и «слово» в информатике

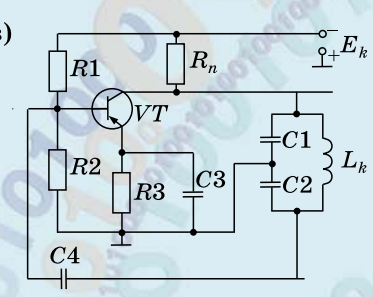
ВСПОМНИТЕ:

- Какими языками пользуется человек для общения
- Какими знаками дополнен язык человеческого общения, чтобы передавать математическую информацию
- Какие знаки используются для управления дорожным движением

а) $y = f(x)$

б) $NaOH + HCl = NaCl + H_2O$

в)



Примеры фраз языков математики (а), химии (б), электроники (в)

Одним из способов, позволяющим фиксировать информацию, является представление её в символьном виде. Рассмотрим особенности использования символьного способа представления информации.

АЛФАВИТ, СЛОВО, ЯЗЫК Само появление символьной информации связано с формированием человеческой речи. Первобытный человек каждому новому предмету или явлению придумывал имя. Сначала имён было около десятка, затем их количество перевалило за сотню... Чем глубже человек познавал окружающий мир, тем больше имён ему требовалось. Но человек способен произносить и различать на слух не так уж много звуков-фонем. Чтобы получить необходимое разнообразие имён, он начал комбинировать звуки друг с другом, получая слова. Так в ходе развития человека появилась идея конечного **алфавита**, т.е. некоторого фиксированного упорядоченного набора знаков, из которого можно составить как угодно много слов. В информатике **словом** принято называть любую последовательность символов заданного алфавита. Количество символов в слове называют **длиной** слова. Совокупность слов, используемых для записи информации в рамках какой-либо предметной области, называют **языком**.

Возникновение некоторых алфавитов было продиктовано необходимостью привлечения технических средств для передачи информации. С такой технической точки зрения наиболее удобным оказалось использование алфавита, состоящего из двух символов. Наличие тока или его отсутствие, наличие намагниченности или её отсутствие, наличие света или его отсутствие — два хорошо различимых состояния кодируют два символа алфавита. Мы в дальнейшем будем обычно считать, что двухсимвольный алфавит состоит из 0 и 1.

ВИДЫ ЯЗЫКОВ Человек в своей практике использует самые разнообразные языки. Прежде всего это языки устной и письменной речи. Довольно часто используются языки различных указателей. К таковым относятся, например, знаки дорожного движения. Кроме того, человек использует ряд языков профессионального назначения. Сюда относятся языки математических и химических формул, обозначений электроники и т.п.

В естественных языках человеческого общения — такие языки называют **естественными**, или **коммуникативными** — смысл слова нередко зависит от его окружения, или, как говорят, от **контекста**. Сравните смысл слова «вода» в двух следующих предложениях: «Вода в реке была тёплой» и «Первые ядерные реакторы предполагалось построить на тяжёлой воде». В первом случае вода — это жидкость, содержащая самые различные химические вещества в растворённом и взвешенном состоянии, во втором случае — это веще-

ство, имеющее два атома изотопа водорода и один атом кислорода. Именно неоднозначность смысла слов, его зависимость от контекста позволяет писателям создавать художественные шедевры. Она порождает ассоциации, воздействует на эмоции читателя, позволяет контекстными связями создавать целостную картину из мозаики отдельных слов. Но такая неоднозначность не годится, когда необходимо описать явление в научных целях. В этом случае за каждым словом закрепляется одно смысловое значение. Процесс закрепления за словом одного смысла называется **формализацией**. А язык, в котором каждое слово имеет единственное значение, называется **формализованным**. Язык любой науки является формализованным.

СЕМАНТИКА, СИНТАКСИС И ГРАММАТИКА ЯЗЫКА Вопросы соотнесения слову его значения и смысла составляют **семантику** языка. Однако, кроме проблемы соотношения слова и того или иного значения, есть ещё проблема образования предложений — **синтаксис** языка.

Со временем люди обнаружили, что бывают множества, содержащие бесконечно много элементов, например множество натуральных чисел. Каждому элементу нужно сопоставить обозначающее его слово. Но бесконечный список слов не составишь, нужно придумать что-то иное. Впрочем, и для конечного, но достаточно большого множества именование каждого объекта может оказаться непростой задачей (к примеру, образование названий органических соединений в химии).

Решение проблемы именования элементов бесконечного множества мы как раз и проиллюстрируем на хорошо знакомом вам бесконечном множестве натуральных чисел. Десять цифр 0, 1, 2, … , 9 — это алфавит, и каждое натуральное число имеет имя, записанное знаками этого алфавита. Имена первых девяти натуральных чисел — это сами знаки данного алфавита. А имена всех последующих чисел получаются по единому (хорошо известному) правилу. Вот оно.

Пусть для числа n уже имеется запись указанными цифрами. Если последняя цифра отлична от 9, то для получения записи следующего числа — числа $n + 1$ — нужно эту последнюю цифру заменить следующей, а все остальные цифры оставить без изменения. Если же последней цифрой числа n является 9, то она заменяется на 0, и если в старших разрядах цифры отсутствуют, то перед цифрой 0 пишется 1, в противном случае вместе с цифрой 9 по тому же правилу меняются цифры в более старших разрядах.

Правила, согласно которым из знаков алфавита образуются слова данного языка, называются **грамматикой** этого языка. Так что приведённое выше правило — это грамматика десятичной нумерации. Язык десятичной нумерации — это множество записей всех натуральных чисел. Чтобы записывать числовую информацию, выражаемую дробями, алфавит приходится расширить, добавив в него знак дробной черты или десятичной запятой. Соответствующим образом модифицируется и грамматика языка.

Формализованный язык со строго определённой грамматикой, а значит, и синтаксисом, называется **формальным**.

В различных сферах информационной деятельности человек использует языки разного вида:

- естественные — предназначенные для общения в социуме;
- формализованные — предназначенные для сохранения и передачи научной и технической информации;
- формальные — предназначенные преимущественно для сохранения и передачи информации, используемой в автоматизированных технических системах.

ИССЛЕДОВАНИЯ

Откройте свои учебники по алгебре и геометрии и сравните по доле занимаемого в них текста употребление естественного, формализованного и формального языков.

Какой из учебников вам представляется более формализованным (т.е. в нём больше доля употребления формализованного и формального языков вместе взятых)?

ВОПРОСЫ:

- Что такое алфавит?
- Как понимают термин «слово» в информатике?
- В чём заключается процесс формализации языка?
- В чём различия между синтаксисом и семантикой языка?
- Что называют грамматикой языка?

4 УНИВЕРСАЛЬНОСТЬ ДВОИЧНОГО КОДИРОВАНИЯ

ВЫ УЗНАЕТЕ:

- Для чего кодируют сообщения
- Что такое двоичный код
- В каких единицах измеряют информационный объём сообщения
- Какие коды используются для компьютеров и компьютерных сетей
- Как кодируют цвет
- Как подсчитать информационный объём сообщения, содержащего видеоинформацию
- Как кодируют звук
- Как подсчитать информационный объём звукового сообщения

ВСПОМНИТЕ:

- Какие алфавиты используются человеком в различных языках, в том числе для написания математических текстов
- В чём различия между понятиями «свет» и «цвет»
- Каковы биологические особенности восприятия цвета и звука человеком

1 Кбайт = 1024 байт
1 Мбайт = 1024 Кбайт
1 Гбайт = 1024 Мбайт
1 Тбайт = 1024 Гбайт

Компьютер как техническое устройство обрабатывает информацию, записанную в двоичном алфавите. Обсудим, как основные виды информации — символьная, визуальная и звуковая — могут быть записаны всего лишь двумя символами.

КОДИРОВАНИЕ СИМВОЛЬНОЙ ИНФОРМАЦИИ Напомним, что символьная информация всегда представлена записью в некотором алфавите. Поэтому достаточно каждый символ алфавита закодировать некоторым словом, записанным посредством двух символов.

Ясно, что слов длины 1, записанных в двухсимвольном алфавите {0; 1}, ровно два — это 0 и 1. Слов длины 2 уже четыре — это 00, 01, 10 и 11; слов длины 3 будет восемь — это 000, 001, 010, 011 и 100, 101, 110, 111. Заметьте: с увеличением длины слова на 1 количество слов в двухсимвольном алфавите увеличивается вдвое. Поэтому **слов длины n, записанных в двухсимвольном алфавите, имеется 2^n**.

В латинском алфавите 26 символов, а пятисимвольных слов из 0 и 1 уже 32, т.е. достаточно, чтобы каждой букве латинского алфавита присвоить свой пятизначный код. Но ведь буквы бывают не только строчные, но и прописные. Чтобы их различать, требуется дополнительный символ в коде. А ведь есть ещё цифры, знаки препинания и арифметических действий, специальные знаки (типа №) и, конечно, особо популярный во всех текстах символ «пробел». Семисимвольных слов хватит уже и для того, чтобы закодировать сообщения не только на английском, но и на хорошем русском языке. Именно таков отечественный код КОИ–7. Сокращение КОИ родилось из первых букв словосочетания «код обмена информацией». Чтобы не употреблять длинный оборот «последовательность из стольких знаков, каждый из которых нуль или единица», люди договорились появление одного такого знака в последовательности называть **битом** (от английского BInary digiT — двоичная цифра). Поэтому говорят, что КОИ–7 — это семибитное кодирование сообщений.

В большинстве первых компьютеров использовался семибитный код. Однако с развитием техники это стало довольно неудобно. Новый код был уже восьмибитным и основывался на американском стандартном коде информационного обмена ASCII — American Standard Code for Information Interchange.

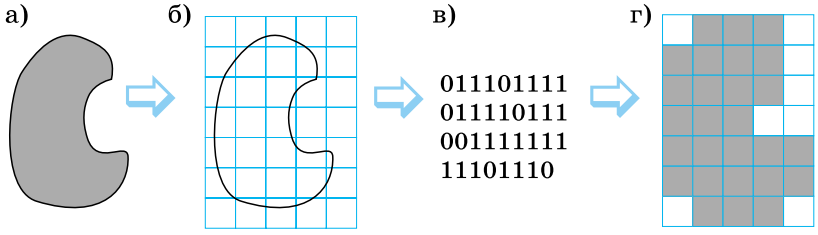
Последовательность восьми битов договорились называть словом **байт**. Но и один байт окажется маловат, если требуется оценить, сколько места в памяти компьютера занимает, скажем, десяток страниц текста. Поэтому были введены более крупные единицы — **килобайт** (обозначение Кбайт), **мегабайт** (Мбайт), **гигабайт** (Гбайт), **терабайт** (Тбайт) и т.д. Каждая из этих единиц в 1024 раза больше предыдущей.

В современном мире, опутанном компьютерными сетями, даже восьмибитного кодирования недостаточно: ведь есть арабский алфавит, два японских, хинди, математическая символика и т.д. Поэтому был введён новый стандарт

символьного кодирования — **UNICODE**, где каждый символ кодируется уже двумя байтами.

КОДИРОВАНИЕ ВИДЕОИНФОРМАЦИИ Чтобы компьютер мог обрабатывать видеоинформацию, она должна быть представлена в двоичном коде. Для этого поступают так.

Пусть для начала речь идёт о чёрно-белом рисунке. Представим себе изображённым его на прямоугольном листе бумаги. Разобьём этот лист на квадратики с достаточно маленькой стороной. А теперь про каждый квадратик мы можем сказать, есть в нём хоть какая-то часть рисунка или он чисто белый. В первом случае присваиваем квадратику код 1, во втором — код 0. Пробегая по всем квадратикам, например слева направо, и затем переходя по строкам сверху вниз, мы получаем двоичный код рисунка. Затем происходит восстановление рисунка, т.е. его декодирование.

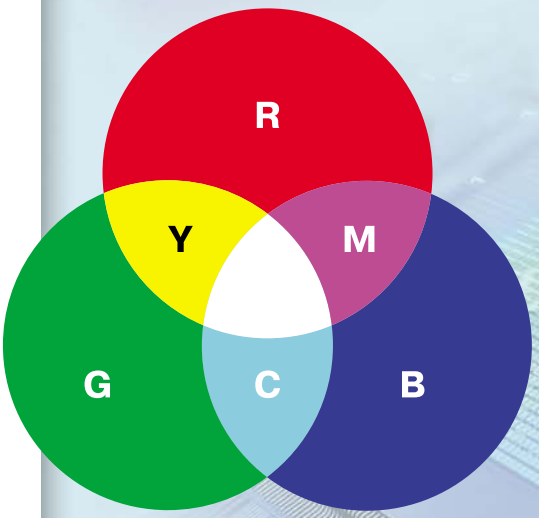


Конечно, результат декодирования почти наверняка не совпадёт с исходным рисунком. Но для того мы и выбирали размеры квадратиков достаточно маленькими, чтобы искажения были незначительными. Кроме того, как вы, наверное, помните из биологии и физики, и человеческий глаз, и любой другой оптический прибор имеет ограниченную разрешающую способность в восприятии изображения, т.е. перестаёт различать соседние участки как различные, если их размеры меньше этой разрешающей способности. В этом случае отдельные квадратики в восприятии человека сливаются в непрерывное целое.

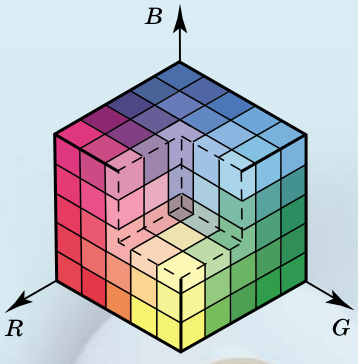
А как быть, если требуется закодировать цветной рисунок? Ведь там для описания цвета квадратика двух символов мало. И здесь природа снова подсказывает решение. Ведь хорошо известно, что любой цвет может быть получен как смесь трёх, как говорят, основных цветов — красного, синего и зелёного. Из таблицы вы легко поймёте, какие получаются цвета при таком смешивании; в таблице символ 1 означает наличие луча данного цвета, символ 0 — его отсутствие. А на рисунке смешивание цветов можно наблюдать воочию.

Здесь, правда, мы имеем только две градации яркости — горит/не горит. Но луч может гореть вполнакала; тогда появится и соответствующий оттенок получаемого цвета. Если у нас только две градации яркости, кодируемые числами 0 и 1, то мы можем получить всего лишь 8 различных цветов. А если градаций для каждого основного цвета будет, например, 256, то нетрудно подсчитать, что различных цветов получится $256^3 = 16\,777\,216$. Число 256 удобно тем, что оно является степенью числа 2, а именно восьмой степенью, поэтому указанное число градаций кодируется однобайтовыми последовательностями, и, значит, для кодирования 16 777 216 цветов потребуется 3 байта.

Красный	Зелёный	Синий	Цвет
0	0	0	Чёрный
1	0	0	Красный
0	1	0	Зелёный
0	0	1	Синий
0	1	1	Голубой
1	1	0	Жёлтый
1	0	1	Пурпурный
1	1	1	Белый



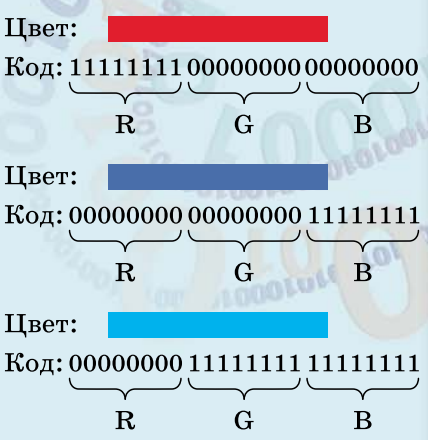
Открытие того, что с помощью трёх цветов можно получить любой цветовой оттенок, принадлежит М. В. Ломоносову. Он установил это в своих экспериментах по производству цветных стёкол для мозаики. Теоретическое обобщение этого открытия было сделано Германом Грассманом.



Куб RGB-кодирования

В основе кодирования цвета лежит его разложение на три составляющие. Цветовые оттенки определяются уровнем яркости каждого из составляющих цветов.

Для кодирования уровня яркости одного цвета строится шкала яркости – интервал от 0 до 1 разбивается на несколько равных частей. При выделении 1 байта на кодирование яркости одного цвета шкала имеет 256 градаций.



Кодирование в True-Color

Сказанное позволяет считать, что любой цвет задаётся тройкой чисел (a, b, c), показывающих, в каком соотношении нужно взять эти три основных цвета. Договоримся, что каждое из чисел меняется в диапазоне от 0 до 1, где числу 1 соответствует максимально возможная яркость источника света, передающего данный цвет, а числу 0 соответствует отсутствие луча света, несущего данный цвет. Указанные три числа можно рассматривать как код любого цвета — ведь они его однозначно определяют. Такой способ кодирования называют **RGB-кодированием** по первым буквам английских названий трёх основных цветов: Red — красный, Green — зелёный и Blue — синий. Описание цвета тройкой чисел наводит на мысль считать эти числа координатами точки в пространстве. Тогда получается, что коды всех цветов заполняют куб с ребром, равным 1. Яркость цвета определяется тем, насколько близка к 1 хотя бы одна из координат точки, соответствующей данному цвету. А чем ближе точка к диагонали, соединяющей чёрный и белый цвета, тем меньше яркость цвета, обозначенного этой точкой.

Уровень яркости каждого цвета — величина, конечно, непрерывная. Если же мы хотим, чтобы уровень яркости кодировался каким-то числом, то проще всего это сделать, разбив весь интервал яркости на заданное число частей. Иными словами, мы строим **шкалу яркости**. Такую замену непрерывно меняющейся величины на набор её отдельных значений называют её **дискретизацией**. На рисунке изображена дискретизация цветового куба для шести градаций яркости.

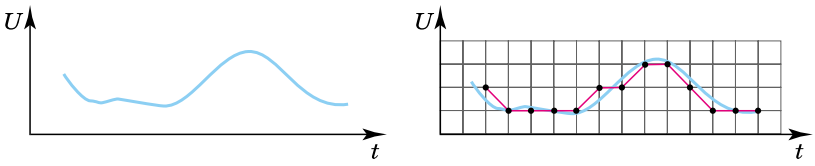
В современных компьютерах используется 16-битное кодирование (режим Hi-Color) и 24-битное кодирование (режим True-Color). В первом случае можно закодировать $2^{16} = 65\,536$ цветов, во втором — $2^{24} = 16\,777\,216$ цветов. В режиме True-Color на кодирование градаций яркости каждого из основных цветов отводится 1 байт: код 00000000 показывает, что данного цвета нет вообще, а код 11111111 соответствует наибольшей интенсивности (яркости) кодируемого цвета.

В случае 16-битного кодирования ситуация иная: на кодирование яркости красного и синего цветов отводится по 5 бит, а на кодирование зелёного — оставшиеся 6 бит. Поэтому в данном режиме шкала яркости для зелёного цвета содержит в 2 раза больше градаций, чем для красного и синего.

Кодирование цвета в режиме True-Color абсолютно обеспечивает качество цветовоспроизведения. Изменение в коде значения одного бита даёт настолько незначительное изменение цвета фигуры, что человеческий глаз, как правило, его не улавливает. Если бы мы на рисунке попытались изобразить для каждого цвета 256 градаций яркости вместо 6, то никакой дискретизации вы бы просто не увидели!

КОДИРОВАНИЕ ЗВУКА С XIX в. известны устройства, которые улавливают звуковые волны и преобразуют их в электрический ток. Устройства эти называются микрофонами. Точно такой же непрерывный, по-другому, *аналоговый*, электрический сигнал подаётся в звуковые усилители, а затем на динамики — для воспроизведения звука. Задача, следовательно, состоит в том, чтобы придумать способ кодирования

аналогового сигнала, такого, какой, например, изображён на рисунке. Для этого поступим так. Разобьём плоскость вертикальными и горизонтальными линиями и, заменяя плавную линию ломаной, будем приближённо считать, что график проходит через узлы получившейся сетки.



Вертикальное разбиение называется **дискретизацией** (по времени) аналогового сигнала и характеризуется **частотой дискретизации**. Для реального процесса это означает, что мы несколько десятков, сотен или тысяч раз в секунду измеряем величину аналогового сигнала.

Обычно для кодирования звука выбирают одну из трёх частот — 44,1 КГц, 22,05 КГц или 11,025 КГц. Первая из этих частот соответствует музыке, записываемой обычно на компакт-диск. Величина аналогового сигнала измеряется в этом случае 44 100 раз в секунду. Естественно, что в случае стереозвучания отдельно кодируется левый и правый канал. Возвращаясь к рисунку, изображающему процесс оцифровки звукового сигнала, можно сказать, что расстояние между вертикальными линиями соответствует 1/44 100 секунды.

Разумеется, важно и разбиение горизонтальными линиями. Чем мельче сетка, тем качественнее мы приблизим аналоговый сигнал числами. Горизонтальная разбивка называется **квантованием** (по уровню). Обычно применяется всего два типа квантования — 8-битное или 16-битное. В первом случае мы получаем 256 различных значений напряжения, во втором — уже известные вам 65 536 значений. Иными словами, интервал от нулевого до максимального напряжения разбивается либо на 256, либо на 65 536 уровней.

Восьмибитное квантование применяется для оцифровки обычной речи (в частности, телефонного разговора); шестнадцатибитное — для оцифровки музыки.

Подсчитаем, например, сколько будет занимать в килобайтах одна минута записи. При шестнадцатибитном кодировании сигнала потребуется 2 байта на запись значения напряжения. Поскольку сигнал измеряется 44 100 раз в секунду, надо $2 \times 44\,100 = 88\,200$ байт на каждую секунду. За минуту это составит $60 \times 88\,200 = 5\,292\,000$ байт. Поскольку сигнал стереофонический, это значение нужно умножить на 2. Получаем 10 584 000 байт. Чтобы выразить это число в мегабайтах, разделим его дважды на 1024. С округлением до десятых получаем 10,1 Мбайт.

ПРИНЦИП УНИВЕРСАЛЬНОСТИ ДВОИЧНОГО КОДИРОВАНИЯ Мы обнаружили, что **независимо от того, в каком виде представлена информация, её можно как угодно точно закодировать последовательностями из двух символов**. Это утверждение называют принципом универсальности двоичного кодирования. Его сформулировал американский учёный Клод Шеннон в середине XX в. Позже мы приведём дополнительные аргументы в пользу этого принципа.

ИССЛЕДОВАНИЯ

В объяснительном тексте мы подсчитали объём памяти, необходимый для записи 1 мин стереофонического звучания музыки. Проведите вычисления и заполните таблицу:

Тип сигнала / Частота дискретизации	16 бит, стерео	16 бит, моно	8 бит, моно
44,1КГц			
22,05 КГц			
11,025 КГц			

В правой нижней клетке таблицы вы получите значение объёма памяти, достаточного для записи 1 мин телефонного разговора. Именно такой объём памяти требуется автоответчику для записи входящих звонков, в частности для программно установленного на компьютере.

ВОПРОСЫ:

- Каков коэффициент пересчёта байтов в килобайты; килобайтов в мегабайты? А коэффициент пересчёта битов в байты? Сколько байтов в одном мегабайте?
- Сколько символов можно закодировать, используя UNICODE?
- Какой сигнал называют аналоговым?
- Что такое дискретизация аналогового сигнала?
- В чём состоит принцип универсального кодирования?
- Какие цвета используются при RGB-кодировании?
- Как получается белый цвет на экране монитора?

5 КОДИРОВАНИЕ ЧИСЛОВОЙ ИНФОРМАЦИИ

ВЫ УЗНАЕТЕ:

- Какие бывают системы счисления
- Какая система счисления называется позиционной и что такое основание такой системы
- Сколько нужно цифр, чтобы записать число в позиционной системе счисления с заданным основанием
- Как записываются числа в позиционной системе счисления с произвольным основанием

ВСПОМНИТЕ:

- Как выполняются действия в десятичной системе счисления

Записывать числа люди учились с древности. Египетские иероглифы, вавилонская клинопись, цифры языка майя и т.д. На уроках математики вас познакомили с действующим сегодня способом записи чисел — десятичной позиционной системой счисления, по-другому, десятичной нумерацией. Но, кроме десятичной, используются и другие нумерации.

СИСТЕМЫ СЧИСЛЕНИЯ

Система счисления — это способ записи чисел с помощью заданного набора специальных знаков, называемых цифрами. Системы счисления бывают позиционные и непозиционные. В **непозиционной** системе счисления значение каждой цифры не зависит от её положения в записи числа. К непозиционным системам счисления относится римская нумерация. **Позиционной** называется система счисления, в которой значение каждой цифры определяется тем, какое место (позицию) она занимает в записи числа. Так, запись 23 означает, что это число составлено из 3 единиц и 2 десятков. Если мы поменяем позиции цифр, то получим совсем другое число — 32. Это число содержит 3 десятка и 2 единицы. Вклад двойки уменьшился в 10 раз, а вклад тройки в 10 раз возрос. Легко понять, что цифры десятичной записи числа — это просто коэффициенты его представления в виде суммы степеней числа 10, которое называют **основанием** данной системы счисления. Например:

$$38\ 054 = 3 \cdot 10^4 + 8 \cdot 10^3 + 0 \cdot 10^2 + 5 \cdot 10^1 + 4 \cdot 10^0.$$

На самом деле числа можно записывать как сумму степеней не только числа 10, но и любого другого натурального числа b, большего 1. Например:

$$38\ 054_b = 3 \cdot b^4 + 8 \cdot b^3 + 0 \cdot b^2 + 5 \cdot b^1 + 4 \cdot b^0.$$

Такую запись называют **представлением данного числа в системе счисления с основанием b**. Чтобы знать, в какой системе счисления записано число, основание системы (в данном случае это b) указывают нижним индексом справа от этого числа. Цифры, используемые при записи числа в системе счисления с основанием b, неотрицательны и меньше, чем b.

Количество цифр, используемых при записи в позиционной системе с основанием b, равно b.
Если основание $b \leqslant 10$, то в качестве цифр используются цифры обычной десятичной системы счисления от 0 до $b - 1$. Если же $b > 10$, то после цифры 9 последовательно применяются буквы латинского алфавита: A, B, C, D и т.д.

ДЕЙСТВИЯ НАД ЧИСЛАМИ В ПОЗИЦИОННОЙ СИСТЕМЕ СЧИСЛЕНИЯ

Главное удобство позиционного принципа состоит в том, что **действия над числами в такой системе счисления выполняются поразрядно** (вспомните, как вы складываете и умножаете, вычитаете и делите многозначные числа в десятичной системе исчисления). Всё, что нужно знать, — это таблицы сложения и умножения для однозначных чисел и, конечно, правила выполнения действий столбиком.

Самая простая из всех позиционных систем счисления, конечно, **двоичная** — ведь в ней всего две цифры: 0 и 1. И, значит, имеется только два однозначных числа. Поэтому в этой системе счисления очень просто выглядят таблицы сложения и умножения:

$0 + 0 = 0$ $\quad$ $0 \cdot 0 = 0$
$1 + 0 = 1$ $\quad$ $1 \cdot 0 = 0$
$0 + 1 = 1$ $\quad$ $0 \cdot 1 = 0$
$1 + 1 = 10$ $\quad$ $1 \cdot 1 = 1$

В этих равенствах лишь результат 10 выглядит удивительно. Но стоит только заметить, что в двоичной системе 10 — это $1 \cdot 2^1 + 0 \cdot 2^0$, и всё становится на свои места.

Вот несколько примеров выполнения действий над многозначными числами в двоичной системе счисления:

$$\begin{array}{r} 101101 \\ +\ 10111 \\ \hline 1000100 \end{array} \qquad \begin{array}{r} 110110 \\ -\ 11101 \\ \hline 11001 \end{array} \qquad \begin{array}{r} 1011 \\ \times\ 101 \\ \hline 1011 \\ 1011 \\ \hline 110111 \end{array} \qquad \begin{array}{r|l} 11011 & 1001 \\ \cline{2-2} -\ 1001 & 11 \\ \cline{1-1} 1001 & \\ -\ 1001 & \\ \cline{1-1} 0 & \end{array}$$

Присмотримся внимательнее к умножению столбиком. Умножая на 1, мы просто переписали число. Во втором множителе две единицы, поэтому первый множитель пришлось переписать дважды, предварительно сдвинув вторую запись влево на необходимое число разрядов. А затем осталось произвести сложение. Таким образом, умножение в двоичной системе сводится к переписыванию одного из множителей несколько раз с необходимыми сдвигами и последующими сложениями этих копий.

За простоту действий над числами в двоичной системе приходится расплачиваться тем, что запись даже совсем небольших чисел в ней оказывается весьма длинной. Ниже в таблице содержатся первые 16 натуральных чисел и их представление в двоичной системе счисления. Эти представления полезно помнить.

Десятичная система	Двоичная система
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000

Для компьютера все числа, естественно, представляются в двоичной системе счисления.

Первым двоичную систему счисления в своей механической вычислительной машине применил немецкий математик Г. В. Лейбниц.

Г. В. Лейбниц (1646–1716)

ИССЛЕДОВАНИЯ

- К какому изменению записи числа, представленному в двоичной системе, приводит его увеличение в 2 раза? А увеличение в 4 раза?
- К чему приводит увеличение в b раз числа, записанного в системе счисления с основанием b? Сформулируйте общее правило.

ВОПРОСЫ:

- Что такое система счисления? Что называют основанием позиционной системы счисления?
- Что называют представлением числа в позиционной системе счисления с данным основанием?
- Сколько цифр используется в записи чисел в позиционной системе счисления с основанием 256?
- Чем привлекательна двоичная система счисления?

6 ПЕРЕВОД ЦЕЛЫХ ЧИСЕЛ ИЗ ОДНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДРУГУЮ

ВЫ УЗНАЕТЕ:

- Как перевести число из десятичной системы счисления в систему счисления с произвольным основанием
- Как перевести в десятичную систему счисления число, записанное в системе с другим основанием
- Как быстро переводить числа из двоичной системы счисления в шестнадцатеричную и обратно

ВСПОМНИТЕ:

- Что означает запись $a_0a_1...a_{n-1}a_n$ некоторого числа в системе счисления с основанием b

Использование различных систем счисления требует умений переводить записи чисел из одной системы счисления в другую. Обсудим, как делать такие преобразования.

ПЕРЕВОД ИЗ ДЕСЯТИЧНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В СИСТЕМУ СЧИСЛЕНИЯ С ДРУГИМ ОСНОВАНИЕМ Пусть некоторое число c требуется записать в системе счисления с основанием b. Это значит, что

$$c = a_0 \cdot b^n + a_1 \cdot b^{n-1} + \ldots + a_{n-1} \cdot b^1 + a_n \cdot b^0,$$

причём каждый коэффициент a_i неотрицателен и меньше b. Из этого равенства видно, что последняя цифра a_n представляет собой остаток при делении числа c на b. А частное от такого деления используется для нахождения предпоследней цифры a_{n-1} — она получается как остаток при делении этого частного снова на b. И т.д.

Слева дан пример перевода десятичного числа 793 в шестеричную систему счисления.

793	6		
6	132	6	
19	12	22	6
18	12	18	3
13	12	4	
12	0		
1			

Результат: $793_{10} = 3401_6$.

Чтобы перевести любое число в b-ичную систему счисления, нужно выполнить несколько раз деление столбиком на b и затем прочитать получившиеся остатки справа налево.

ПЕРЕВОД ИЗ СИСТЕМЫ СЧИСЛЕНИЯ С ПРОИЗВОЛЬНЫМ ОСНОВАНИЕМ В ДЕСЯТИЧНУЮ Пусть теперь некоторое число c записано в системе счисления с основанием b. Это значит, что

$$c = a_0 \cdot b^n + a_1 \cdot b^{n-1} + \ldots + a_{n-1} \cdot b^1 + a_n \cdot b^0,$$

где a_0, a_1, ..., a_{n-1}, a_n — цифры числа c, используемые в данной системе счисления. Тогда надо просто каждую цифру записать соответствующим десятичным числом, а затем выполнить действия.

Например:

$$3B7_{12} = 3 \cdot 12^2 + 11 \cdot 12 + 7 = 571_{10},$$

поскольку $B_{12} = 11_{10}$.

ПЕРЕВОД ИЗ ДВОИЧНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ШЕСТНАДЦАТЕРИЧНУЮ И ОБРАТНО Двоичное представление чисел (и информации вообще), столь удобное для электронной техники, вряд ли можно признать удобным для человека. У любого программиста — так называется специалист, который разрабатывает программное обеспечение для компьютеров, — зарябит в глазах от нулей и единиц, если он попытается прочитать программу в том виде, как она представлена для компьютера. А такая потребность иногда возникает. Поэтому программисты решили в четыре раза сократить количество знаков в записи машинных чисел, используя для этого шестнадцатеричную систему счисления. Впрочем, оказалось, что шестнадцатеричная система удобна и специалистам по компьютерной графике, и кодированию звука, и администраторам компьютерных сетей.

Прежде всего заметим, что для записи чисел в шестнадцатеричной нумерации требуется 16 цифр. Какие именно — показано в таблице.

Десятичная система	Двоичная система	Шестнадцатеричная система
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10

Пусть число 100111011 записано в двоичной системе счисления (в десятичной системе это 315):

$$100111011_2 = 1 \cdot 2^8 + 0 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + \\ + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0.$$

Сгруппируем слагаемые по четыре, рассматривая эту сумму справа налево и вынося общий множитель:

$$(1 \cdot 2^0) \cdot 16^2 + (0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0) \cdot 16^1 + \\ + (1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0) \cdot 16^0.$$

Посмотрите теперь в таблицу и вы увидите, что в каждой скобке фактически записана цифра шестнадцатеричной системы счисления. Наше число равно $13B_{16}$. Отсюда вытекает простое правило перевода чисел из двоичной системы счисления в шестнадцатеричную: двоичное число разбивается на четвёрки цифр справа налево, и каждая такая четвёрка заменяется шестнадцатеричной цифрой. Такие четвёрки называются **тетрадами**. Вот как выглядит этот процесс для того же числа 100111011:

$$\underset{1}{\underline{1}}\ \underset{3}{\underline{0011}}\ \underset{B}{\underline{1011}}$$

Легко выполнять и обратный перевод из шестнадцатеричной системы счисления в двоичную — для этого каждую цифру надо заменить на соответствующую в тетраду.

Шестнадцатеричная система счисления по сути та же двоичная, но в ней каждая группа из четырёх цифр заменяется всего одним символом. Такое превращение двоичного кода в шестнадцатеричный легко выполнить в уме, поэтому его используют для записи кодов не только чисел, но и любой информации. Например, шестнадцатеричный код красного цвета (с. 13–14) выглядит так: FF0000.

Чтобы перевести число из двоичной системы счисления в шестнадцатеричную, нужно запись числа разбить на тетрады справа налево и каждую тетраду заменить шестнадцатеричной цифрой.

Перевод чисел из шестнадцатеричной системы счисления в двоичную осуществляется заменой каждой цифры двоичной тетрадой.

ИССЛЕДОВАНИЯ

По аналогии с переводом чисел из двоичной системы счисления в шестнадцатеричную исследуйте возможность похожего метода перевода из двоичной системы счисления в восьмеричную. Совет: рассмотрите разбиение двоичной записи числа на тройки цифр — их называют триадами.

ВОПРОСЫ:

- Как выполнить перевод числа из десятичной системы счисления в систему счисления с другим основанием?
- Как перейти от записи числа в шестнадцатеричной системе счисления к записи в двоичной системе и обратно?
- Какое максимальное число воспринимает компьютер, если в двоичном коде он может оперировать только с семибитными последовательностями?

7

МЕТОД ДИХОТОМИИ. ИЗМЕРЕНИЕ КОЛИЧЕСТВА ИНФОРМАЦИИ

ВЫ УЗНАЕТЕ:

- Как ответить на вопрос, много ли информации содержит данное сообщение
- Как связаны понятия «информационный объём сообщения» и «количество информации в сообщении»
- Что такое алфавитный и семантический подходы к понятию количества информации

ВСПОМНИТЕ:

- Каковы единицы измерения информационного объёма сообщения
- Как связана величина информационного объёма с длиной сообщения, представленного в двоичном коде

С единицами измерения информационного объёма вы уже знакомы — биты, байты, килобайты и т.п. Но нужно понимать, что введение указанных единиц преследовало чисто технические цели, никак не отражая смысловое содержание информационного объекта. К примеру, если на диск дважды записано одно и то же сообщение, то места на нём занято в 2 раза больше, чем в том случае, когда на диск сообщение записано однократно. Значит, и в байтах мы получим число в 2 раза большее, чем для одного сообщения. С другой стороны, из повторного сообщения мы никакой новой (по смыслу) информации не имеем, так что количество информации остаётся тем же, что и после получения первого сообщения. И вообще, если в сообщении говорится о том, что уже известно получателю информации, то естественно считать, что количество информации у получателя не изменилось. Впрочем, если сообщение не может быть понято получателем информации — не важно, в силу недостатка знаний у получателя или искажений в ходе передачи информации, — то оно тоже не несёт для него информацию. Обсудим, как связаны технический и семантический (т.е. смысловой) подходы в измерении количества информации.

МЕТОД ДИХОТОМИИ Прежде чем ответить на поставленный вопрос, давайте рассмотрим игру «Угадайка». Правила игры таковы.

Один игрок задумывает целое число из заранее определённого диапазона, например от 1 до 16. Второй игрок, задавая такие вопросы, на которые первый может отвечать только «Да» или «Нет», должен выяснить, какое число было задумано. К примеру, можно спросить: «Верно ли, что задуманное число равно числу 7?» или «Больше ли задуманное число числа 3?».

Конечно, задуманное число можно выяснить, задав не более 16 вопросов, спрашивая последовательно, равно ли задуманное число 1, равно ли оно 2 и т.д. На какой-то из вопросов ответ будет «Да». Но не исключено, что это будет шестнадцатый по счёту вопрос. Впрочем, последний, шестнадцатый, вопрос можно было и не задавать: если на все предшествующие вопросы был получен ответ «Нет», то задумано число 16. А за какое наименьшее число вопросов можно гарантированно определить задуманное число?

Поступим так. Разделим наши числа от 1 до 16 на две равные группы, например от 1 до 8 и от 9 до 16. Вопросом «Верно ли, что задуманное число больше 8?» мы заставим первого игрока сказать нам, в каком из этих двух интервалов находится задуманное им число. Вторым вопросом мы этот интервал разделим ещё раз пополам: если, скажем, ответ игрока на первый вопрос был утвердительным, то спросим: «Верно ли, что задуманное число больше 13?» Теперь задуманное число находится уже в интервале, состоящем всего лишь из четырёх чисел. Следующий, третий

вопрос разделит этот интервал ещё раз пополам. Из оставшихся двух чисел узнать задуманное можно, задав всего лишь один вопрос. Итак, четырёх вопросов здесь оказалось достаточно, чтобы определить задуманное число. Можно доказать, что за меньшее число вопросов гарантированно угадать число не получится.

Нетрудно понять, что если исходный диапазон чисел был бы от 1 до 32, то хватило бы пяти вопросов для гарантированного угадывания задуманного числа. Если исходный диапазон от 1 до 64, то хватило бы шести вопросов. И т.д.

Но в «Угадайку» можно играть не только с числами. Можно расшифровывать названия загаданных животных или фамилии знаменитых писателей. Можно разгадывать, в конце концов, любую фразу. И вообще, задавая вопросы, на которые даются лишь ответы «Да» и «Нет», можно узнать любые сведения, т.е. *смысловое содержание информации*.

Метод, при котором множество объектов делится на две взаимоисключающие части, называется **дихотомией** (от греческого διχο — надвое, τομία — деление). Его применение оптимально, если получающиеся после этого множества близки по числу элементов.

КОЛИЧЕСТВО ИНФОРМАЦИИ КАК СНЯТИЕ НЕОПРЕДЕЛЁННОСТИ

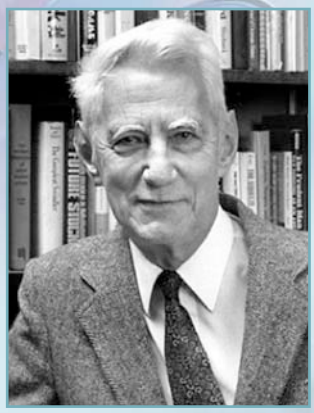
Ситуацию в игре «Угадайка» можно описать так. В начале игры неопределённость в выборе задуманного числа составляла 16 условных единиц — любое число из 16 могло оказаться задуманным. После первого вопроса неопределённость уменьшилась вдвое — задуманным теперь могло оказаться любое из 8 чисел. После второго вопроса неопределённость уменьшилась ещё вдвое и т.д. Иными словами, ответ на каждый вопрос давал нам информацию, вдвое уменьшающую неопределённость. Поэтому в информатике договорились принять **количество информации, уменьшающей вдвое неопределённость исходной ситуации, равным одному биту**. Количество информации, уменьшающей неопределённость в 4 раза, равно двум битам, уменьшающей в 8 раз — трём битам и т.д.

Вообще, если полученная информация уменьшает неопределённость в 2^n раз, то говорят, что количество этой информации равно n бит. Иными словами, если необходимо выделить один объект из множества, содержащего 2^n равноправных объектов, мы должны располагать n бит информации об этом объекте.

СВЯЗЬ МЕЖДУ АЛФАВИТНЫМ И СЕМАНТИЧЕСКИМ ПОДХОДАМИ К ИЗМЕРЕНИЮ КОЛИЧЕСТВА ИНФОРМАЦИИ

На с. 12–15 измерение информационного объёма сообщения основывалось на кодировании сообщения в двухсимвольном алфавите — наличие одного символа оценивалось в 1 бит информации. Такой подход к измерению информации называют **алфавитным**. Почему же и при алфавитном, и при семантическом подходах используются одни и те же единицы — биты, байты, килобайты?

Алфавитный подход к измерению информационного объёма важен уже потому, что всю информацию, циркулирующую в компьютере и хранящуюся в его памяти, можно представлять себе закодированной в двухсимвольном алфавите, например в виде последовательностей, составленных из 0 и 1.



К. Шеннон (1916–2001)

Первым понятие количества информации ввёл американский учёный Клод Шеннон. Занимаясь проблемой передачи информации по каналам связи, он заложил основы теории информации, которая теперь является одним из краеугольных камней информатики.

То же самое можно сказать и об информации, передаваемой по каналам связи. Сама схема передачи информации ещё раз представлена на рисунке. Под **приёмником** информации мы здесь понимаем человека или автоматическое устройство, которое просто фиксирует сигналы, поступающие к нему по каналу связи. Он при этом не обязан вникать в смысл передаваемой информации, да может и просто не знать код, которым закодировано исходное сообщение. Тем самым перед получением очередного сигнала приёмник информации находится в состоянии неопределённости — придёт сигнал, соответствующий символу 0, или сигнал, соответствующий символу 1. Когда сигнал приходит, неопределённость уменьшается вдвое (или полностью исчезает), то есть приёмник информации получает, как мы выше договорились, 1 бит. Получив последовательность из n символов двоичного алфавита, приёмник информации получил n бит. Вот мы и пришли к тому же выводу, который ранее предлагался в качестве определения количества информации при алфавитном подходе.

А можно ли, наоборот, исходя из кодирования, прийти к смысловому пониманию количества информации? Чтобы ответить на этот вопрос, проведём следующий эксперимент. Дабы он был не слишком утомительным, ограничимся числами от 0 до 15. Пусть задумано какое-нибудь число из этого интервала (к примеру, число 5). А теперь, задавая вопросы оптимальным образом, попытаемся угадать это число. Договоримся только, что все вопросы будут иметь форму «Верно ли, что задуманное число больше ... ?». При этом, получив ответ «Да», будем записывать 1, а на ответ «Нет» писать 0. Вот эти вопросы и ответы для нашего числа:

Верно ли, что задуманное число больше 7?	Нет	0
Верно ли, что задуманное число больше 3?	Да	01
Верно ли, что задуманное число больше 5?	Нет	010
Верно ли, что задуманное число больше 4?	Да	0101

После четырёх вопросов число угадано — 5. Но самое главное — в правом столбце в тот момент, когда число угадано, появляется его запись в двоичной системе счисления! Иными словами, оказалось, что последовательность ответов на вопросы — это двоичное кодирование информации. Сами же вопросы (точнее, метод их генерирования; в нашем случае — оговорённая заранее форма вопроса) определяют способ кодирования. Верно и обратное: если способ кодирования задан, то вопросы можно задавать не последовательно, формулируя следующий в зависимости от ответа на предыдущий, а сразу. К примеру, в данной игре можно сформулировать вопросы так:

1. Верно ли, что в двоичной записи задуманного числа первая цифра 1?
2. Верно ли, что в двоичной записи задуманного числа вторая цифра 1?
3. Верно ли, что в двоичной записи задуманного числа третья цифра 1?
4. Верно ли, что в двоичной записи задуманного числа четвёртая цифра 1?

ИССЛЕДОВАНИЯ

Проведите следующий эксперимент.

- Пусть задумано натуральное число n от 1 до 12. Используя метод дихотомии, определите для каждого n количество вопросов, потребовавшихся для угадывания этого числа (если интервал содержит нечётное количество чисел, то разделите его на части, отличающиеся на 1).
- Вычислите среднее арифметическое полученных вами количеств вопросов. Теория информации говорит, что это число должно быть близким к 3,585. А что получилось у вас?
- Поскольку для гарантированного угадывания разных чисел из одного и того же множества требуется разное количество вопросов, то среднее количество таких вопросов естественно принять за количество информации, необходимое для определения одного элемента из этого множества. Так появляются дробные количества бит.

Фактически именно эти вопросы и задавались в игре, но поскольку о двоичном кодировании договорённости не было, пришлось в неявном виде это кодирование проводить.

Последовательность вопросов задаёт кодирование, и если последовательность фиксирована, то по двоичному коду можно однозначно восстановить смысл всех ответов. Заметим, что задавая лишь конечное число вопросов, мы способны выявить не всю информацию об объекте, а только какую-то её часть. К примеру, играя в «Угадайку», мы пытаемся определить задуманное животное и уже выяснили, что это кит. Дальше мы можем выяснять, к какому виду принадлежит этот кит, каков его возраст, вес и т.д. Каждый объект описывается бесконечным числом свойств, мы же всегда ограничиваемся конечным набором тех из них, которые нам представляются существенными. Иными словами, процесс кодирования, сопровождающий получение информации, мы всегда обрываем на конечном шаге, удовлетворяясь подходящим для нас приближением.

Такая же ситуация возникает при построении двоичного кодирования звуковой и видеоинформации — для увеличения точности приходится наращивать длину кодирующего слова. По существу, мы снова имеем дело с принципиальным фактом теоретической информатики: **всегда существует двоичное кодирование, как угодно полно отражающее информацию об объекте.**

ИЗМЕНЕНИЕ ИНФОРМАЦИОННОГО ОБЪЁМА ПРИ УВЕЛИЧЕНИИ ЧИСЛА СИМВОЛОВ В АЛФАВИТЕ Как правило, человек для записи сообщений использует алфавит, в котором более двух символов. Если количество символов в алфавите совпадает со степенью числа 2, то всё просто — на кодирование каждого алфавитного символа тратится столько бит, каков показатель степени числа 2. Если, например, в алфавите 32 символа, то на кодирование каждого символа требуется 5 бит. Ясно, что в этом случае, чтобы найти информационный объём сообщения, нужно умножить количество символов в сообщении на 5 — ведь мы в сообщении, записанном символами рассматриваемого алфавита, просто каждый алфавитный символ заменяем на пятибитовую последовательность двоичных символов. Аналогично поступают и в тех случаях, когда количество символов в алфавите отличается от степени числа 2. А именно:

1) Определяют количество битов, необходимых для кодирования одного символа алфавита. Это число называют **информационной ёмкостью (объёмом) одного символа данного алфавита.** Оно равно показателю наименьшей целочисленной степени числа 2, которая больше или равна числу символов в алфавите. Например, если в алфавите используется 26 символов, то поскольку

$$2^4 = 16 < 26 < 32 = 2^5,$$

каждый символ кодируется 5 битами.

2) Умножают информационную ёмкость одного символа на количество символов в сообщении.

Полученное число и является информационным объёмом данного сообщения.

ВОПРОСЫ:

- Как связано уменьшение неопределённости с количеством полученной информации?
- Чем отличается алфавитный подход к измерению количества информации от семантического?
- Какие ограничения принимаются при алфавитном подходе к измерению количества информации?
- Что такое информационный объём символа алфавита?
- Как вычислить информационный объём одного символа в алфавите?
- Как найти информационный объём сообщения, если известен информационный объём одного символа алфавита, в котором записано данное сообщение?

ВСПОМНИТЕ:

- Основные информационные процессы
- Почему двоичное кодирование считается универсальным
- Каковы основные единицы измерения информационного объёма
- Как осуществляется перевод числа из десятичной системы счисления в позиционную систему счисления с произвольным основанием и обратно
- Как осуществляется перевод числа из двоичной системы счисления в шестнадцатеричную и обратно
- Как связано количество информации с уменьшением неопределённости

Н. Винер (1894–1964)

В. М. Глушков (1923–1982)

ТЕОРЕТИЧЕСКИЙ ПРАКТИКУМ

Чтобы быть уверенным в своих знаниях, надо не только уметь отвечать на вопросы о том, что вы уже изучили, но и использовать эти знания при выполнении различных заданий. Часть заданий помещена здесь, другие вы найдёте в тетради-практикуме.

ИНФОРМАЦИЯ И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ

1 Норберт Винер, один из основоположников информатики как науки, говорил об информации так: «Информация — это обозначение содержания, полученного из внешнего мира в процессе нашего приспособления к нему наших органов чувств». Сравните это определение с теми, которые приведены в тексте на с. 6. Что у них общего и чем они различаются?

2 Академик В. М. Глушков, теоретик и разработчик электронно-вычислительных машин (так раньше назывались компьютеры), писал: «Под информацией в современной науке принято понимать меру неоднородности распределения материи и энергии в пространстве и времени. При таком понимании информации оказывается возможным говорить, например, об информации, которую несёт солнечный луч, шум горного обвала, шорох листвы и т.п. Во всяком случае при этом не обязательно требовать осмысленности, с которой обычно неразрывно связывается понятие информации в её житейском понимании». Сравните это объяснение понятия «информация» с теми, которые приведены в тексте на с. 6: что между ними общего и чем они различаются. К какому из них это объяснение, по вашему мнению, ближе всего?

3 Приведите примеры получения информации: а) человеком; б) животными.

4 Человек изобрёл много различных способов хранения информации. Какие из них вы могли бы назвать?

5 Приведите примеры передачи информации
а) между людьми;
б) между человеком и животным;
в) между человеком и техническим устройством;
г) между двумя техническими устройствами.

6 Приведите примеры информационных процессов, в которых изменяется: а) содержание информации; б) форма представления информации.

УНИВЕРСАЛЬНОСТЬ ДВОИЧНОГО КОДИРОВАНИЯ. ЕДИНИЦЫ ИЗМЕРЕНИЯ ИНФОРМАЦИОННОГО ОБЪЁМА

7 а) Информационный объём сообщения равен 256 бит. Укажите объём этого сообщения в байтах.
б) Информационный объём сообщения составляет 2,5 Кб. Укажите объём этого сообщения в битах.

8 В сообщении

Информатика — наука современности!

каждый символ кодируется одним байтом. Каков информационный объём этого сообщения?

9 Для пяти букв русского алфавита заданы их двоичные коды: О — 001; П — 100; К — 10; М — 101; Т — 01. Какое слово русского языка закодировано двоичной строкой 1000110110000101?

10 Пусть используется режим True-Color. Укажите цвет, который задаётся кодом:
а) 000000001111111111111111;
б) 011111110111111101111111;
в) 111111110000000011111111.

11 а) Подсчитайте, сколько различных значений напряжения кодируется при использовании 24 битов для записи информации о квантовании звуковой информации.
б) Вычислите, какой объём памяти потребуется для записи 1 мин аудиоинформации при режиме 24 бита × 96 КГц.

СИСТЕМЫ СЧИСЛЕНИЯ

12 Переведите числа 579 и 2468 из десятичной системы счисления в систему счисления с основанием: а) 5; б) 7; в) 9.

13 Переведите в десятичную систему счисления числа 8669, 128 711, 4 529 712, 54 237, 742 689.

14 Переведите числа 1001, 10101, 111001, 10111101 из двоичной системы счисления:
а) в десятичную систему счисления;
б) в шестнадцатеричную систему счисления.

15 Переведите числа 25, 4F, 1A7, ABC, D1AE, FFFF из шестнадцатеричной системы счисления:
а) в десятичную систему счисления;
б) в двоичную систему счисления.

16 На с. 19 говорится, что для сокращения записи вместо двоичного кода используется шестнадцатеричный. Какой цвет в режиме True-Color кодируется последовательностью:
а) 00FF00; б) 0000FF; в) FFFFFF?

17 Нередко для большего удобства вместо двоичного кода используется десятичная запись. Так, при кодировании цвета каждые два байта, кодирующие оттенок одного из основных цветов, переводятся как число в десятичную систему счисления (см. рисунок).
а) Переведите в десятичную запись код цвета: 00FF00; AACC55; D0D0D0.
б) Переведите в двоичную запись десятичный код цвета: 127.255.0; 51.5.102.

ИЗМЕРЕНИЕ КОЛИЧЕСТВА ИНФОРМАЦИИ

18 Какое количество информации нужно получить, чтобы угадать:
а) одно из 64 первых натуральных чисел;
б) одно из 128 первых натуральных чисел?

19 Задумывается нечётное число от 1 до 63. Какое количество информации нужно получить, чтобы его угадать?

20 Каждый символ алфавита и символ «пробел» кодируется двоичной последовательностью одной и той же минимально возможной длины. Сообщение, записанное в этом алфавите, содержит 4096 символов, а его информационный объём равен 2 Кбайт. Какое количество символов в этом алфавите?

ИССЛЕДОВАНИЯ

Вам хорошо известны различные признаки делимости. Например, чтобы узнать, делится ли число на 3, надо найти сумму цифр этого числа и проверить, делится ли она на 3. Но это в десятичной системе счисления. А для каких чисел аналогичный признак можно сформулировать, если числа записываются, скажем, в семеричной системе счисления? Исследуйте, как трансформируются известные вам признаки делимости в зависимости от основания системы счисления.

Цвет:

Двоичный код:

$$\underbrace{00000000}_{R}\ \underbrace{00000000}_{G}\ \underbrace{11111111}_{B}$$

Десятичный код: 0.0.255

ВЫ УЗНАЕТЕ:

- Как с помощью компьютерного приложения Калькулятор можно переводить числа из десятичной системы счисления в двоичную и обратно
- Как с помощью приложения Калькулятор выполнять действия над числами в двоичной и шестнадцатеричной системах счисления

ВСПОМНИТЕ:

- Как записываются числа в двоичной и шестнадцатеричной системах счисления

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ

Выполнять задания лабораторных работ, помещённых в этом разделе главы, вы будете в компьютерном классе. Поэтому нелишне напомнить правила техники безопасности — ведь к каждому рабочему месту подведено опасное для жизни напряжение электричества.

1. Если вы обнаружили какую-либо неисправность, немедленно сообщите об этом преподавателю. Не работайте на неисправном оборудовании!
2. Не включайте и не выключайте компьютеры без указания учителя.
3. Не дёргайте и вообще не трогайте различные провода.
4. Не стучите по клавиатуре и мыши.

Для вашего здоровья важно и то, как вы сидите перед компьютером:

- Позвоночник должен быть вертикальным; спина — опираться на спинку стула.
- Ступни удобно стоят на полу или специальной подставке.
- Расстояние до экрана — не менее вытянутой руки.
- Верхний край экрана располагается примерно на уровне ваших глаз.
- Если вы посмотрите на центр экрана, то линия вашего взгляда должна быть перпендикулярной плоскости экрана, то есть экран должен быть развёрнут немного вверх.
- Плечи развёрнуты и опущены, вам не составляет труда удобно расположить руки над основными клавишами среднего ряда.

Хотя практикум называется компьютерным, это вовсе не означает, что, приступая к выполнению заданий лабораторной работы, нужно тут же включать компьютер. Мы советуем сначала внимательно прочитать задание и поразмышлять, как именно будет использоваться та или иная компьютерная технология для его выполнения.

ЛАБОРАТОРНАЯ РАБОТА № 1. СИСТЕМЫ СЧИСЛЕНИЯ С ОСНОВАНИЕМ, РАВНЫМ СТЕПЕНИ ЧИСЛА 2 Как вы знаете, для компьютера вся информация кодируется двумя символами. При этом для чисел обычно используют их запись в двоичной системе счисления. Но современное программное обеспечение создается дружелюбным к тем, кто им пользуется, и поэтому вся числовая информация любезно предоставляется в десятичной системе счисления. Впрочем, по вашему желанию вы можете увидеть представление числа и в двоичной системе. Да и не только в двоичной.

Откройте приложение *Калькулятор* в режиме, который указан в подписях к рисунку. Найдите на панели трёхбуквенные сочетания Hex, Dec, Oct и Bin. Это сокращения латинских слов, которые переводятся на русский как шестнадцатеричный, десятичный, восьмеричный и двоичный. Речь идёт, как вы понимаете, о системах счисления.

По умолчанию в *Калькуляторе* включается представление чисел в десятичной системе счисления.

1 Наберите число 19. Переключите калькулятор в режим Bin. На табло появилось представление этого числа в двоичной системе. Переведите число 19 в двоичную систему счисления без помощи компьютера. Сверьте свой ответ с ответом компьютера. Сошёлся ответ? Если нет, найдите у себя ошибку.

2 Переведите в двоичную систему счисления числа 44, 129, 561, 1322. Сверьте результат компьютера с вашим.

3 С помощью приложения *Калькулятор* переведите десятичные числа 19, 44, 129, 561, 1322 в шестнадцатеричную систему счисления. Сверьте компьютерные результаты с теми, которые получены вами.

4 А теперь переведите числа 1001, 10101, 111001, 10111101 из двоичной системы счисления в десятичную. Сначала сделайте это без помощи компьютера, а затем с помощью приложения *Калькулятор*. Сравните ваш результат с результатом компьютера.

5 С помощью *Калькулятора* и без него переведите числа 25, 4F, 1A7, ABC, D1AE, FFFF из шестнадцатеричной системы счисления в десятичную. Сравните ваш результат с результатом компьютера.

6 Сколько двоичных разрядов содержит табло *Калькулятора*? Какое самое большое двоичное число может быть записано на табло *Калькулятора*? Переведите это число в десятичную систему счисления с помощью *Калькулятора*.

7 Наберите на табло *Калькулятора* отрицательное десятичное число −19. Перейдите в двоичную систему счисления. Попытайтесь объяснить полученный результат.

8 Вычислите с помощью *Калькулятора* значения следующих выражений:

а) $11011010001_2 + 11101010001_2$;
б) $111011101_2 - 1101110110_2$;
в) $110101_2 \cdot 10101_2$.

Переведите каждый из результатов в десятичную систему счисления.

9 Вычислите с помощью *Калькулятора* значения следующих выражений:

а) $1A7_{16} + ABC_{16}$; б) $D1AE_{16} - FFFF_{16}$; в) $25_{16} \cdot A0C_{16}$.

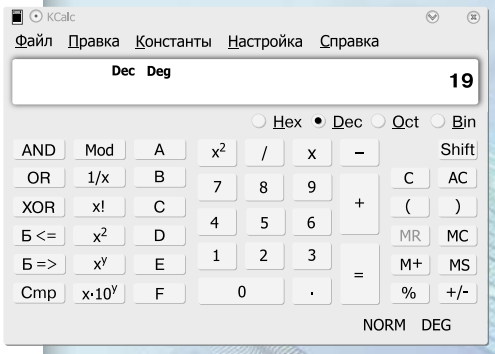
Переведите каждый из результатов в десятичную систему счисления.

10 Вычислите с помощью *Калькулятора* значения следующих выражений:

а) $11011010001_2 + CAB_{16}$; б) $D1AE_{16} + 111011101_2$.

Переведите каждый из результатов в десятичную систему счисления.
Закройте приложение.
Лабораторная работа завершена.

Приложение *Калькулятор* в ОС *Windows*



Приложение *KCalc* в ОС *Linux*

ВЫ УЗНАЕТЕ:

- Как влияет изменение кода цвета на его оттенок

ВСПОМНИТЕ:

- Как двоичный код преобразуется в шестнадцатеричный

ЛАБОРАТОРНАЯ РАБОТА № 2. КОДИРОВАНИЕ ЦВЕТА

Рассказывая о кодировании цвета, мы привели 24-битовые коды только трёх цветов (см. рисунок на с. 14). Но, конечно, интересно, каковы коды других цветов и, наоборот, какой цвет соответствует тому или иному коду. И в этом вам поможет программа Colors. Запустить её вы можете в любом браузере, который имеется на вашем компьютере. Надо только иметь в виду, что коды цветов для более экономной записи представлены в шестнадцатеричном виде (см. с. 14).

1 Сначала давайте поэкспериментируем с яркостью основных цветов. Каким будет цвет, если вместо кода **FF0000** взять, скажем, **AA0000**. Из текста на с. 14 следует, что это должен быть оттенок красного цвета. Но какой?

Обратимся к программе Colors. На экране она выглядит так, как изображено на рисунке внизу страницы.

Введите в диалоговое окно (оно отмечено знаком #) код **AA0000** и щёлкните по кнопке *Проверить*. Ниже вы увидите прямоугольник, окрашенный этим цветом, и рядом слова «Небезопасный цвет!». Не пугайтесь — ни вам, ни вашему компьютеру ничего не угрожает; позже мы поясним смысл этого предупреждения.

Безопасные цвета

Проверить цвет # [ ] Проверить

255.255.204 FFFFCC	255.255.153 FFFF99	255.255.102 FFFF66	255.255.51 FFFF33	255.255.0 FFFF00	204.204.0 CCCC00
255.204.102 FFCC66	255.204.0 FFCC00	255.204.51 FFCC33	204.153.0 CC9900	204.153.51 CC9933	153.102.0 996600
255.153.0 FF9900	255.153.51 FF9933	204.153.102 CC9966	204.102.0 CC6600	153.102.51 996633	102.51.0 663300
255.204.153 FFCC99	255.153.102 FF9966	255.102.0 FF6600	204.102.51 CC6633	153.51.0 993300	102.0.0 660000
255.102.51 FF6633	204.51.0 CC3300	255.51.0 FF3300	255.0.0 FF0000	204.0.0 CC0000	153.0.0 990000
255.204.204 FFCCCC	255.153.153 FF9999	255.102.102 FF6666	255.51.51 FF3333	255.0.51 FF0033	204.0.51 CC0033
204.153.153 CC9999	204.102.102 CC6666	255.51.51 CC3333	153.51.51 993333	153.0.51 990033	51.0.0 330000
255.102.153 FF6699	255.51.102 FF3366	255.0.102 FF0066	204.51.102 CC3366	153.102.102 996666	102.51.51 663333
255.153.204 FF99CC	255.51.153 FF3399	255.0.153 FF0099	204.0.102 CC0066	153.51.102 993366	102.0.51 660033
255.102.204 FF66CC	255.0.204 FF00CC	255.51.204 FF33CC	204.102.153 CC6699	204.0.153 CC0099	153.0.102 990066

Цвет оказался довольно тёмным. Попробуйте теперь код DD0000. Видите: цвет стал ещё темнее. Если же вы наберёте, скажем, 550000, то получите совсем тёмный цвет, который скорее можно назвать чёрным с красноватым отливом.

2 Проведите такие же эксперименты по уменьшению численного значения кода с основными цветами, которые кодируются как 00FF00 и 0000FF.

Пора объяснить, что означают слова «Безопасный цвет» и «Небезопасный цвет». Дело в том, что каждый браузер, загружая интернет-страницу, преобразует код цвета по-своему. Безопасные цвета — это те цвета, которые всеми браузерами отображаются одинаково. А вот при выборе небезопасного цвета для оформления, скажем, web-страницы можно получить совершенно не тот эффект, который был задуман её создателем. Все безопасные цвета перечислены в этой программе в виде таблицы, так что данная программа ещё и справочник по безопасным и небезопасным цветам.

3 А теперь поэкспериментируйте со смешиванием цветов. Результат одного из вариантов смешивания двух основных цветов — зелёного и синего — максимальной яркости приведён на рисунке на с. 14. Получился бирюзовый цвет. Таких парных комбинаций, естественно, три. Получите оставшиеся два цвета.

4 Для каждой пары основных цветов смешайте их, уменьшив яркость одной из двух составляющих по очереди. Сравните полученные результаты с теми, которые вы наблюдали в предыдущем эксперименте. Сделайте вывод, как при этом меняется цвет смеси?

5 Что будет происходить, если смешивать все три основных цвета? Убедитесь, что если яркость всех трёх основных цветов одинакова, то мы получаем оттенки серого цвета.

6 Число 05 составляет треть числа F. Нетрудно, однако, убедиться, что в зрительном восприятии яркость уменьшается вовсе не в три раза — если взят только один из основных цветов, то он воспринимается как почти чёрный. Проверьте это.
Если же это код одного цвета из участвующих в смешивании цветов, то его присутствие практически не сказывается на оттенке получающегося цвета. Проверьте и это утверждение.

7* Попытайтесь, не обращаясь к программе Colors, ответить на вопрос «Какой цвет кодируется как 999900?». Ответ может показаться довольно простым: поскольку FFFF00 — код жёлтого цвета, то это должен быть, скажем так, тёмно-жёлтый. Проверьте эту гипотезу с помощью Colors.
Выполните аналогичное задание для цвета с кодом 00DDDD. В чём, на ваш взгляд, причина расхождения между ответом, основанным на рассуждении, и ответом, полученным экспериментально?
Лабораторная работа завершена.

ПОДВЕДЁМ ИТОГИ

Информация — нематериальное отражение материального мира, социальных процессов и явлений, а также духовного мира человека. Информация может быть зафиксирована в текстовой, графической или звуковой форме, а также отдельным знаком (сигналом).

Информационным называется процесс, в ходе которого вырабатывается новая информация, изменяется содержание информации или форма её представления. Основные виды информационных процессов: получение, передача, сохранение и обработка (преобразование) информации.

Важнейшим средством передачи информации является язык. Существуют различные виды языков: коммуникативные, формализованные и формальные.

Для сохранения и передачи информации применяется её кодирование. В технических системах обработки, сохранения и передачи информации наиболее удобным оказывается двоичное кодирование.

ПОДРОБНЕЕ...

Гашков С. Б. Системы счисления и их применение. — М.: МЦНМО, 2004. — 52 с.

Гейн А. Г. Информатика и ИКТ. Задачник-практикум. 10—11 класс. — М.: Просвещение, 2019. — 157 с.

Гитт В. В начале была информация. — М.: ДИАЙПИ, 2013. — 352 с.

Информатика как наука об информации / Под ред. Р. С. Гиляревского. — М.: Гранд-Фаир, 2006. — 592 с.

Информатика: Энциклопедический словарь для начинающих/ Сост. Д. А. Поспелов. — М.: Педагогика-Пресс, 1994. — 352 с.

Лещинер В.Р., Крылов С. С., Ушаков Д. М. Я сдам ЕГЭ! Информатика. Типовые задания. — М.: Просвещение, 2019. — 240 с.

Моисеев Н. Н. Человек и ноосфера. — М.: Молодая гвардия, 1990. — 351 с.

Энциклопедия школьной информатики. — М.: БИНОМ: Лаборатория знаний, 2014. — 400 с.

ВОПРОСЫ ДЛЯ ОБСУЖДЕНИЯ:

- Чтение – это процесс получения информации или процесс её обработки?
- Всегда ли повторение изученного материала – это только процесс сохранения информации в памяти человека?
- В каких науках – гуманитарных или естественных – слова из коммуникативного языка чаще становятся терминами?
- Математику часто называют языком науки. Согласны ли вы с этим тезисом?
- В чём состоят положительные и отрицательные стороны избыточности языка?
- В чём преимущества и в чём недостатки цифрового кодирования аудиоинформации? Можно ли ожидать, что оцифрованный звук полностью вытеснит аналоговые способы записи и воспроизведения?

Википедия
http://ru.wikipedia.org/wiki/Информация

Глава 2

ОСНОВНЫЕ ИНФОРМАЦИОННЫЕ ОБЪЕКТЫ. ИХ СОЗДАНИЕ И КОМПЬЮТЕРНАЯ ОБРАБОТКА

ОБРАБОТКА ИНФОРМАЦИОННЫХ ОБЪЕКТОВ

ТЕКСТОВЫЙ РЕДАКТОР

ЭЛЕКТРОННАЯ ТАБЛИЦА

ГРАФИЧЕСКИЙ РЕДАКТОР

САМОЕ... САМОЕ...

Чтобы с информацией можно было работать, она должна быть зафиксирована соответствующим образом. Зафиксированную информацию называют информационным объектом. Объекты, которые можно обрабатывать с помощью компьютера, обычно называют оцифрованными. Овладение приёмами обработки оцифрованных информационных объектов — важная составляющая информационной грамотности.

8 КОМПЬЮТЕРНАЯ ОБРАБОТКА ТЕКСТА

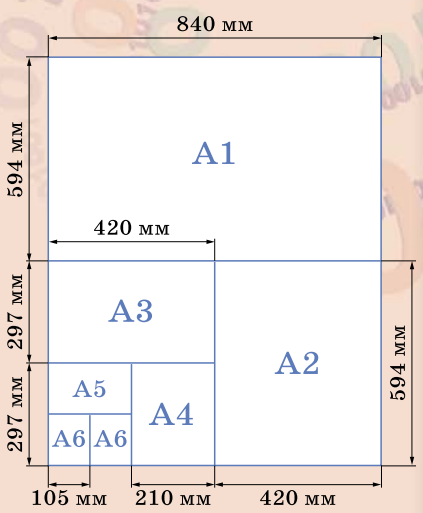
ВЫ УЗНАЕТЕ:

- Что следует понимать под термином «компьютерная обработка текста».
- Что такое форматирование текста.
- Как снабдить текст колонтитулом.
- Как изготовить компьютерный текст, если он уже есть на бумаге.

ВСПОМНИТЕ:

- Почему компьютер называют формальным исполнителем.

Редактирование текста – это его обработка с целью сделать более доступным для понимания и удобным для восприятия.



В «Словаре русского языка» С. И. Ожегова сказано: «Текст – всякая записанная речь». И совершенно неважно, записана ли она буквами, иероглифами, стенографическими значками или ещё как-нибудь. С позиций информатики такая формулировка означает, что текст – это просто последовательность знаков некоторого алфавита. Значит, обработка текста – это преобразование данной последовательности. Обсудим, какие именно преобразования текста можно осуществлять с использованием компьютера.

ФОРМАЛЬНЫЕ ХАРАКТЕРИСТИКИ ТЕКСТА Компьютер, как вы помните, — формальный исполнитель, и поэтому ему доступна только формальная обработка текста. В чём же в таком случае состоит формальное преобразование текста? В том, что исправляются орфографические, синтаксические и стилистические ошибки, удаляются ненужные повторы, вводятся уточняющие слова и т.п. Текст структурируется, т.е. разбивается на разделы — главы, параграфы, пункты — и абзацы.

Наконец, в текст вставляются иллюстрации — чертежи, рисунки, диаграммы, графики и т.п. Структурные элементы текста тем или иным образом располагаются на странице — центрируются заголовки, определённым образом располагаются эпиграфы к главам или параграфам, выравниваются (или, наоборот, не выравниваются) края текста, выделяются каким-либо образом основные положения и т.д.

Подобные преобразования текста обычно называют его **редактированием**, а компьютерное программное обеспечение, используемое для обработки текстов в электронном виде, называют в зависимости от предоставляемых возможностей текстовыми редакторами, текстовыми процессорами или программами верстки. Каждый текстовый редактор позволяет вставлять новые слова. Можно стирать отдельные буквы и переставлять целые абзацы, автоматически заменять во всём тексте одно слово другим. Многие редакторы текстов умеют автоматически разбивать текст на страницы и нумеровать их.

Текст состоит из символов, символы складываются в слова, слова — в предложения, предложения — в абзацы. Сам текст распределяется по страницам. *Страница*, *абзац* и *символ* — вот те объекты, о задании параметров которых надо прежде всего задуматься, создавая текст.

Расположение текста на странице определяется размерами страницы и полями. Обычно страница имеет размеры 210 × 297 мм. Это так называемый формат А4. Половина такой страницы называется форматом А5. Если же две страницы формата А4 соединить по длинной стороне, получится лист формата А3. Легко догадаться, как получается лист форматов А2 или А6. Применяются листы и других размеров, их параметры и маркировку вы посмотрите, выполняя лабораторную работу. Страница может быть книж-

ной (вертикальной) или альбомной (горизонтальной) ориентации.

Поля страницы определяют расположение текста относительно краёв листа. Это, однако, не означает, что вы не сможете ничего разместить на полях.

На боковых полях текст можно разместить, если раздвинуть ограничители текстового поля, расположенные на горизонтальной линейке (они имеют форму треугольничков). Текстовое поле можно расширить и с помощью диалогового окна *Абзац*. На вкладке *Отступы и интервалы* достаточно указать левый отступ (отрицательный, если надо, чтобы текст выдвинулся на левое поле) и правый отступ (положительный, если надо, чтобы текст выдвинулся на правое поле).

Текст, выносимый на верхнее и нижнее поля, называют **колонтитулом**. Обычно в колонтитуле указывают название текущего раздела, главы или параграфа. Этот текст будет воспроизводиться на всех страницах в пределах одного раздела, но предусмотрено, что может быть различным на чётных и нечётных страницах. Туда же, в колонтитул, выставляются номера страниц (колонцифры).

Текст на странице можно расположить в несколько колонок — назначить количество колонок, их ширину, расстояние между ними.

Теперь поговорим о следующем объекте — абзаце. Об одном параметре абзаца — величине отступов от левого и правого полей — мы уже сказали выше. Ещё один параметр — выравнивание текста. Оно отражает вид границы текста в абзаце.

Выравнивание по левому краю означает, что первые символы всех строк абзаца (кроме, быть может, первой) находятся на одной вертикальной прямой.

Аналогично, **выравнивание по правому краю** означает, что последние символы всех строк абзаца, кроме последней, располагаются вдоль одной вертикальной прямой.

Выравнивание по ширине — это одновременное выравнивание по левому и правому краям. Наконец, выравнивание по центру означает, что каждая строка размещается симметрично относительно вертикальной прямой, делящей страницу пополам.

Выравнивание по центру применяется обычно для заголовков, вынесенных в отдельную строку формул, рисунков и других элементов текста, к которым требуется привлечь особое внимание.

Для первой строки абзаца может быть применено отдельное форматирование. Эта строка может быть сделана **красной**, т.е. набираться с отступом вправо, или **висячей**, т.е. с выступом относительно расположения остальных строк абзаца.

Строки внутри абзаца могут располагаться на разном расстоянии друг от друга. Это расстояние называется **межстрочным интервалом**. Величина интервала измеряется в **пунктах** (здесь пунктом называется не раздел текста, а специальная единица длины, применяемая в издательском деле: 1 пункт = 0,376 мм). Интервал может быть одинарным (рав-

Выравнивание текста

Текст, выносимый на верхнее и нижнее поля, называют **колонтитулом**. Обычно в колонтитуле указывают название текущего раздела, главы или параграфа.	Текст, выносимый на верхнее и нижнее поля, называют **колонтитулом**. Обычно в колонтитуле указывают название текущего раздела, главы или параграфа.	Текст, выносимый на верхнее и нижнее поля, называют **колонтитулом**. Обычно в колонтитуле указывают название текущего раздела, главы или параграфа.
По левому краю	По центру	По правому краю

Первая строка

Для первой строки абзаца может быть применено отдельное форматирование.	Для первой строки абзаца может быть применено отдельное форматирование.
Отступ	Выступ

Величина межстрочного интервала, интервала между абзацами и высота символов текста измеряется в **пунктах**.

ным по величине размеру используемых заглавных букв), полуторным (в 1,5 раза больше одинарного), двойным (в 2 раза больше одинарного) и вообще любым. В последнем случае просто устанавливают величину множителя в диалоговом окне *Абзац*.

Интервал между абзацами также можно менять. Надо только иметь в виду, что значение 0 соответствует межстрочному интервалу внутри абзаца. Любое натуральное значение этого параметра показывает, на сколько интервал между абзацами больше межстрочного интервала внутри абзаца (предыдущего или последующего).

Расположение текста на странице называют его **форматом**. Поэтому задание (или изменение) тех или иных рассмотренных выше параметров текста называют **форматированием**.

Иногда требуется некоторую совокупность абзацев пронумеровать. Эту формальную процедуру тоже можно поручить компьютеру. Для этого, выделив подлежащие нумерации абзацы, щёлкните по кнопке *Нумерация*. Если какой-то из выделенной совокупности абзац надо исключить из нумерации, то установите курсор внутри этого абзаца и ещё раз щёлкните по кнопке *Нумерация*. Абзац окажется ненумерованным, зато последующие получат номер, на 1 меньший.

Вместо номера в начале абзаца можно поставить тот или иной символ, называемый **маркером**. В качестве маркера могут выступать разные символы, например: –, •, ✤, ⇒.

Списки

1. Верно ли, что первая цифра 1?
2. Верно ли, что вторая цифра 1?
3. Верно ли, что третья цифра 1?
4. Верно ли, что четвёртая цифра 1?

Нумерованный список

- Верно ли, что первая цифра 1?
- Верно ли, что вторая цифра 1?
- Верно ли, что третья цифра 1?
- Верно ли, что четвёртая цифра 1?

Маркированный список

1) Определяем первую цифру.
   a) Верно ли, что первая цифра 0?
   b) Верно ли, что первая цифра 1?
2) Определяем вторую цифру.
   a) Верно ли, что вторая цифра 0?
   b) Верно ли, что вторая цифра 1?

Многоуровневый список

ОСНОВНЫЕ ПАРАМЕТРЫ ШРИФТОВ Основных параметров, которыми характеризуются символы текста, три: гарнитура, кегль и начертание. **Гарнитурой** называется вид шрифта. Наиболее часто используемые гарнитуры — Times и Arial. Они представляют два разных типа шрифтов: шрифты с засечками и рубленые шрифты. Например:

Этот текст набран шрифтом с засечками.

А этот текст набран рубленым шрифтом.

Шрифты с засечками как бы визуально объединяют слово в единое целое, и это увеличивает скорость чтения на 10–15%. Рубленые шрифты часто используются в заголовках и подписях к рисункам.

Кроме названий, определяющих вид букв, шрифты имеют размер, называемый **кеглем**. Стандартный шрифт, к которому стараются приблизиться, печатая деловые документы, очень похож на шрифт гарнитуры Times размером в 12 пунктов. Вообще, хорошо читаются шрифты с кеглем от 9 до 14 пунктов.

Обычно каждый шрифт имеет по крайней мере три модификации начертания:

полужирный, *наклонный* (курсивом),
подчёркнутый.

Могут также использоваться комбинации модификаций, например одновременно полужирный, наклонный и подчёр-

Основные модификации начертания шрифтов: полужирный, наклонный, подчёркнутый.

кнутый — ***вот так***. Подчёркивание можно выполнять различными типами линий, например, **двойной** чертой или пунктиром.

Любую комбинацию символов можно расположить в виде верхнего или нижнего индексов; можно изменить расстояние между символами, например, записать слово

в р а з р я д к у .

Интервал между символами здесь также измеряется в пунктах. Для записи текста в разрядку крайне нежелательно использовать символ «пробел» между буквами, поскольку, во-первых, при расположении такого слова на правом краю строки часть этого слова может быть автоматически перенесена на новую строку, а во-вторых, выполняя выравнивание по ширине, большинство редакторов добивается этого растягиванием пробелов между словами. Поскольку в этом случае каждый символ будет восприниматься как отдельное слово, то и пробелы между ними также будут растягиваться. Все изменения шрифтовых параметров производятся с помощью вкладки Шрифт в меню Формат.

Конечно, мы привыкли к белому фону, на котором располагается текст, как правило, чёрного цвета. Но при желании можно поместить текст на цветной фон или сделать цветными сами символы текста.

Изменяя размер и форму шрифта, можно добиваться самых различных эффектов, но не надо впадать в крайности: большое количество шрифтов на одной странице ухудшает восприятие текста и вряд ли свидетельствует о хорошем вкусе. (Можете это проверить на примере практически любой газеты, основу которой составляют рекламные объявления.)

ТЕКСТЫ НА МОБИЛЬНЫХ ПРИЛОЖЕНИЯХ Современные мобильные устройства — планшеты, смартфоны и даже обычные сотовые телефоны — допускают работу с текстом. Конечно, в обычном сотовом телефоне вам не потребуется форматировать страницы или применять шрифтовые выделения. А вот на планшете или смартфоне вы можете создавать практически полноценные текстовые документы. Многие текстовые редакторы на мобильных устройствах при наборе текста предлагают готовые варианты слов, что ускоряет набор. Но окончательный выбор остаётся, конечно, за вами.

ТЕХНИЧЕСКИЕ СРЕДСТВА ВВОДА ТЕКСТОВ Если текст уже существует на бумаге, то вводить его посимвольно в компьютер, конечно, не хочется. Поэтому были созданы программы, позволяющие избавить человека от этой рутинной работы. Сначала такой текст сканируется, чтобы возник электронный документ, а затем к нему применяется программа распознавания. Надо только иметь в виду, что в силу дефектов в распечатанном тексте не всегда символы распознаются правильно. Поэтому после получения текстового документа его надо отредактировать, применяя средства проверки орфографии, форматирования и т.д.

В последнее время успешно разрабатываются средства распознавания рукописного текста и устной речи.

ВОПРОСЫ:

- Что такое текстовый редактор?
- Что такое форматирование текста?
- Какие типы выравнивания текста применяются? Что означает каждый из них?
- Что такое колонтитул и для чего он используется?
- Какая строка называется красной, а какая — висячей?
- Что такое межстрочный интервал? В каких единицах он измеряется?
- Что такое кегль? В каких единицах он обычно измеряется? Какие значения кегля предпочтительны для обычного текста?

9

ВСТАВКА ОБЪЕКТОВ В ТЕКСТ ДОКУМЕНТА

ВЫ УЗНАЕТЕ:

- Какими возможностями создания и редактирования таблиц располагает текстовый редактор
- Как создать простой рисунок в текстовом редакторе
- Что называют обтеканием рисунка
- Как в текстовый документ вставить объект другого приложения

ВСПОМНИТЕ:

- В чём состоит различие между векторным и растровым способом создания рисунков

Обтекание вставленного объекта (таблицы, рисунка и т.п.) текстом — это способ расположения объекта, при котором строки текста будут ограничены границами вставленного объекта.

а)

Рисуем и вставляем рисунки в текст. Рисунок мож ному располагаться отно текста. Например, так, как показано здесь.

б)

Рисуем и вставляем рисунки в текст. Рисунок может по-разному располагаться относительно текста. Например, так, как показано здесь.

в)

Рисуем и вставляем рисунки в текст. Рисунок может по-разному располагаться относительно текста. Например, так, как показано здесь.

Текстовое сообщение — важная форма сохранения и передачи информации. Но немалую роль играют другие формы представления информации — рисунки, таблицы, схемы, диаграммы и т.д. Документ, в котором, кроме текста, использованы другие информационные объекты, лучше воспринимается и нередко оказывается более продуктивным для информационной деятельности человека. Поэтому практически все современные текстовые редакторы предусматривают возможность внедрения указанных информационных объектов в текстовый документ.

ТАБЛИЦЫ В ТЕКСТОВОМ РЕДАКТОРЕ Составной частью текстового редактора является достаточно простой, можно сказать примитивный, табличный редактор. Он позволяет разграфить часть страницы документа горизонтальными и вертикальными линиями. При вводе текста в клетку получившейся таблицы (такую клетку называют **ячейкой**) можно быть уверенным, что он будет строго ограничен левой и правой вертикальными линиями и не выйдет за их пределы. Можно изменять размеры ячеек и даже их расположение относительно друг друга, например сдвигать в горизонтальном направлении. При необходимости горизонтальные и вертикальные линии можно сделать невидимыми, а некоторые ячейки выделить цветной рамкой или цветным фоном. Всё это позволяет использовать таблицы как весьма эффективное средство редактирования текста.

Текстовый редактор может автоматически преобразовывать данные, заданные в форме текста, в таблично организованные. Для этого надо выделить текст для будущей таблицы и указать символ, играющий роль разделителя текстовой информации, которая будет заноситься в разные ячейки.

Если в ячейках таблицы находятся числа, то редактор может выполнить с ними различные арифметические действия. Например, найти сумму по столбцу или строке, вычислить среднее или максимальное значение и т.п.

Таблицу можно передвигать в тексте как единое целое, в частности, можно организовать обтекание таблицы текстом. Если же таблица была создана в другом приложении, то её можно вставить как объект.

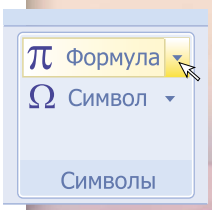
РИСУНКИ В ТЕКСТОВОМ РЕДАКТОРЕ Вставка рисунка тоже может производиться разными способами. Во-первых, имеется встроенный векторный графический редактор, позволяющий рисовать прямо в документе. При этом может оказаться, что нарисованная фигура закроет часть текста (рис. *а*). Если это не входит в ваши планы, легко поменять взаимное расположение фигуры и текста (рис. *б*) или сделать так, что текст будет обтекать рисунок (рис. *в*).

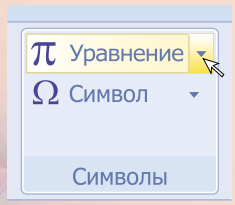
Во-вторых, рисунок может быть создан с помощью какого-либо другого приложения или взят готовым из библиотеки рисунков. В этом случае рисунок обычно вставляется как объект с эффектом обтекания. Тип рисунка — векторный он или растровый — роли не играет.

ВСТАВКА ФОРМУЛ В ТЕКСТОВОМ РЕДАКТОРЕ Многие закономерности, как вы знаете, описываются языком математических формул. Если математическое выражение не очень сложное, например многочлен от одной или нескольких переменных, то для его записи хватает обычных средств работы со шрифтами, имеющимися в редакторе Word. Но как поступить, если в тексте требуется представить формулу, в которой фигурируют дроби, корни, переменные с верхними и нижними индексами одновременно, векторы и другие математические символы? Как, к примеру, создать такие формулы:

$$S = v_0 t + \frac{at^2}{2}, \quad A = \frac{mv_{нач}^2}{2} - \frac{mv_{кон}^2}{2}?$$

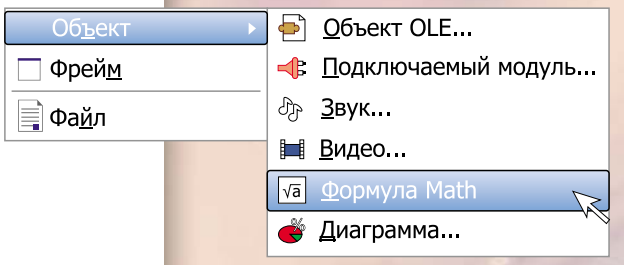
В этом случае приходится пользоваться специальными средствами. Одним из них является редактор формул.





Чтобы активировать редактор формул в Microsoft Office Word, надо на вкладке *Вставка* в группе *Символы* щёлкнуть по пункту *Формула* (см. рисунок).

Чтобы активировать редактор формул в OpenOffice.org Write, надо в меню на вкладке *Вставка* выбрать пункт *Объект*, а затем в нём — пункт *Формула Math* (см. рисунок).



Активировав редактор формул, вы увидите в тексте документа выделенную рамкой область для набора формулы и панель инструментов, с помощью которых формула конструируется. Русские и латинские буквы, знаки основных арифметических действий, индексы набираются в обычном режиме с клавиатуры. Если же вам понадобятся греческие буквы, то их придётся вставлять с помощью панели с символами. Там же вы найдёте символы типа ±, ⩽, → и др. А для того чтобы набрать обыкновенную дробь, придётся воспользоваться специальным меню. То же самое необходимо, если вам одновременно нужно создать верхний или нижний индекс. Другие возможности редактора формул вам пока едва ли понадобятся, но вы можете с ними познакомиться с помощью справки. После того как формула набрана, для выхода из режима создания формулы достаточно щёлкнуть левой клавишей мыши за пределами поля формулы. Вернуться в поле формулы для её редактирования можно, сделав один или два щелчка по нужной формуле.

ВСТАВКА ДРУГИХ ОБЪЕКТОВ В ТЕКСТОВОМ РЕДАКТОРЕ В текстовый документ можно встраивать и другие объекты, в частности те, которые созданы с помощью других приложений. Это могут быть анимированные изображения, звук, видеофрагменты и т.д. Для их встраивания применяется единый механизм OLE — Object Linking Embedding, который позволяет внедрять объекты не только в текст, но и в другие электронные документы. Этот инструмент был создан в 1990 году. Позже он был усовершенствован и развит, после чего компания Microsoft переименовала его в ActiveX.

Текстовый редактор OpenOffice также позволяет вставлять, как говорят, импортировать, объекты, созданные в других приложениях.

ВОПРОСЫ:

- Какие возможности в создании текстового документа предоставляет вставка объектов?
- Какие объекты могут быть созданы и внедрены в текстовый документ собственными средствами редактора Word?
- Какие варианты расположения объекта относительно текста допускает текстовый редактор?
- Каков универсальный механизм внедрения объектов, созданных в других приложениях?

10

ГИПЕРТЕКСТ

ВЫ УЗНАЕТЕ:

- Что такое гипертекст
- Как связать между собой фрагменты текста с помощью гиперссылки
- В чём разница между сносками и ссылками
- Как создать гипертекст
- Чем отличаются гипертекстовые редакторы от редакторов обычного текста
- Что такое браузер и как он работает

ВСПОМНИТЕ:

- Основные структурные элементы текста
- Как в текст внедряются объекты

При всём том, что было рассказано о тексте, одно его свойство остаётся неизменным — линейный характер. Это означает, что для перехода от одного фрагмента текста к другому надо пролистать весь промежуточный текст. Ещё сложнее такой поиск осуществить, если связанные по смыслу фрагменты текста находятся в разных документах. Чтобы автоматизировать переход с одного фрагмента на другой, их надо связать между собой. Осуществлять такую связь призваны так называемые **гиперссылки**.

СНОСКИ И ССЫЛКИ Самые простые виды связей между фрагментами текста, известными ещё с докомпьютерной эры, — это сноски и ссылки. Сноски бывают постраничные (в этом случае они размещаются на нижнем поле страницы) и концевые (тогда они размещаются в конце текста). Текстовый редактор автоматически нумерует сноски и ссылки. Конечно, теперь перейти на сноску или ссылку можно одним щелчком клавиши мыши.

ГИПЕРТЕКСТ Текст, в котором возможен автоматизированный переход от одного фрагмента к другому, называют гипертекстом. Можно представлять себе, что гипертекстовая страница состоит из видимого и «подвального» этажей. На видимом располагаются самый обычный текст и самые обычные картинки. Но только отдельные слова и картинки могут быть особо выделены, и курсор мыши, оказавшись на них, может принимать другую форму. Это означает, что под этими элементами страницы находятся ссылки на другие электронные документы или даже на целые программы, которые будут выполнены в случае нажатия на кнопку мыши при видоизменённом курсоре. Что же располагается в «подвале», какие бывают ссылки?

- Самое простое — переход к другим страницам или к другому документу.
- Тоже не очень сложно — показ картинки или фильма, прослушивание звукового фрагмента.
- Немного сложнее (с точки зрения аппаратуры, но не с точки зрения того, кто просматривает гипертекстовый документ) — перенаправление пользователя к другому гипертекстовому документу на другом сервере, расположенном, быть может, за тысячи километров.
- Активизация специальных программ (чаще всего это программы пересылки файлов).

Простейший гипертекстовый документ можно создать с помощью редактора Word, используя, например, объект *Закладка*. Чтобы создать закладку, нужно выделить в тексте фрагмент, затем на закладке *Вставка* выбрать пункт *Закладка* и в открывшемся окне ввести имя закладки. Затем выделить фрагмент текста и выбрать теперь пункт *Гиперссылка*. На диалоговой панели указать, с чем вы намерены связать эту закладку (с местом в текущем документе или в каком-то другом документе), и выбрать имя закладки.

Термин «гипертекст» был предложен в 1963 году Тедом Нельсоном, а в 1968 году была продемонстрирована первая работающая гипертекстовая система. Именно в ходе разработки этой системы была придумана компьютерная мышь, многооконный интерфейс, контекстнозависимые меню.

Гипертекстом называется текст, в котором организованы гиперссылки на фрагменты или объекты того же текста или другого документа.

ГИПЕРТЕКСТОВЫЕ РЕДАКТОРЫ Обычный текстовый редактор вовсе не предназначается для создания гипертекстовых страниц. Это инструмент для создания прежде всего обычных текстовых документов. А для создания гипертекстовых страниц, в которых использовались бы все возможности, перечисленные нами выше, нужен специальный гипертекстовый редактор. Таких редакторов существует довольно много, и у каждого свой стандарт форматов страниц. Понятно, что было необходимо выбрать один из них для обеспечения всем желающим беспрепятственного использования постоянно растущего множества разнообразных гипертекстовых объектов и особенно в глобальных компьютерных сетях. Таким стандартом стал HTML-стандарт. Сама аббревиатура HTML происходит от **H**yper-**T**ext **M**arkup **L**anguage (в переводе — язык разметки гипертекста).

Любой текстовый редактор, формируя текст абзац за абзацем, страница за страницей, одновременно создаёт описание форматирования текста и хранит его в файле вместе с самим текстом. Когда на экран вызывается текст документа, то программа-редактор, следуя этому описанию, воссоздаёт вид документа. Иногда это описание занимает не меньше памяти, чем сам текст. Редактор Word в этом отношении никак нельзя назвать экономным — вы, возможно, замечали, что добавление к уже имеющемуся тексту ещё нескольких строк (особенно если их вставлять в середину) может увеличить размеры файла на 5–10 килобайт. Это плата за то, что пользователь освобождён от забот по созданию такого описания. Он даже не видит этой работы, так же как богач-аристократ не видит работы своего садовника, а только наслаждается лужайками и цветниками в своём саду.

Но существует и иной тип редакторов. В них описание форматирования делает сам автор текста. Получается обычный текстовый файл, в котором фактически присутствует только один управляющий символ форматирования — конец строки. Такой текст имеет, как правило, информационный объём, всего лишь на несколько процентов превышающий информационный объём собственно текста. Затем такой файл обрабатывается специальной программой, которая, следуя указаниям автора, создаёт документ с требуемым форматированием текста. Программа просмотра файла в формате HTML называется **браузером** (от английского *browser* — просмотрщик). Браузер, получив HTML-файл, формирует соответствующий гипертекст. Из современных браузеров наиболее популярными являются Google Chrome, Safari, Opera, Mozilla Firefox, Microsoft Internet Explorer.

Браузер — программа просмотра электронных гипертекстовых страниц, хранящихся как на одном компьютере, так и на серверах глобальных компьютерных сетей.

ВОПРОСЫ:

- В чём разница между обычной текстовой страницей и гипертекстовой страницей?
- Что такое гипертекст?
- Как выделяются гипертекстовые ссылки среди других элементов гипертекстовой страницы?
- Какой стандартный редактор используется для создания гипертекстовых страниц?
- В чём основные различия текста в обычном текстовом формате и текста в формате HTML?
- Как называется программа, предназначенная для просмотра файлов в HTML-формате?

11 ОРГАНИЗАЦИЯ ВЫЧИСЛЕНИЙ ПРИ ПОМОЩИ КОМПЬЮТЕРА

ВЫ УЗНАЕТЕ:

- Как производятся вычисления в электронной таблице
- Как записываются формулы в электронной таблице
- Как устанавливаются связи между ячейками электронной таблицы

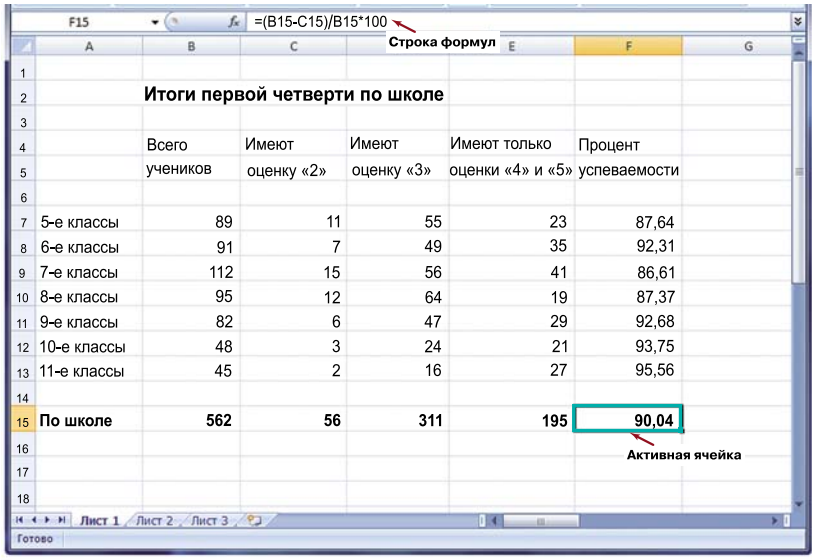
ВСПОМНИТЕ:

- Основные виды информации

Электронная таблица – программа, с помощью которой можно производить вычисления с данными, представленными в табличной форме.

Уже в самом слове «компьютер», что с английского переводится как вычислитель, кроется первоначальное предназначение этого устройства. Не случайно даже в текстовых редакторах предусмотрена возможность выполнения вычислений. Но вычислительные возможности компьютера, конечно, намного шире, чем те, которые представлены в текстовых редакторах. Одним из мощных средств организации вычислений при помощи компьютера являются электронные таблицы.

ПРЕДСТАВЛЕНИЕ ИНФОРМАЦИИ В ЭЛЕКТРОННОЙ ТАБЛИЦЕ Из названия ясно, что информация в электронной таблице представлена в табличном виде. Любая таблица образована как совокупность строк и столбцов. Пересечение строки и столбца образует клетку, которую обычно называют **ячейкой**. Столбцы электронной таблицы обозначены буквами латинского алфавита, а строки перенумерованы. Каждая ячейка тем самым получает **адрес**, состоящий из обозначения столбца и номера строки. На рисунке показано, как выглядит электронная таблица на экране компьютера. В ней представлена информация об успеваемости учащихся с 5 по 11 класс некоторой школы.



Итоги первой четверти по школе

	Всего учеников	Имеют оценку «2»	Имеют оценку «3»	Имеют только оценки «4» и «5»	Процент успеваемости
5-е классы	89	11	55	23	87,64
6-е классы	91	7	49	35	92,31
7-е классы	112	15	56	41	86,61
8-е классы	95	12	64	19	87,37
9-е классы	82	6	47	29	92,68
10-е классы	48	3	24	21	93,75
11-е классы	45	2	16	27	95,56
По школе	**562**	**56**	**311**	**195**	**90,04**

Ширину столбцов и высоту строк можно менять, выбирая для различных данных наиболее подходящие, а некоторые строки пропускать. Это позволяет красиво и удобно оформить документ.

Рассматривая заполнение электронной таблицы, представленной на рисунке, легко заметить, что информация в клетках таблицы имеет разное происхождение. Например, информация о том, сколько человек имеет тройки, должна быть занесена в таблицу пользователем, а вот процент успеваемости может вычислить компьютер. Поэтому информа-

цию, содержащуюся в электронной таблице, делят на **исходные данные**, т.е. вводимые пользователем, и **рассчитываемые данные**, или, по-другому, **результаты**. Результаты с исходными данными связаны формулами. Формула записывается в ту ячейку, где должен отображаться результат. В самих ячейках формулы, как правило, не видны, но их можно увидеть в отдельной строке (*Строке формул*), если ячейка активна (чтобы сделать ячейку активной, по ней нужно щёлкнуть мышью). На рисунке выделена ячейка с адресом F15. И главное, при изменении исходных данных все результаты автоматически пересчитываются и изменяются прямо у вас на глазах.

Для каждой таблицы существуют специальные правила заполнения, указанные в инструкции пользователю.

Каждая ячейка электронной таблицы имеет определённый формат, который определяет, какого типа информация содержится в ячейке — текст, число, дата и т.д.

Важно помнить, что если формат ячейки задан как число, то среди вводимых в ячейку символов могут быть только цифры, знак «минус» и один символ, делящий целую часть от дробной, — точка или запятая. Например, последовательности символов:

Василий, 10.234.245, a1234 —

это всё текст.

На закладке *Главная* в области *Число* вы можете найти целый список различных форматов: числовой, денежный, время, текст и т.д. Среди них есть формат *Общий*, который обычно стоит по умолчанию во всех ячейках. Если для ячейки задан формат *Общий*, это означает, что сам компьютер будет определять, какого рода информация заносится в ячейку.

В диалоговом окне *Формат ячейки* вы сможете не только выбрать формат, но и указать, в каком виде вы хотите видеть информацию, к примеру, как будет выглядеть дата или сколько нужно показывать цифр после запятой в десятичной дроби.

Чтобы электронная таблица распознавала, что содержится в ячейке — просто информация или формула, — в начале формулы ставится специальный знак. Для Microsoft Excel и OpenOffice.org Calc это знак =.

В формулах могут использоваться знаки всех известных вам арифметических операций. Надо только помнить, что умножение обозначается символом * и он обязательно должен стоять между сомножителями; деление обозначается символом /. Кроме того, все формулы обязательно пишутся в одну строку. Вы не можете в ячейке электронной таблицы написать, например:

$$\frac{(3+5)(7-4)}{5+6:8}.$$

Такое выражение нужно преобразовать в запись

= (3 + 5)*(7 — 4)/(5 + 6/8).

Как правило, электронная таблица имеет, помимо всего прочего, целый набор стандартных функций, облегчающих жизнь пользователю. Как вы думаете, какая формула стоит в ячейке B15 таблицы, которая изображена на с. 40?

Идея электронной таблицы кажется простой, однако додумались до неё спустя целых двадцать лет после того, как стали применять компьютеры бухгалтерии. Сделал это Дэниэл Бриклин в 1979 г. Вместе с программистом Робертом Фрэнкстоном он и создал первую в мире электронную таблицу с названием «Визикалк», что по замыслу разработчиков означало «Визуальный калькулятор». Даже одна эта программа оправдывала в глазах пользователей приобретение персонального компьютера: более чем 100 000 человек купили персональный компьютер ради возможности работать с «Визикалком».

Блоком ячеек в электронной таблице называется совокупность всех ячеек, заполняющих сплошным образом некоторый прямоугольник.

Принцип **относительной адресации** означает, что адреса ячеек в формуле определены не абсолютно, а относительно той ячейки, где стоит формула.
Абсолютная адресация применяется для ячеек, адреса которых не меняются в ходе вычислений.

Формулы в ячейках электронной таблицы записываются в линейной форме.

Возможно, такая:

=B7 + B8 + B9 + B10 + B11 + B12 + B13

А если надо сложить не семь, а двадцать семь или сто семь чисел? Для этого имеется стандартная функция — суммирование содержимого блока ячеек:

СУММ(B7:B13), если используется Microsoft Excel,

или

SUM(B7:B13), если используется OpenOffice.org Calc.

Для того чтобы электронная таблица знала, с каким блоком ей иметь дело, указывают через двоеточие адреса ячеек, стоящих в левом верхнем и правом нижнем углах прямоугольника.

У электронной таблицы есть много разных операций над блоками ячеек. Это поиск максимального или минимального элемента (операции МАКС и МИН в Microsoft Excel и MAX и MIN в OpenOffice.org Calc соответственно), расчёт среднего арифметического значения (операция СРЗНАЧ для Microsoft Excel и AVERAGE для OpenOffice.org Calc) и т.д.

Блок ячеек можно скопировать или перенести в другое место таблицы. При этом в электронной таблице действует так называемый принцип **относительной адресации**. Он означает, что адреса ячеек в формуле определены не абсолютно, а относительно той ячейки, где стоит формула. Поэтому при копировании содержимого ячейки или блока ячеек в другое место автоматически пересчитываются адреса ячеек, фигурирующие в формулах копируемого фрагмента. Таким образом, относительная адресация проявляет себя в том, что всякое изменение места расположения формулы приводит к автоматическому пересчёту адресов ячеек, фигурирующих в этой формуле. Например, если формулу

=СУММ(B6:B12)

из ячейки B14 скопировать в ячейку C15, то в ней окажется формула

=СУММ(C7:C13).

Если же в копируемых формулах всё время должна использоваться одна и та же ячейка (например, в ней хранится нужная константа), то адрес такой ячейки помечают в формуле как **абсолютный**. Обычно для этого применяется какой-либо зарезервированный символ (довольно часто это знак $). Если, к примеру, в ячейке B14 записана формула

=СУММ($B$6:B12),

то после её копирования в ячейку C15 формула примет вид

=СУММ($B$6:C13).

Среди операций, которые можно выполнять над электронной таблицей, всегда присутствуют операции вставки и удаления строк и столбцов, копирования блока ячеек, очистки содержимого блока ячеек и т.п.

ПРИМЕНЕНИЕ ЭЛЕКТРОННОЙ ТАБЛИЦЫ ДЛЯ РЕШЕНИЯ ЗАДАЧ

Использование электронной таблицы требует тщательного предварительного продумывания: каковы исходные данные задачи и в каких ячейках вы будете их размещать, каковы формулы, связывающие исходные данные с результатами,

куда их поместить и т.д. Чтобы вам было проще этому научиться, обсудим следующую задачу.

На некоторых соревнованиях, где результат не измеряется в метрах, секундах, забитых и пропущенных мячах, а определяется техничностью и красотой исполнения, принята следующая форма судейства. Каждый судья выставляет свою оценку спортсмену, затем самая высокая и самая низкая оценки отбрасываются, а для остальных оценок вычисляется среднее арифметическое. Это число и принимается за результат, показанный спортсменом.

Нужно с помощью электронной таблицы организовать вычисление результатов для каждого спортсмена, а затем и получение итогового протокола. Для конкретности пусть в соревновании участвует 8 спортсменов, а судейская бригада состоит из 5 судей.

Подготовим электронную таблицу. Фамилии спортсменов расположим в столбце B в том порядке, как они идут в протоколе старта, а столбец A оставим для порядкового номера. Каждому судье в этой таблице естественно предоставить столбец. Мы не будем уточнять их фамилии, а просто присвоим номера от 1 до 5.

	A	B	C	D	E	F	G	H
1	№	Фамилия	1	2	3	4	5	Результат
2	1	Сотов						
3	2	Петров						
4	3	Минин						
5	4	Козлов						
6	5	Семёнов						
7	6	Белов						
8	7	Жилин						
9	8	Котов						

По мере выступления того или иного спортсмена ячейки соответствующей строки будут заполняться оценками судей. Хочется поручить компьютеру в столбце H вычислять результат для данного спортсмена. Это означает, что нам придётся выполнять операции над блоком ячеек, расположенных в данном случае в одной строке. Но у разных спортсменов максимальная (как и минимальная) оценка может быть получена у совершенно разных судей, т.е. располагаться в разных столбцах. Значит, в ячейку H2 надо записать формулу

=(СУММ(C2:G2)–МАКС(C2:G2)–МИН(C2:G2))/3

Обратите внимание: здесь удобно использовать именно относительную адресацию ячеек в блоке C2:G2, тогда при копировании этой формулы в другие ячейки столбца H в них будут получаться нужные нам формулы.

Более обстоятельно вы будете знакомиться с возможностями электронной таблицы, выполняя задания компьютерного практикума.

ВОПРОСЫ:

- Для чего предназначены электронные таблицы?
- Какая информация, содержащаяся в электронной таблице, называется исходными данными, а какая — результатами?
- В ячейке электронной таблицы записано выражение:

  =18 – 12*5/3*2.

  Какое значение будет вычислено в этой ячейке? Ответьте на тот же вопрос для выражения

  =(17 – 8)*8/(3*4).
- Что такое адрес ячейки электронной таблицы?
- В чём состоит принцип относительной адресации?
- Какой адрес называется абсолютным?

12

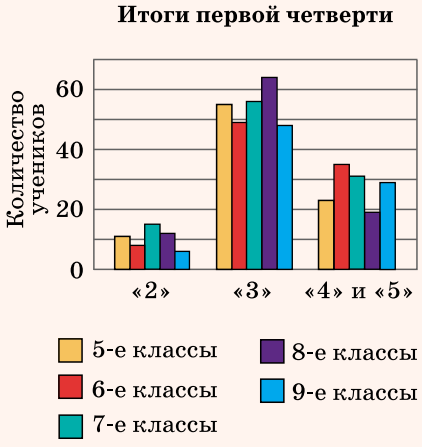
ПОСТРОЕНИЕ ГРАФИКОВ И ДИАГРАММ С ПОМОЩЬЮ ЭЛЕКТРОННОЙ ТАБЛИЦЫ

ВЫ УЗНАЕТЕ:

- Как построить диаграмму с помощью электронной таблицы
- Как с помощью электронной таблицы построить график зависимости между величинами
- Какие виды диаграмм и графиков можно строить с помощью электронной таблицы
- Как построить изображение поверхности

ВСПОМНИТЕ:

- Что называют диаграммой
- Какие виды диаграмм обычно используются
- Что такое график функции



Человек, в отличие от компьютера, легче воспринимает и анализирует информацию, представленную не в числовой, а в визуальной форме — графиков, диаграмм и др. Электронные таблицы обладают инструментами визуализации числовых данных.

ПРЕОБРАЗОВАНИЕ ТАБЛИЧНЫХ ДАННЫХ В ДИАГРАММУ До появления компьютера преобразование числовых данных, даже если они были представлены в табличной форме, осуществлялось вручную. Есть много разных видов диаграмм, но особенно часто используют **столбчатые** и **круговые диаграммы**. Примеры таких диаграмм показаны на рисунке. Столбчатые диаграммы удобны, когда надо сравнить, например, различие одних и тех же показателей в разных условиях. Круговые диаграммы показывают соотнесённость различных частей целого. Точно так же соотнесённость различных частей целого хорошо демонстрируют и диаграммы с областями.

Для построения диаграммы в электронной таблице есть специальный инструмент.

В Microsoft Excel построить диаграмму можно, если предварительно поставить курсор в одну из ячеек, где расположены данные для построения диаграммы, а потом нажать на одну из кнопок на вкладке *Вставка* в области *Диаграммы*.

В OpenOffice.org Calc построить диаграмму можно, если предварительно выделить блок ячеек, в которых расположены данные для построения диаграммы, а затем в меню *Вставка* выбрать пункт *Диаграмма*. После этого появится мастер диаграмм, который поможет вам выбрать тип диаграммы и её внешнее оформление.

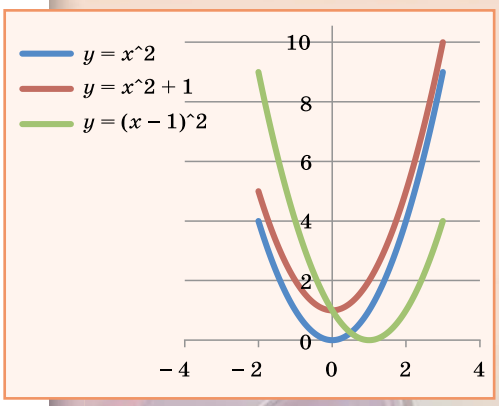
На первом шаге нужно выбрать тип диаграммы. Если вы решили строить столбчатую диаграмму, то надо выбрать режим *Гистограмма*. Но и в этом режиме имеется несколько вариантов изображения диаграммы. Второй шаг — это выбор нужного вида диаграммы. Теперь вы должны указать, из каких ячеек таблицы будет взята информация для построения диаграммы. Если вы выделяете блок ячеек, стоящих в одной строке, то на диаграмме вы получите одну последовательность столбиков. Если же вы выделите блок ячеек сразу в нескольких строках, то на диаграмме будет столько рядов из столбиков, из скольких строк состоит выделенный блок ячеек. Впрочем, можно указать компьютеру, чтобы он брал данные не из строки, а из столбца таблицы.

ПОСТРОЕНИЕ ГРАФИКОВ С ПОМОЩЬЮ ЭЛЕКТРОННОЙ ТАБЛИЦЫ

Изобретение Р. Декартом системы координат позволило графически изображать зависимости между величинами. Для этого на одной из осей откладываются значения аргумента, а на другой — значение функции. Если это сделать для достаточно большого числа значений аргумента, а получившиеся на координатной плоскости точки соединить линией, то получится график функции. Поэтому в электронной таблице такой тип диаграмм получил название *точечная*.

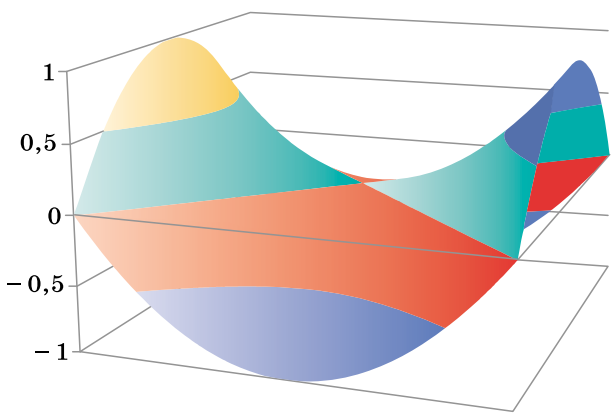
Для построения графика значения аргумента записываются в столбец электронной таблицы (впрочем, их можно записать и в строку). Поскольку изменение значений аргумента обычно делают с некоторым постоянным шагом, для облегчения этой работы можно записать только два первых значения, а дальше воспользоваться режимом автозаполнения. В следующем столбце записывается формула для вычисления значения функции, после чего она копируется в остальные ячейки столбца. Выделив эти два столбца, нужно перейти в режим построения диаграмм и выбрать тип *Точечная*. График функции будет построен. После этого вы можете приступить к редактированию полученного изображения — выбрать подписи к осям, названию графика и т.п.

Можно построить графики сразу нескольких функций (см. рисунок). Для этого надо заполнить формулами для вычисления значений функций столько столбцов, сколько графиков вы собираетесь изобразить.



ИЗОБРАЖЕНИЕ ПОВЕРХНОСТЕЙ С ПОМОЩЬЮ ЭЛЕКТРОННОЙ ТАБЛИЦЫ

Поверхность задаётся как функция двух аргументов. Поэтому для её построения берётся блок ячеек. В первый столбец этого блока заносятся значения первого аргумента, в первую строку — значения второго аргумента. А затем для соответствующих значений аргументов оставшиеся ячейки блока заполняются формулами для вычисления значений функции (или самими значениями функции), после чего надо выделить блок полученных данных и в режиме построения диаграмм выбрать тип *Поверхность*. На рисунке изображена поверхность для функции $z = y^2 - x^2$.



Слово *диаграмма* происходит от греческого διάγραμμα (δια — через, γραμμα — изображение). Оно означает графическое представление числовых данных.

ВОПРОСЫ:

- Какие формы графического представления информации обычно используются?
- Укажите, какой вид диаграмм — столбчатая или круговая — вам кажется более удобным для наглядного представления информации:

а) об использовании земли на некоторой территории: посевные площади, пастбища, хозяйственные постройки, лесные угодья, отчуждаемые земли (например, под шоссейные или железные дороги);

б) об изменении численности населения, проживающего в некотором регионе, за несколько последних лет;

в) о потреблении воды промышленным предприятием в течение года;

г) о содержании солей в питьевой минеральной воде;

д) об изменении цен на данный вид товаров.

13

ПРИМЕНЕНИЕ ЭЛЕКТРОННОЙ ТАБЛИЦЫ В ЗАДАЧАХ ПРОЕКТИРОВАНИЯ

ВЫ УЗНАЕТЕ:

- Что такое задачи оптимизации
- Как использовать надстройку *Поиск решения* для решения задач оптимизации

ВСПОМНИТЕ:

- Как построить график функции в электронной таблице

В жизненных задачах нередко ставится цель получить лучшее в том или ином смысле решение. Такое решение называют оптимальным, а сами такие задачи называются задачами оптимизации.

Важно не только знать возможности электронных таблиц, но и уметь применять эти возможности для решения задач. Многие задачи, реально возникающие в человеческой практике, требуют не просто нахождения какого-нибудь ответа, а указания наилучшего в том или ином смысле решения. Такие задачи получили название задач оптимизации. Особенно часто задачи оптимизации возникают при проектировании или конструировании каких-либо объектов — ведь всегда хочется найти оптимальное решение.

ЗАДАЧА ПРОЕКТИРОВАНИЯ ЦИЛИНДРИЧЕСКОЙ ЁМКОСТИ Требуется изготовить ёмкость цилиндрической формы без крышки объёмом 10 л. Каким должен быть радиус основания и какой высота ёмкости, чтобы на её изготовление потребовалось как можно меньше материала?

Договоримся прежде всего об обозначениях. Высоту цилиндра обозначим буквой h, а радиус основания — буквой r. Измерять высоту и радиус будем в сантиметрах. Тогда объём цилиндра запишется формулой

$$V = \pi r^2 h,$$

а мы знаем, что по условию он равен 10 000 см^3.

Количество материала, необходимого для изготовления данной ёмкости, определяется площадью её поверхности. Обозначив площадь поверхности через S, нетрудно понять, что

$$S = \pi r^2 + 2\pi r h.$$

Однако

$$h = \frac{1000}{\pi r^2},$$

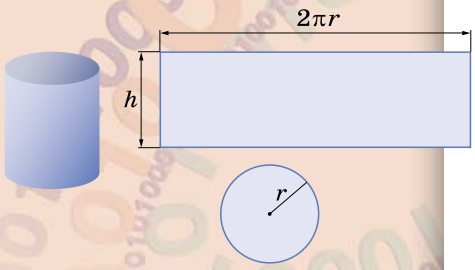
откуда

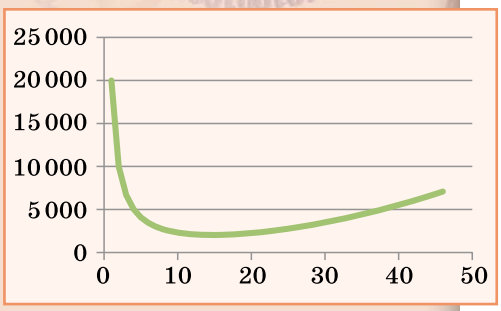
$$S = \pi r^2 + \frac{20000}{r}.$$

Значит, нам нужно найти, при каком положительном значении r величина S будет наименьшей.

Начнём с какого-либо разумного значения r, например с $r = 1$. Поручим электронной таблице вычислить значения S, когда r последовательно принимает различные значения от 1 с шагом 1 (т.е. через 1 см). По вычисленным данным построим график, на котором видно, что наименьшее значение получается, когда $r \approx 15$. Теперь нужно уточнить значение r. Для этого можно в столбце A начать со значения 14 с шагом 0,1. Это позволит получить требуемое значение r уже с точностью 0,1 см. Если нужно ещё более точное значение, то можно ещё раз поступить аналогично.

Но нельзя ли сразу найти то значение r, для которого значение S было бы минимальным? Можно. Для этого в электронной таблице имеется специальный инструмент: *Поиск решения* (Solver). Правда, эта программа является дополнительной, поэтому при установке электронной табли-

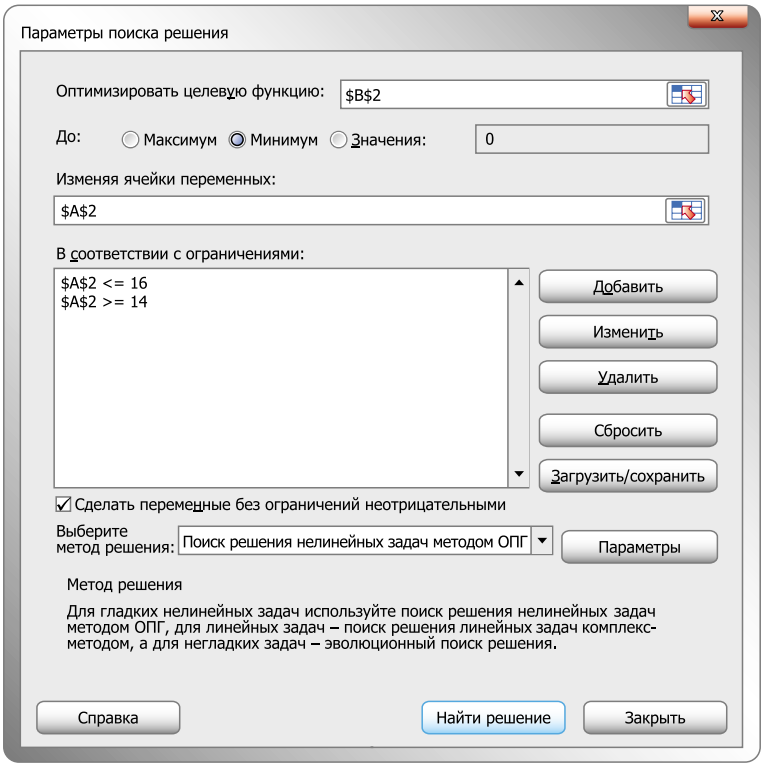




цы она обычно не подключена (если это Microsoft Excel) или подключена с ограниченными возможностями (если это OpenOffice.org Calc).

Теперь давайте поручим компьютеру найти r, для которого значение S будет минимальным. Для этого достаточно заполнить только одну строку таблицы, записав, например, в ячейку A2 какое-либо значение r, а в ячейку B2 — формулу площади S.

Установите курсор в ячейку B2. Затем вызовите подпрограмму *Поиск решения*. На экране появится диалоговое окно (см. рисунок), причём адрес целевой ячейки уже автоматически будет указан, поскольку вы в ней установили курсор.



Изменять мы намерены r, начальное значение которого поместили в ячейку A2. Значит, в окне строки *Изменяя ячейки* надо указать A2 (если вам придётся работать с функцией от нескольких переменных, то здесь указывается блок ячеек, в которых помещены значения этих переменных).

Мы уже знаем, что значение r надо искать на отрезке от 14 до 16. Поэтому их естественно добавить в ограничения. Наконец, надо указать, что требуется найти — максимум или минимум.

Нажав на кнопку *Найти решение*, запускаем программу. Через мгновение появится окно *Результаты поиска решения*, а в ячейке A2 — искомое значение радиуса r. В свою очередь, в ячейке B2 вы увидите значение площади поверхности.

ВОПРОСЫ:

- Какие задачи называют задачами оптимизации?
- Какими возможностями обладает электронная таблица для решения задач оптимизации?

14

ПРИМЕНЕНИЕ ЭЛЕКТРОННОЙ ТАБЛИЦЫ В ЭКОНОМИЧЕСКИХ ЗАДАЧАХ

ВЫ УЗНАЕТЕ:

- Как использовать электронные таблицы для поиска решений в экономических задачах
- Как по экспериментальным данным подбирать описывающую их функцию

ВСПОМНИТЕ:

- Как построить график функции в электронной таблице
- Как использовать надстройку *Поиск решения*

Владелец небольшого кинотеатра хочет знать, какую цену надо установить на билеты, чтобы выручка была максимальной.
Дирекция авиакомпании хочет установить цену на билеты для пассажиров так, чтобы доход компании был максимальным.
Автопредприятие предоставляет в аренду автобусы. Его руководство хочет установить плату за 1 час аренды так, чтобы доход был максимальным.
Во всех трёх случаях ответ на первый взгляд представляется очевидным: чем выше цена (или арендная плата), тем и доход выше. Однако если назначить слишком высокую цену, то кинозал будет пустым, самолёты будут совершать порожние рейсы, а автобусы простаивать на автобазе. Вместо доходов одни убытки. Какой должна быть оптимальная цена?

ЭЛЕКТРОННЫЕ ТАБЛИЦЫ В ЗАДАЧЕ ЭКОНОМИЧЕСКОГО ПРОГНОЗИРОВАНИЯ Из сказанного выше следует, что количество приобретённых билетов или заказов на автобусы зависит от установленной цены. Пусть цена обозначена буквой x, тогда количество потребителей — это некоторая функция $f(x)$. Конечно, цена зависит от качества фильма или популярности авиамаршрута, но мы сейчас интересуемся другим — как должна формироваться цена?

Экономистов волнует ответ на этот вопрос, а не конкретная величина цены.

Если x — количество потребителей услуги, а $f(x)$ — её цена, то доход (обозначим его буквой D), очевидно, будет равен $D = xf(x)$. Понятно, что изменение величины x ограничено: для кинотеатра — числом мест в зале, для авиакомпании — числом мест в самолёте, в автопарке — количеством автобусов. Нам требуется найти, каково то значение x, при котором значение функции $D = xf(x)$ будет максимальным. Затем, вычислив $f(x)$, можно будет найти и оптимальную цену.

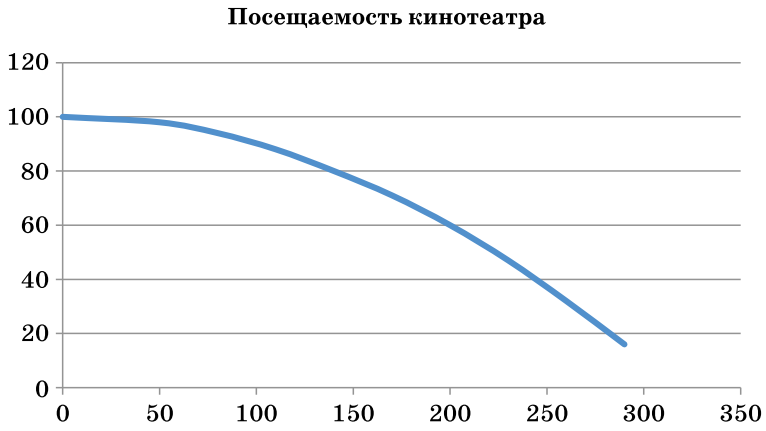
Такая задача была бы совсем несложной, если бы нам была известна формула для функции $f(x)$. Но такую формулу нам никто не предоставит. Однако любой опытный предприниматель, прежде чем предложить свой товар, обязательно изучит спрос на него и, в частности, то, за какую цену потребитель готов покупать его товар (такое исследование называется маркетинговым). Мы взяли у владельца некоторого маленького кинотеатра — всего на 100 мест — отчёт о подобном исследовании (см. таблицу).

Цена, р.	50	80	110	140	170	200	230	260	290
Количество посетителей, чел.	98	94	88	80	71	60	47	32	16

Выше 300 рублей владелец цену поднимать не стал — понял, что никто вообще к нему не придёт. А от себя мы

добавили точку $x = 0$ и $f(0) = 100$: если денег не брать совсем, то зал будет заполнен до отказа.

Воспользовавшись электронной таблицей, мы построили график предполагаемой функции $f(x)$. Он представлен на рисунке.



Такой вид линии вполне объясним: пока цена невелика, количество зрителей убывает незначительно, но потом каждые 10 рублей уже ощутимы для потребителя, и количество посетителей кинотеатра резко убывает. Что касается формы кривой, то получившаяся линия очень похожа на параболу. При этом её вершина, очевидно, приходится на значение $x = 0$, ветви направлены вниз, так что коэффициент при x^2 отрицателен. Следовательно, функция $f(x)$ имеет вид $c - ax^2$, где оба коэффициента a и c положительны. Сами коэффициенты придётся, конечно, подбирать так, чтобы график функции как можно ближе подходил к экспериментальным значениям. Впрочем, коэффициент c равен значению функции $f(x)$ при $x = 0$, так что его значение очевидно.

Чтобы найти неизвестный параметр в формуле, задающей функцию, которая предположительно описывает экспериментальные данные, математики предложили следующий метод. Пусть функция задана некоторой формулой $f(x)$, в которой присутствует параметр a, и таблица значений (x_i, y_i) функции. Было бы идеально, если бы $y_i = f(x_i)$ при каком-то значении параметра a. Но идеал, как правило, недостижим, поэтому постараемся найти такое значение a, чтобы сумма

$$(y_1 - f(x_1))^2 + (y_2 - f(x_2))^2 + \ldots + (y_n - f(x_n))^2$$

была минимальной. А искать минимум функции — это задача, решать которую мы уже учились. Сам же предложенный метод называется **методом наименьших квадратов**. Название, как нам кажется, говорит само за себя.

Остаётся найти неотрицательное значение переменной x, при котором функция $D = x(100 - ax^2)$ имеет наибольшее значение. Ответ вы получите, выполнив лабораторную работу. Он, скорее всего, покажется вам удивительным.

Метод наименьших квадратов — один из наиболее часто применяемых методов, когда по экспериментальным данным требуется подобрать формулу, достаточно точно описывающую эти данные.

Первым метод наименьших квадратов использовал великий немецкий математик К. Гаусс, а название ему спустя 10 лет дал французский математик А. Лежандр.

Вместо суммы квадратов разностей значений можно использовать сумму абсолютных величин. На сколько при этом будут различаться значения параметра a?

ВОПРОСЫ:

- Можно ли экономическую задачу определения цены отнести к задачам оптимизации?
- В чём заключается метод наименьших квадратов?

15

СОЗДАНИЕ И КОМПЬЮТЕРНАЯ ОБРАБОТКА ВИЗУАЛЬНЫХ ИНФОРМАЦИОННЫХ ОБЪЕКТОВ

ВЫ УЗНАЕТЕ:

- Что является наименьшим элементом изображения на экране
- Чем отличаются друг от друга растровая и векторная графика
- Каковы возможности графического редактора

ВСПОМНИТЕ:

- Как создаётся на экране цветное изображение
- Чем характеризуется цвет при создании изображения на экране

Пиксель — наименьший элемент изображения, создаваемого на экране компьютера.
Растр — прямоугольная сетка пикселей на экране.
Разрешение экрана — количество пикселей в строке и количество строк, состоящих из пикселей, на экране.

О создании таких визуальных информационных объектов, как диаграммы, графики, поверхности, мы рассказали. Ещё раньше мы уже коснулись вопроса, как создаётся и кодируется на экране изображение. Но этим вовсе не исчерпывается тема визуализированной информации, создаваемой и обрабатываемой компьютерными технологиями.

ОСНОВНЫЕ ВИДЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РАБОТЫ С КОМПЬЮТЕРНОЙ ГРАФИКОЙ Сегодня имеется немало разнообразных программ для работы с графическим изображением. Они делятся на несколько групп.

Графические редакторы. Незаменимы, когда требуется нарисовать или подправить несложный рисунок.

Программы корректировки и преобразования фотоизображений. С их помощью можно добавить изображению яркость или контрастность, отретушировать его, создать те или иные эффекты (например, добиться того, что изображение кажется нарисованным на шаре или отчеканеным на металле и т.д.).

Программы создания слайд-фильмов и мультфильмов, в том числе программы анимации, т.е. создающие иллюзию движения рисованных объектов.

Программы презентационной графики. Из названия ясно, где эти программы используются. Графическое изображение и звуковое сопровождение, объединённые с их помощью, просто незаменимы для наглядной иллюстрации любого выступления.

Конечно, возможны самые различные варианты объединения названных выше программ и друг с другом, и с текстовыми редакторами, и с электронными таблицами, и с базами данных, о которых речь пойдёт чуть позже.

СПОСОБЫ СОЗДАНИЯ ИЗОБРАЖЕНИЯ НА ЭКРАНЕ Изображение на экране может создаваться **растровым** или **векторным** способом. Если рассмотреть экран компьютера через лупу, то можно увидеть, что изображение состоит из прямоугольников или каких-либо иных мелких отдельных элементов. Наименьший элемент изображения называется **пикселем. Растр** — это прямоугольная сетка пикселей на экране. При растровом способе представления графической информации отдельно задаётся каждый пиксель растра — его цвет, яркость, место расположения на экране. Растровые изображения получаются, например, при сканировании изображений, а также при использовании цифровых фотоаппаратов и видеокамер.

При векторной обработке графической информации изображение создаётся посредством отрезков прямых, дуг эллипсов и окружностей, других линий, рисование которых запрограммировано в соответствующем графическом редакторе.

Векторная графика используется в системах проектирования для создания чертежей.

Именно с векторной графикой вы имеете дело, когда создаёте рисунки в текстовом редакторе. Место, где будет возникать очередной элемент рисунка, указывает графический курсор. Он может выглядеть по-разному: в виде стрелки, крестика или точки. Курсор можно перемещать по экрану с помощью мыши (иногда клавиш со стрелками). При этом в специально отведённом месте экрана можно увидеть два изменяющихся числа — координаты курсора. Они показывают, на каком пикселе находится курсор.

ИНСТРУМЕНТЫ ГРАФИЧЕСКОГО РЕДАКТОРА Знакомство с программами обработки графической информации начнём с изучения графического редактора. Отметим, что принципиально графические редакторы, как правило, не сильно отличаются друг от друга, поэтому, освоив один, вы быстро разберётесь, как работать и с любым другим. Речь вовсе не идёт о том, чтобы создавать шедевры живописи, используя компьютер и графический редактор (хотя, быть может, и здесь они бы пригодились, например для отработки композиции). Мы говорим прежде всего о подготовке деловых документов и видеоматериалов. Для их оформления графический редактор весьма полезен. При этом лёгкость корректирования только одно из многих его замечательных свойств.

Перечислим стандартные инструменты и возможности, которыми обладает практически любой графический редактор. Стандартными эти инструменты называются потому, что позволяют делать то, что человек обычно делает с помощью карандаша, линейки, циркуля, ластика и тому подобных привычных инструментов.

Карандаш — смысл ясен из названия. При нажатой (обычно левой) клавише мыши курсор оставляет за собой линию заранее выбранного цвета.

Отрезок позволяет начертить отрезок прямой линии. Для этого надо отметить мышью начальную точку, растянуть отрезок до конечной точки и снова нажать клавишу мыши.

Прямоугольник позволяет рисовать прямоугольники выбранного цвета любых размеров со сторонами, параллельными краям экрана. Перемещением мыши выбирается нужный размер прямоугольника.

Овал ещё одна из стандартных фигур. Выбрав этот инструмент, нужно установить курсор в центр будущего овала и зафиксировать его (в соответствии с инструкцией). Затем перемещением мыши выбирается нужный размер овала. Этим же инструментом легко рисуются и окружности.

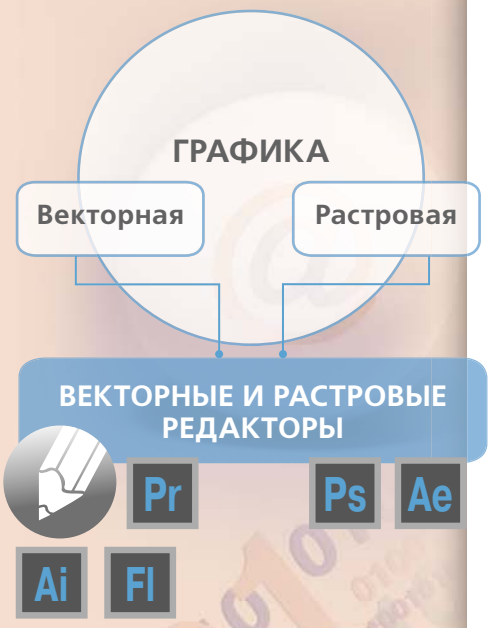
Выбор толщины линии — это необходимо не только для карандаша, но и при рисовании любых стандартных фигур.

Ластик — вы можете выбрать размер ластика, а затем стирать ненужные фрагменты рисунка. Как правило, стирание происходит цветом фона.

Разбрызгиватель разбрызгивает краску (как будто из аэрозольного баллончика).

Заливка заливает выбранным цветом часть рисунка, ограниченную замкнутым (необязательно одноцветным) контуром. Но если в контуре есть отверстие хотя бы в один пиксель, краска разольётся по всему рисунку. К счастью, беду можно поправить кнопкой *Отмена*.

Первый графический редактор был создан в 1963 году. Именно в нём впервые был использован исключительно графический интерфейс, в том числе те инструменты рисования, о которых здесь рассказывается. Он предназначался как для инженерного проектирования, так и для работы дизайнеров.



Узор — залить можно не только сплошным цветом, но и некоторым рисунком. Можно, например, сделать заливку красными кирпичами или фиолетовыми цветами.

Редактирование узора позволяет изменить стандартный узор.

Лупа — после выбора этого инструмента появится небольшой прямоугольник. Наложив его на часть рисунка, которую хотелось бы рассмотреть лучше, можно увидеть, какой цвет имеет каждый пиксель. Это особенно полезно проделать перед заливкой. С помощью лупы легко отыскать малейшие отверстия в контуре.

Отмена (Откатка) даёт возможность отменить только что выполненное действие, вернувшись к предыдущему изображению. Например, с помощью этого инструмента можно тут же отменить заливку, если что-то не понравилось.

Стандартные фигуры, создаваемые графическим редактором, нередко называют г**рафическими примитивами.** Графические примитивы можно перемещать по экрану, накладывать один на другой, копировать, вращать и т.п. В частности, к ним можно применять различные спецэффекты.

ОСОБЕННОСТИ РАСТРОВОЙ И ВЕКТОРНОЙ ГРАФИКИ Надо понимать, что на экране компьютера изображение, независимо от того, какой графический редактор используется — растровый или векторный, всегда представляет собой совокупность нескольких сотен тысяч пикселей. Вместе с тем, в редакторе любого вида присутствуют инструменты, позволяющие работать с графическими примитивами (т.е. как бы векторно) и с отдельными пикселями изображения. Но вот после того как рисунок создан, в растровом редакторе он просто совокупность пикселей, а в векторном редакторе сохраняются те составные части, из которых был создан рисунок.

В векторном редакторе, для того чтобы зафиксировать расположение фигур относительно друг друга, применяется команда *Сгруппировать*. После этого фигуры образуют единое целое, с которым можно работать как с одним объектом — перемещать, изменять размеры и т.д. Если нужно изменить часть сгруппированного объекта, то объект нужно разгруппировать.

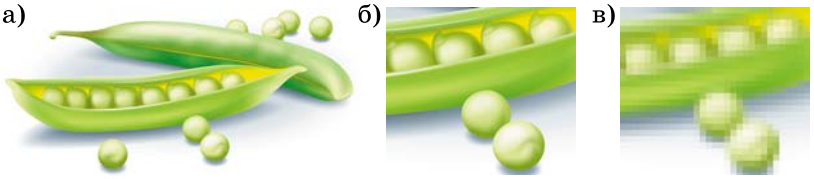
Ещё одна особенность растровых изображений — чувствительность к уменьшению и увеличению созданного рисунка. Дело в том, что при уменьшении несколько соседних пикселей преобразуются в один, и потому теряется чёткость мелких деталей. При увеличении, наоборот, растровые элементы добавляются, а их параметры, определяющие цвет и яркость, берутся такими же, как у исходных пикселей. Появляется ступенчатый эффект. Об этом важно помнить. Если вы сначала уменьшили рисунок, а потом его увеличили, то вовсе не вернётесь к тому рисунку, что был у вас вначале!

Для векторных изображений таких проблем нет. Масштабирование (т.е. увеличение или уменьшение) приводит к появлению коэффициентов в формулах, а сам процесс создания рисунка остаётся прежним. На рисунке изображение в пункте *б* увеличено как векторное, а в пункте *в* — как растровое.

Поэтому если требуется иметь один и тот же рисунок разного размера (например, эмблему фирмы или какой-либо

знак), чтобы размещать его на разной печатной продукции, то предпочтительнее от растрового изображения перейти к векторному. Современные компьютерные технологии позволяют сделать это без существенных усилий. Одной из прикладных программ, позволяющих выполнить такую работу, является Adobe Illustrator.

а) б) в)



В то же время в векторной графике труднее создаются плавные переходы цвета, детали рисунка имеют более явно выраженные контуры.

На рисунках нередко приходится делать те или иные надписи. Обычно в графическом редакторе вы можете использовать всё то многообразие шрифтов, которое имеется и в текстовом редакторе. Кроме того, в графическом редакторе очень легко придать надписям самую разную форму.

В каждом графическом редакторе есть спецэффекты, с помощью которых можно перевернуть фрагмент картинки, перекосить его, зеркально отразить, оставить от него только тень, изменить контур, получить копию объекта... Одним словом, изменить иногда до неузнаваемости картинку или её фрагмент. С их помощью можно, например, нарисовав всего одно деревце, создать за пару минут целую рощу.

Фрагменты рисунка можно вращать. Надо только помнить, что при векторном представлении графической информации выполнение поворота осуществляется пересчётом формул, задающих изображение. Поэтому, например, при повороте рисунка сначала на 120°, а потом в том же направлении на 240° даст в точности исходный рисунок. В случае растрового редактора при каждом повороте будут неизбежно происходить изменения растровой структуры, и окончательный рисунок будет отличаться (как правило, в худшую сторону) от исходного.

ВОПРОСЫ:

- Что такое растр?
- Чем растровый способ создания изображения отличается от векторного?
- В чём преимущества и в чём недостатки растрового способа создания изображения? А векторного?
- Какие имеются средства для работы с компьютерной графикой?
- С помощью каких внешних устройств можно преобразовать рисунок на бумаге в изображение на экране компьютера?

16

ТОНКАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ

ВЫ УЗНАЕТЕ:

- Какие существуют инструменты для работы с графическими объектами сложной формы

ВСПОМНИТЕ:

- Основные инструменты создания и обработки графических изображений

По мере развития компьютерной техники развивались и технологии обработки информационных объектов. Как именно они развивались, мы рассмотрим на примере одного из популярных редакторов Adobe Photoshop, хотя сегодня аналогичные возможности предоставляют и другие редакторы (например, редактор GIMP).

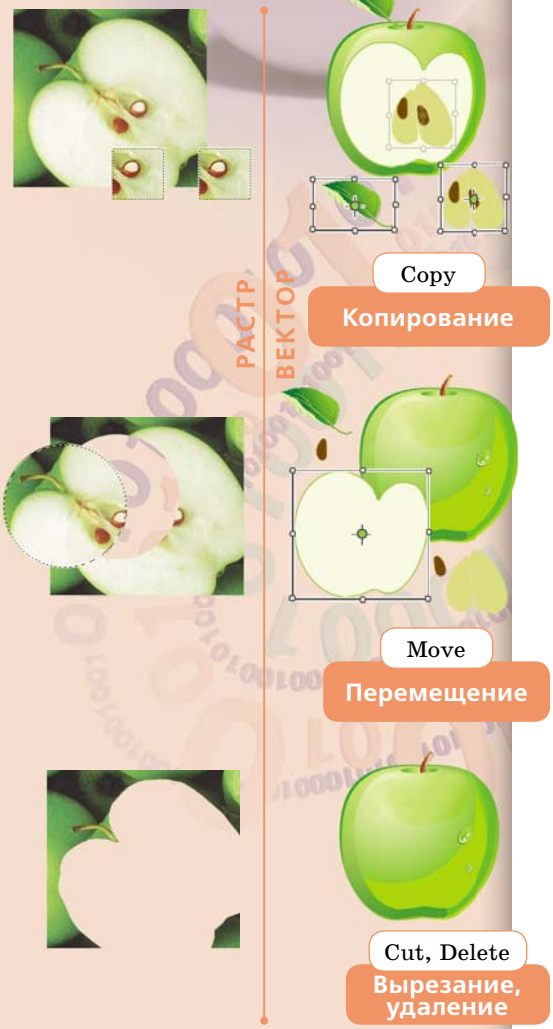
ДОПОЛНИТЕЛЬНЫЕ ИНСТРУМЕНТЫ ГРАФИЧЕСКОГО РЕДАКТОРА Один из дополнительных инструментов графического редактора — это *Лассо*, инструмент, позволяющий выделять объекты более сложной формы, а также выделять области на основе цвета. Один из вариантов *Лассо*, обозначенный пиктограммой , позволяет обрисовывать выделяемую область движением мыши по краю области при нажатой левой клавише мыши.

Многоугольное (*полигональное*) *лассо* (его пиктограмма выглядит так:) предназначено для получения сложной выделенной области, ограниченной отрезками прямых. Каждый щелчок левой клавишей мыши фиксирует точку начала или конца отрезка. Двойной щелчок левой клавишей завершает построение выделенной области соединением первой и последней точек. Другой способ завершения — щелчок в первой точке, с которой было начато построение выделяемой области. При этом у курсора снизу справа появляется окружность, означающая замыкание выделяемой области.

Инструмент *Магнитное лассо* (его пиктограмма:) позволяет выделить в изображении область неправильной формы перемещением мыши вдоль границы области. В местах изменения кривизны (перегиба фигуры) нужно щёлкнуть левой клавишей, чтобы кривая выделения меняла на этом участке своё направление. Линия выделения сама изгибается за курсором по разделу цветовых оттенков. Окончание выделения — двукратный щелчок левой клавишей, при этом контур автоматически замыкается.

Все предыдущие инструменты выделяют указанную форму области, а инструмент *Волшебная палочка* (её пиктограмма:) позволяет создавать выделенную область, исходя из цветовой близости рядом расположенных пикселей. Её удобно применять для выделения областей с одинаковыми или близкими цветами. В панели свойств этого инструмента имеется параметр *Допуск*, определяющий цветовой диапазон выделения. Значение этого параметра может быть от 0 до 255. Чем меньше значение, тем ближе будут цвета, которые попадут в выделенную область. При допуске 0 выделятся только элементы указанного цвета. При допуске 255 в выделенную область попадут все цвета изображения.

Над выделенными с помощью любого инструмента фрагментами можно выполнять разнообразные операции: копирование, перемещение в пределах данного изображения, удале-



ние, копирование и перенос в другие документы и многое другое.

ДОПОЛНИТЕЛЬНЫЕ МЕХАНИЗМЫ ГРАФИЧЕСКОГО РЕДАКТОРА Второй важной особенностью является механизм слоёв. Редактировать растровый графический объект сложной структуры весьма непросто. Чтобы облегчить эту работу, можно распределить фрагменты изображения по отдельным слоям — редактировать его станет намного удобнее. Опишем типы слоёв.

При создании нового документа сначала создаётся один экранный лист указанного размера, называемый **фоном**. Он не может иметь прозрачных участков, так как всегда является самым нижним. Но зато над ним можно надстроить ещё целый ряд дополнительных слоёв — прозрачных плёнок, называемых **изобразительными слоями**. В каждом изображении может быть до 100 слоёв. На каждом слое вы создаёте фрагмент рисунка, а остальная часть изобразительного слоя остаётся прозрачной. Сквозь прозрачные участки слоя видны слои, лежащие ниже. Фрагмент на данном слое вы можете редактировать независимо от фрагментов рисунка, расположенных на других слоях.

Особый вид слоёв — **корректирующие слои**. Они могут работать как маски, пропуская из нижележащих слоёв в общее изображение то, что попадёт в «отверстие» в маске. С их помощью можно применять различные эффекты к нижележащим слоям, например осуществлять цветовую и тоновую коррекцию. Меняя параметры корректирующего слоя, вы получаете разные результаты, никак не затрагивая самого изображения.

Для выполнения действий над слоями имеется специальная палитра, вызов которой осуществляется через меню *Окно*, *Показать слои*. В палитре имеется список слоёв с миниатюрами — уменьшенными изображениями — и названиями слоёв. В каждый момент можно редактировать только один активный слой. Для активации другого слоя следует щёлкнуть по строке с его названием.

Для активного слоя в верхней части палитры можно изменить уровень прозрачности накладываемой краски: 0% означает, что слой остаётся прозрачным; 100% означает, что исходное изображение полностью закрашивается. Это действие можно выполнять, только если установлен режим *Норма*; именно он устанавливается по умолчанию. В нём при рисовании происходит полная замена исходного цвета желаемым.

СОХРАНЕНИЕ ГРАФИЧЕСКОГО ОБЪЕКТА Пока вы работаете в редакторе, изображение сохраняется в формате редактора с расширением **psd**. Но такой файл занимает очень много места на диске, а большинство других редакторов не умеют его читать. У финишной черты — перед вставкой на сайт или в презентацию — необходимо включить видимость всех слоёв изображения и выполнить объединение слоёв командой *Слой*, *Объединить видимые*. После этого вам будут доступны команды сохранения в широко применяемых форматах **gif**, **jpeg** (jpg), где размеры файлов уменьшаются в десятки раз.

Чтобы иметь возможность работать с фрагментом общего рисунка, не затрагивая других его частей, этот фрагмент размещают на отдельном экранном листе, называемом **изобразительным слоем**. Кроме изобразительных, имеются **корректирующие слои**, которые играют роль масок, изменяющих отображение ниже лежащих слоёв. Общее изображение создаётся наложением слоёв.

ВОПРОСЫ:

- Назовите преимущества, которые даёт механизм слоёв для формирования изображения.
- Перечислите типы слоёв, существующих в Adobe Photoshop. Каково назначение слоя каждого типа?
- Какие способы выделения обрабатываемой области рисунка предусмотрены в Adobe Photoshop?

17

ЦВЕТОВЫЕ МОДЕЛИ

ВЫ УЗНАЕТЕ:

- Что такое яркость и насыщенность цвета
- Как измеряют яркость цвета
- Как измеряется насыщенность цвета
- Как кодируется оттенок цвета
- Что такое контраст

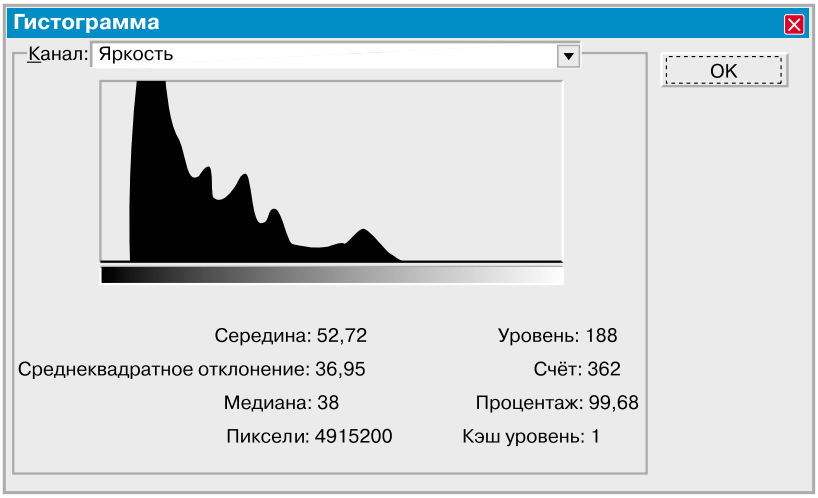
ВСПОМНИТЕ:

- Что такое RGB-модель кодирования цвета

Попросите любого художника назвать оттенки красного цвета и услышите, вероятнее всего, следующее: красный, тёмно-красный, алый, розовый, бледно-розовый и т.п. А ещё, говоря о цвете, обычно отмечают его яркость, насыщенность. Эти характеристики цвета, в отличие от RGB-модели, передают именно субъективное восприятие цвета человеком. Но чтобы эти характеристики можно было использовать при работе на компьютере, их надо формализовать и описать числовыми параметрами.

ЯРКОСТЬ, НАСЫЩЕННОСТЬ И ОТТЕНОК КАК ОСНОВНЫЕ ХАРАКТЕРИСТИКИ ЦВЕТА С физической точки зрения **яркость** — это количественная мера световой энергии, излучаемой или отражаемой в сторону наблюдателя. Так, при солнечном свете и в сумерках один и тот же цвет выглядит по-разному, хотя цветовой оттенок один и тот же. На экране компьютера яркость цвета фактически определяется тем, сколько имеется пикселей данного цвета.

Графический редактор Adobe Photoshop, о котором мы уже рассказывали, позволяет для любого рисунка посмотреть распределение яркостей по тонам. Для этого надо воспользоваться диалоговым окном в меню *Изображение*, *Гистограмма*. Представленная в этом окне диаграмма отображает по горизонтали значение тона, а по вертикали количество пикселей данного тона в изображении, т.е. яркость, освещённость. К примеру, из диаграммы, представленной на рисунке, видно, что яркость исходного рисунка недостаточна: отсутствуют пиксели светлых тонов, зато очень много пикселей тёмных тонов. Вывод: требуется коррекция. Отметим, что гистограмма используется только для получения информации об изображении и для корректировки распределения яркостей в рисунке не предназначена! Такую коррекцию можно сделать, используя диалоговое окно изменения уровней яркости.

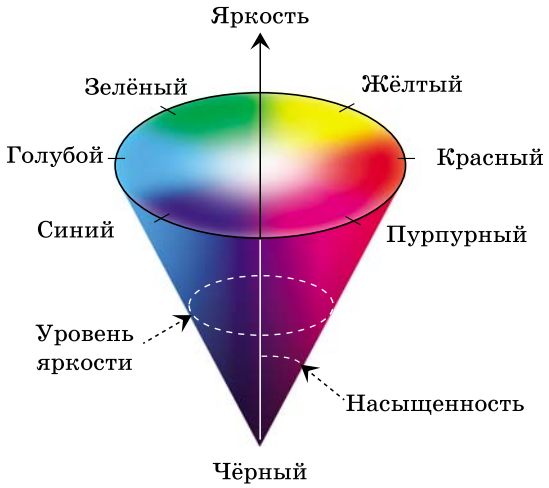


Насыщенность цвета характеризует степень его разбавления белым цветом. Чем больше белого цвета вы добавите, тем меньше насыщенность.

Что касается цветового оттенка, то он определяется расположением цвета в световом спектре. Как вы знаете из физики, видимые человеком цвета располагаются в спектре в следующем порядке: красный — оранжевый — жёлтый — зелёный — голубой — синий — фиолетовый. Нередко их располагают по кругу (см. рисунок), который называют **кругом Манселла**. Тогда цветовой оттенок кодируется либо числом, либо длиной дуги, если считать, что длина всей окружности равна 1. При этом 0° или дуга нулевой длины соответствуют красному цвету. Цвета, которые расположены на противоположных концах диаметра круга Манселла, называются **дополнительными** друг к другу.

Понятие дополнительного цвета важно, в частности, тем, что при коррекции цветов в палитре графического редактора уменьшение доли одного цвета сопровождается увеличением в изображении доли дополнительного цвета.

Чтобы описать две другие характеристики, используют пространственное изображение этой модели в виде конуса. Угол между осью конуса и его образующей характеризует насыщенность цвета, а удалённость от вершины — яркость.



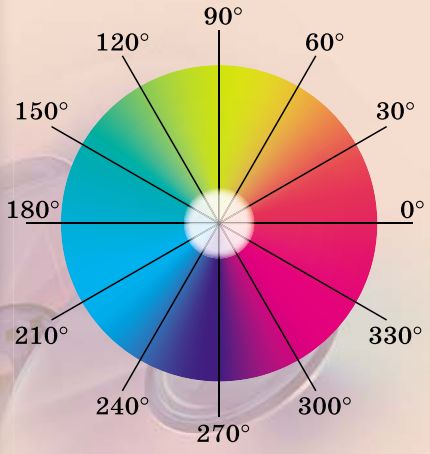
Таким образом, и в этой модели цветопередача характеризуется тремя числами. Сама модель получила название **HSB** — по первым буквам слов hue (цветовой оттенок), saturation (насыщенность) и brightness (яркость).

Специфика работы с цветом заключается в том, что изменение одного цвета обязательно отразится на других. Поэтому в основе любой цветовой коррекции изображения лежит изменение характеристик не отдельных цветов, а настройка **баланса цветов**. Уменьшая в рисунке яркость одного цвета, вы увеличиваете яркость дополнительного.

Кроме яркости, важной характеристикой изображения является его контрастность. **Контраст** — степень тонового различия между областями изображения. Степень контраста обычно измеряют в процентах.

Вызов окна настройки яркости и контраста осуществляется командой *Изображение*, *Настройки*, *Цветовой баланс*.

Круг Манселла



ИССЛЕДОВАНИЯ

В соответствии с законом трёхмерности любой цвет можно получить смешением трёх независимых цветов. А какой цвет получится, если смешать два дополнительных друг к другу цвета?

ВОПРОСЫ:

- Какими характеристиками цвета оперирует HSB-модель?
- Что такое контраст? Что такое яркость изображения?
- Как кодируется цветовой оттенок в HSB-модели?
- Какие цвета называются дополнительными? Какую роль играет соотношение между дополнительными цветами при работе с ними в палитре графического редактора?

18 МУЛЬТИМЕДИЙНЫЕ ПРЕЗЕНТАЦИИ

ВЫ УЗНАЕТЕ:

- Как представить результаты своей работы
- Какие компьютерные средства позволяют создавать мультимедийные презентации

ВСПОМНИТЕ:

- Какая информация называется мультимедийной

Каждый информационно грамотный пользователь компьютерной техники должен уметь не только использовать её возможности для получения информации и решения возникающих перед ним задач, но и представлять результаты своей работы в как можно более доступном виде.

ЦЕЛИ СОЗДАНИЯ ЭЛЕКТРОННЫХ ПРЕЗЕНТАЦИЙ Эффективность восприятия информации возрастает, если она будет представлена в разных видах — видео, звуковой, текстовой. Напомним, что такое представление информации принято называть **мультимедийным**. Для создания мультимедийных документов имеется специальное инструментальное программное обеспечение.

Нередко презентации создаются для сопровождения выступления или доклада по какому-либо вопросу. Ведь хочется, чтобы выступление было интересным, информационно насыщенным, запоминающимся. Презентация, которую вы подготовите заранее, будет вашим надёжным помощником.

На выставке презентация может освободить вас от необходимости много раз повторять один и тот же текст разным посетителям. Презентацию, показываемую компьютером автоматически, т.е. без участия человека, обычно называют **слайд-фильмом**.

ПОДГОТОВКА И СОЗДАНИЕ ПРЕЗЕНТАЦИИ Основным структурным элементом презентации является **слайд**. Во время демонстрации на смену одному слайду приходит другой. Работу по созданию презентации можно разбить на следующие этапы:

- подготовка: выбор темы, определение цели, круга слушателей, содержания и структуры презентации в целом и каждого слайда;
- создание слайдов: размещение объектов на слайде, выбор фона, применение эффектов, подключение звука;
- создание последовательности слайдов: определение переходов от слайда к слайду, создание меню и т.п.;
- репетиция выступления с презентацией;
- редактирование презентации: внесение изменений по результатам репетиции.

Первый этап полностью ложится на вас. А вот про второй мы сейчас расскажем.

Любая презентация начинается с так называемого **титульного слайда**. На нём приводятся название или тема презентации и фамилии авторов того проекта, который она представляет. Такой слайд нередко содержит **логотип** того учреждения, сотрудниками которого осуществлялся проект. К примеру, если вы на научно-практической конференции защищаете свою творческую работу, то очень хорошо на титульном слайде выглядит логотип школы, в которой вы учитесь. И всем сразу понятно — вы уважаете труд ваших наставников, которые помогли вам достичь определённых высот в знаниях и умениях.

Полезно, если на следующем слайде после титульного вы представите основную цель и задачи вашего проекта.

Не забудьте в заключительных слайдах презентации указать источники информации, которые использовались при её создании.

Наверно, вам захочется получить отклики о вашей работе. Тогда на заключительном слайде укажите, как с вами связаться.

И поблагодарите слушателей за внимание к вашей работе.

А теперь поговорим о создании слайдов.

Прежде всего надо выбрать макет будущего слайда. Компьютер любезно предложит вам несколько вариантов. Можете, например, выбрать макет без разметки — ведь интересно всё попробовать сделать самому.

На поле слайда вы можете добавлять самые разнообразные объекты: это и стандартные фигуры, и надписи, и объекты из библиотек готовых рисунков. А может быть, это будет ваш личный рисунок. Можете сделать гиперссылки на звуковые или видеофайлы. Вставки объектов делаются так же, как, например, в обычном текстовом редакторе.

Как правило, любой размещённый на слайде объект можно заставить двигаться. В этом случае говорят, что на объекте настроена анимация. Само слово «анимация» означает «оживление» (от англ. animal — животное). В первую очередь определяют порядок и время анимации. Можно выбрать два режима для запуска объекта в движение: по щелчку мыши или автоматически. В последнем случае надо указать время. Затем следует назначить эффекты анимации. Среди эффектов есть визуальные — возникновение, вращение, вспышка и т.п., а есть звуковые — барабан, аплодисменты и т.п. Впрочем, звук вы можете добавить, используя любой звуковой файл.

Создав слайд, вы можете тут же просмотреть его.

Перейдём к третьему этапу. Прежде всего заметим, что предусмотрено несколько режимов отображения созданной презентации.

Режим отображения	Назначение режима
Обычный	Именно в этом режиме создаются слайды. Могут отображаться панели *Структура*, *Слайд*, *Заметка*
Страница заметок	Каждый слайд на экране сопровождается пустой страницей ниже основного кадра. Здесь можно разместить комментарий к данной презентации
Сортировщик слайдов	Организует порядок следования слайдов
Показ слайдов	Показ слайдов в том виде, в каком они будут демонстрироваться в презентации

Переходить от одного способа отображения к другому можно с помощью кнопок, расположенных в нижней части окна, или выбирая соответствующую команду на закладке *Вид*.

ВОПРОСЫ:

- В чём преимущество мультимедийных продуктов перед другими видами информации?
- Для каких целей может создаваться презентация?
- Каковы основные этапы разработки электронной презентации?

ВСПОМНИТЕ:

- Что такое редактирование текста
- Что называют форматированием текста
- Основные параметры форматирования
- Основные характеристики шрифта
- В каких единицах измеряются межстрочный и межсимвольный интервалы
- Что такое адрес ячейки электронной таблицы
- Что такое абсолютная и относительная адресация ячеек электронной таблицы
- Чем запись адреса ячейки при абсолютной адресации отличается от записи адреса при относительной адресации
- Что такое блок ячеек
- Какие требования предъявляются к записи формул в ячейки электронной таблицы
- В чем состоят различия между столбчатыми и круговыми диаграммами
- Что такое яркость, насыщенность и оттенок цвета
- Как кодируются характеристики цвета в модели HSB

ТЕОРЕТИЧЕСКИЙ ПРАКТИКУМ

Чтобы вы могли проверить, хорошо ли вами усвоен изученный материал, предлагаем выполнить следующие задания. Более широкий комплекс заданий вы найдёте в задачнике-практикуме.

КОМПЬЮТЕРНАЯ ОБРАБОТКА ТЕКСТА

1 а) Из приведённого списка действий по преобразованию текста укажите те, которые, по вашему мнению, можно производить с помощью текстового редактора:

- автоматическая вставка данного символа между двумя заданными символами во всём тексте;
- автоматическая замена одного слова (и только его!) другим во всём тексте;
- автоматическое заключение в кавычки заданного слова во всём тексте;
- автоматическая замена числа, набранного цифрами, соответствующим числительным, набранным буквами;
- подсчёт, сколько раз в тексте встречается заданное слово;
- автоматическая ликвидация всех пробелов между словами;
- автоматическое удвоение всех пробелов;
- автоматическая вставка пустой строки после каждого абзаца во всём тексте;
- автоматическая ликвидация пустых строк между абзацами во всём тексте.

б) Из действий, перечисленных в пункте *а*, которые, по вашему мнению, нельзя поручить выполнить текстовому редактору автоматически сразу во всём тексте, укажите те, которые можно выполнить пошагово, т.е. каждый раз давая компьютеру разрешение на выполнение данного действия или отказывая ему в этом.

2 Пользуясь информацией со с. 32, подсчитайте, сколько строк текста поместится на странице формата А4, если верхнее и нижнее поля имеют величину 2 см, а текст набран через полуторный интервал кеглем 14 пунктов.

ОРГАНИЗАЦИЯ ВЫЧИСЛЕНИЙ ПРИ ПОМОЩИ КОМПЬЮТЕРА

3 а) Первоначально ячейки электронной таблицы были заполнены числами так, как показано в таблице. Затем в ячейку B4 записали формулу =C$1 + $B3 и скопировали её в ячейку A5. Какое значение будет вычислено в ячейке A5?
б) Объясните, почему формулу из ячейки B4 нельзя скопировать в ячейку D1.

	A	B	C	D	E
1	2	3	5	7	11
2	1	4	9	12	8
3	3	7	1	6	5
4	5	2	8	5	1
5	2	6	4	2	1

4 Запишите вид формул, которые будут введены соответственно в ячейки B1, C1, D1 и E1 для вычисления значений функций

$$y = 2x^3 - 3x + 1,\ y = \frac{3x}{x^2+1},\ y = \frac{3x-1}{1-x} \text{ и } y = \frac{1+x^2}{3x},$$

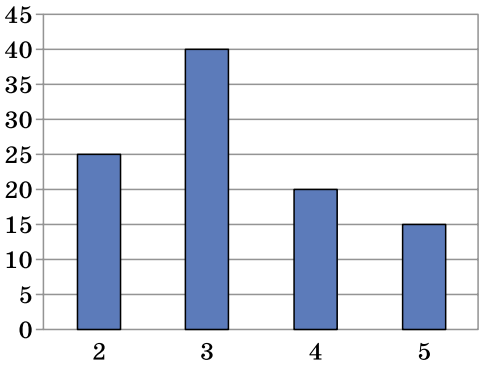
если значение аргумента x записано в ячейку A1.

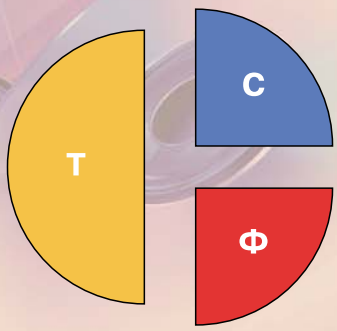
5 Дан фрагмент электронной таблицы в режиме отображения формул (см. таблицу). Затем формула из ячейки A4 была

скопирована в ячейки A5, B4 и B5. Какая формула будет записана в ячейке B5? Какое число окажется в ячейке B5 после того, как будут выполнены вычисления?

	A	B
1	1	1
2	2	
3	3	
4	=СУММ($A1:A3)	
5		

6 В цехе трудятся рабочие трёх специальностей — токари (Т), слесари (С) и фрезеровщики (Ф). Каждый рабочий имеет разряд не меньший второго и не больший пятого. На диаграмме внизу отражено количество рабочих с различными разрядами, а на диаграмме справа — распределение рабочих по специальностям. Каждый рабочий имеет только одну специальность и один разряд.





Имеется несколько утверждений:
А) Все рабочие третьего разряда не могут быть токарями.
Б) Все рабочие третьего разряда могут быть фрезеровщиками.
В) Все слесари могут быть пятого разряда.
Г) Кто-то из токарей имеет разряд ниже четвёртого.
Д) Если среди токарей нет рабочих пятого разряда, то кто-то из токарей имеет третий разряд.
Е) Среди фрезеровщиков обязательно есть рабочий, разряд которого выше второго.
Укажите, какие из этих утверждений не противоречат данным обеих диаграмм.

СОЗДАНИЕ И ОБРАБОТКА ВИЗУАЛЬНЫХ КОМПЬЮТЕРНЫХ ОБЪЕКТОВ

7 Какие цвета на круге Манселла кодируются числами 120, 150, 210?

8 Укажите, где на цветовом круге Манселла располагаются коричневый, оранжевый и фиолетовый цвета. Какими числами, на ваш взгляд, эти цвета кодируются в модели HSB?

9 Каким числом на круге Манселла кодируется цвет, дополнительный к цвету, имеющему код n?

ВЫ УЗНАЕТЕ:

- Как выполняются основные операции над текстом в текстовых редакторах
- Как в текстовом редакторе работать с таблицами и векторной графикой

ВСПОМНИТЕ:

- Каковы основные характеристики текста
- Что называют форматированием текста

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ

Выполнять задания лабораторных работ № 3 и 4, посвящённых технологии обработки текста, вы будете в компьютерном классе. Чтобы выполнить задания лабораторных работ, вам потребуются тексты, которые необходимо обработать в текстовом редакторе. Файлы с этими текстами лучше подготовить заранее, чтобы не тратить время на уроке.

ЛАБОРАТОРНАЯ РАБОТА № 3. ОСНОВНЫЕ ИНСТРУМЕНТЫ ТЕКСТОВОГО РЕДАКТОРА Несколько первых заданий посвящены таким действиям, как вставка, замена и копирование, разбиение на абзацы и т.п.

1 Некто, набиравший текст стихотворения, постоянно забывал вовремя нажимать клавишу *Конец строки* (хотя и начинал новую строфу каждый раз с заглавной буквы). Получилась одна длинная строка:

ПОЛОСАТАЯ ЖИЗНЬ Друг за другом очень часто Ходит счастье и несчастье. От того и получается такая Полосатая жизнь. И не надо зря тратить нервы. Вроде зебры жизнь, да вроде зебры: Чёрный цвет, а потом будет белый цвет — Вот и весь, вот и весь секрет.

а) Определите (не пересчитывая!), сколько символов в получившейся строке.

б) Преобразуйте текст стихотворения (не набирая заново ни одного символа!) к обычному виду. Не забудьте выделить в отдельную строку заголовок и поместить его по центру строки.

2 Ниже записаны в случайном порядке половинки пословиц. Каждая половинка заканчивается многоточием; первая буква в пословице — заглавная.

Слово — не воробей...	рот не разевай...
И хочется, и колется...	Своя рубашка...
вылетит — не поймаешь...	придётся напиться...
Не плюй в колодец...	и мама не велит...
Дурная голова...	а сам не плошай...
Чем дальше в лес...	ближе к телу...
ногам покоя не даёт...	На чужой каравай...
Делу — время...	На Бога надейся...
тем больше дров...	потехе час...
Что с возу упало...	то не вырубишь топором...
Друга ищи...	всем — нет...
Что написано пером...	то пропало...
Старый друг...	а найдёшь, береги...
Одному страшно...	а недруг — ложь...
Друг скажет правду...	лучше новых двух...

Восстановите правильные тексты пословиц (не набирая ни одной буквы!), заменив многоточия нужными знаками препинания или пробелом. Каждая пословица должна быть записана в отдельной строке.

3 В приведённом ниже списке неправильно соотнесены фамилии поэтов и писателей с именами и отчествами.

Михаил Михайлович Горький
Александр Сергеевич Чехов
Михаил Юрьевич Пушкин
Самуил Яковлевич Крылов
Николай Алексеевич Лермонтов
Агния Львовна Гайдар
Константин Георгиевич Пришвин
Алексей Максимович Маршак
Аркадий Петрович Паустовский
Иван Андреевич Толстой
Антон Павлович Некрасов
Лев Николаевич Барто

Не набирая ни одной буквы, составьте правильный список.

4 В приведённом ниже стихотворении символ # заменяет одно и то же сочетание букв. Догадайтесь, какое именно, и сделайте в автоматическом режиме замену, восстанавливающую текст стихотворения.

«О, Л#! — возмутился Ол#. —
Чего ты лежишь, словно п#?»
«Тю, Л#! — удивился Тюл#. —
Чего ты блуждаешь, как т#?»
Но им не ответила Л#: «Л#!»

5 В приведённом ниже стихотворении число 100 в некоторых словах заменяет трёхбуквенное сочетание «сто». Выполните обратную замену. Обратите внимание на то, что в двух случаях заменять число 100 данным сочетанием не следует.

У про100го 100рожа непро100рный дом:
Ча100 в нём 100ножка бродит под 100лом.
Дорожит 100ножка чи100тою ног
И 100личной ваксой чистит 100 сапог.
Вме100 двух не про100 вычистить все 100,
Сразу 100лько обуви не носил никто!
У про100й 100ножки 100ит по100ять
И у той 100ножки опыт перенять!

Лабораторная работа завершена.

ЛАБОРАТОРНАЯ РАБОТА № 4. РАБОТА СО ШРИФТАМИ. ОФОРМЛЕНИЕ ТАБЛИЦ И ВЫЧИСЛЕНИЯ В ТАБЛИЦАХ. ГРАФИЧЕСКИЕ СРЕДСТВА ТЕКСТОВОГО РЕДАКТОРА

1 В текст благодарственного письма вставьте реальные данные (вместо текста в скобках) и продумайте, какие шрифтовые выделения удобно было бы использовать.

Благодарственное письмо

Уважаемые (имя и отчество родителей)!

Дирекция школы (лицея, гимназии) № (номер) г. (название города) благодарит Вас за воспитание вашего (сына, дочери)

(ФАМИЛИЯ, ИМЯ),

которое выражается (в хорошей успеваемости, активной общественной деятельности, примерном поведении, добром отношении к товарищам и т.п.).

Директор (школы, лицея, гимназии) № (номер)

(Фамилия, имя, отчество)

2 В приведённом ниже тексте представлены лирические стихотворения русских поэтов разного времени.

Пушкин А. С. «Я вас любил, любовь ещё, быть может...», 1829; Лермонтов М. Ю. «Над морем красавица-дева сидит...», 1829; Фет А. А. «Ещё люблю, ещё томлюсь...», 1890; Блок А. А. «Гадай и жди. Среди полночи...», 1902; Волошин М. А. «Раскрыв ладонь, плечо склонила...», 1910; Ахматова А. А. «Любовь покоряет обманно...», 1911; Цветаева М. И. «Вчера ещё в глаза глядел...», 1920; Набоков В. В. «О, любовь, ты светла и крылата...», 1923; Есенин С. А. «Глупое сердце, не бейся!..», 1925; Маяковский В. В. «Любит? не любит? Я руки ломаю...», 1928; Пастернак Б. Л. «Любить иных — тяжёлый крест...», 1931; Симонов К. М. «Жди меня, и я вернусь...», 1941; Евтушенко Е. А. «Я разлюбил тебя... Банальная развязка...», 1966; Высоцкий В. С. «Красивых любят чаще и прилежней...», 1968.

а) Поручите компьютеру преобразовать этот текст в таблицу с колонками:

Автор	Первая строка стихотворения	Год создания

б) Добавьте слева столбец и поручите компьютеру пронумеровать строки в таблице, начиная со второй.
в) В вашей таблице теперь четыре столбца, и фамилии авторов расположены в строках второго столбца. Отсортируйте по алфавиту строки второго столбца, начиная со второй строки (у вас там сейчас должно быть написано: Пушкин А. С.). Проследите, чтобы при этом осталось правильное соответствие между содержанием второго и третьего столбцов, а строки оказались верно пронумерованными.

3 Ниже приведена заготовка таблицы.

Расчёт почасовой оплаты труда сотрудников

	A	B	C	D	E	F
1	Фамилия	Плата за 1 час, р.	Количество часов	Плата за часы, р.	Подоходный налог, р.	Итого
2	Иванов	150	82			
3	Петров	230	75			
4	Сидоров	180	94			
5	Степанов	340	39			
6	Михайлов	270	76			
7	Данилов	300	45			

а) В ячейки столбца D введите формулу, позволяющую вычислить оплату труда каждого из работников.
б) В ячейки столбца E введите формулу, позволяющую вычислить подоходный налог — 13% от начисленной суммы.
в) В ячейки столбца F введите формулу, позволяющую вычислить сумму, которую следует выдать работнику в качестве заработной платы.

4 Ниже приведён список населённых пунктов и указаны расстояния между ними.

1. Васильево — Тюрин: 15 км;
2. Васильево — Полевской: 11 км;
3. Васильево — Фоминск: 9 км;
4. Тюрин — Полевской: 19 км;
5. Тюрин — Мосино: 5 км;
6. Фоминск — Полевской: 7 км;
7. Мосино — Полевской: 10 км.

а) Графическими средствами текстового редактора создайте схему, которая представляла бы информацию о дорожном сообщении между населёнными пунктами. Фрагмент такой схемы, содержащий обозначения двух населённых пунктов и информацию о протяжённости дороги между ними, уже нарисован. Предварительно обдумайте, как расположить обозначения населённых пунктов, чтобы схема имела наиболее удобный вид.
б) Сгруппируйте элементы рисунка, чтобы можно было работать с ним как с единым объектом.

Лабораторная работа завершена.

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ

ВЫ УЗНАЕТЕ:
- Как создавать нумерованные и маркированные списки
- Как нумеровать страницы

ВСПОМНИТЕ:
- Что такое колонтитул
- Чем отличаются нумерованные списки от маркированных

Прежде чем вы приступите к выполнению лабораторной работы этого компьютерного практикума, посвящённого созданию текстовых информационных объектов, вам нужно выбрать какое-либо знакомое небольшое литературное прозаическое произведение. Очень хорошо, если оно будет у вас перед глазами. Ваша задача — составить и оформить подробный план этого произведения. Прежде всего определим уровень детализации. Вы знаете, что в литературном произведении каждый абзац обладает свойством логической завершённости. Поэтому самый нижний уровень детализации, представленный в вашем будущем плане, — это, можно сказать, заголовок для абзаца. В свою очередь, несколько соседних между собой абзацев могут быть объединены в группу, которая также обладает неким логическим единством. Для них естественно иметь свой пункт плана, более высокого ранга, нежели для абзацев, и т.д. Наметьте не менее трёх уровней в вашем будущем плане и приступайте к реализации.

ЛАБОРАТОРНАЯ РАБОТА № 5. СОЗДАНИЕ МНОГОУРОВНЕВЫХ СПИСКОВ

Поскольку ваш план будет иметь не менее трёх структурных уровней, то и создавать его имеет смысл в виде многоуровневого списка. Схема такого списка представлена на поле слева.

Заголовок произведения
1. Заголовок первой части
   а. Заголовок первой группы абзацев
      - Заголовок первого абзаца
      - Заголовок второго абзаца
      - ...
   б. Заголовок второй группы абзацев
      - Заголовок первого абзаца
      - Заголовок второго абзаца
      - ...
   в. ...
2. Заголовок второй части
3. ...

1 Чтобы создать такой список, надо сначала воспользоваться кнопкой *Нумерация*.

Чтобы создать в этом списке элементы разного уровня, надо установить курсор в начало строки и нажать клавишу Tab. Это переведёт элемент списка на следующий уровень.

И каждое нажатие клавиши Tab будет переводить элемент списка на следующий, ещё более низкий, уровень. Правда, может оказаться, что символы нумерации и маркировки будут совсем не те, какие вы планировали. Их можно поменять, используя команду *Определить новый маркер*.

Сначала добейтесь, чтобы ваш план выглядел так, как показано на рисунке. Затем поэкспериментируйте и выберите такой вариант оформления, который вам больше всего нравится. Обсудите с одноклассниками ваше решение. Для большей выразительности можете использовать шрифтовые выделения.

Выберите кегль, межстрочный интервал и формат страницы такими, чтобы ваш план занимал не менее двух страниц.

2 Включите автоматическую нумерацию страниц. Обсудите с одноклассниками, следует ли ставить номер на первой странице.

3 А теперь создайте верхний колонтитул. Сделайте так, чтобы на чётных страницах стояла ваша фамилия, а на нечётных — заголовок «План ...».

Создание информационного текстового объекта закончено. Сохраните его.

Теперь создайте ещё один информационный объект. Попытайтесь, используя составленный вами план, передать

в 3—4 строках содержание произведения. Как следует, по вашему мнению, назвать созданный вами информационный объект?
Нередко такой текст, кратко передающий основное содержание, называют **аннотацией** или **резюме**. Подберите для него подходящий шаблон документа.

4 Сохраните созданный объект в файле с подходящим именем.

5 Усовершенствуем созданный вами план произведения. Создайте таблицу из двух столбцов и столько строк, сколько у вас получилось пунктов в плане. Разместите в левом столбце пункты вашего плана так, чтобы каждый пункт занимал ровно одну клетку таблицы.
В каждой строке, где стоит заголовок произведения, заголовок части или группы абзацев, объедините ячейки, чтобы соответствующий заголовок занимал всю строку создаваемой таблицы.

6 А теперь в правом столбце для каждого маркированного пункта плана запишите цитату из произведения, которая, на ваш взгляд, наиболее точно отражает содержание этой группы абзацев. Ниже приведён макет такой таблицы.

Заголовок произведения	
1. Заголовок первой части	
а. Заголовок первой группы абзацев	
• Заголовок первого абзаца	Цитата
• Заголовок второго абзаца	Цитата
• Заголовок третьего абзаца	Цитата
…	
б. Заголовок второй группы абзацев	
• Заголовок первого абзаца	Цитата
• Заголовок второго абзаца	Цитата
• Заголовок третьего абзаца	Цитата
…	
2. Заголовок второй части	
…	

7 А теперь посмотрите, соответствует ли смысл цитаты заголовку группы абзацев. Если не соответствует, то скорректируйте заголовок группы абзацев.

Такой информационный объект обычно называют **планом-конспектом** произведения. Сохраните получившийся информационный объект под другим именем.

Лабораторная работа завершена.

ИССЛЕДОВАНИЯ

Объединение ячеек создаёт в таблице ячейки разной ширины столбцов. Можно ли другим способом создавать ячейки разной ширины в одном столбце? Поэкспериментируйте. Как изменяется таблица, если объединять ячейки соседних строк в одном столбце?

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ

ВЫ УЗНАЕТЕ:
- Какие есть способы копирования данных в электронной таблице
- Как осуществлять сортировку данных по тому или иному параметру

ВСПОМНИТЕ:
- Что такое относительная и абсолютная адресация ячеек
- Как записываются формулы в ячейки электронной таблицы

Выполнять задания лабораторных работ № 6 и 7 этого практикума, посвящённого технологии обработки числовой информации, вы будете в компьютерном классе. Хотя практикум называется компьютерным, это вовсе не означает, что, приступая к выполнению заданий лабораторной работы, нужно тут же включать компьютер. Мы советуем сначала внимательно прочитать задание и поразмышлять, как именно будет использоваться соответствующая компьютерная технология для его выполнения.

Чтобы выполнить задания лабораторных работ, вам будут нужны числовые данные, которые требуется обработать с помощью электронной таблицы. Лучше приготовить соответствующие файлы заранее, чтобы не тратить время на уроке.

ЛАБОРАТОРНАЯ РАБОТА № 6. ОСНОВНЫЕ ИНСТРУМЕНТЫ ЭЛЕКТРОННОЙ ТАБЛИЦЫ

1 В таблицах 2—4 приведены сведения о количестве некоторых товаров, проданных в трёх филиалах торговой фирмы «Товары для дома». Требуется заполнить итоговую таблицу (таблица 1).

Таблица 1. **Количество проданных товаров в трёх филиалах торговой фирмы «Товары для дома»**

Наименование	Помесячно						Поквартально		За полгода
	Январь	Февраль	Март	Апрель	Май	Июнь	1	2	
Чайник									
Кофеварка									
СВЧ-печь									
Мультиварка									
Миксер									

а) Придумайте и запишите в соответствующую ячейку формулу, по которой будет найдено количество чайников, проданных во всех трёх филиалах в течение января.

б) Скопируйте эту формулу в нужные ячейки Таблицы 1 для подсчёта количества проданного товара помесячно.

в) Придумайте и запишите в соответствующую ячейку формулу, по которой будет найдено количество чайников, проданных в 1-м квартале. Используя эту формулу, заполните остальные ячейки столбца для подсчёта количества проданного товара в 1-м квартале.

г) Выполните заполнение ячеек таблицы, позволяющее узнать итоги работы фирмы во 2-м квартале.

д) Выполните заполнение ячеек таблицы, позволяющее узнать итоги работы фирмы за полугодие.

Таблица 2

Количество проданных товаров в Санкт-петербургском филиале торговой фирмы «Товары для дома»						
Наименование	**Январь**	**Февраль**	**Март**	**Апрель**	**Май**	**Июнь**
Чайник	225	123	54	68	45	41
Кофеварка	156	96	73	63	103	59
СВЧ-печь	126	137	49	52	109	64
Мультиварка	43	85	64	99	68	34
Миксер	37	50	99	79	44	71

Таблица 3

Количество проданных товаров в московском филиале торговой фирмы «Товары для дома»						
Наименование	**Январь**	**Февраль**	**Март**	**Апрель**	**Май**	**Июнь**
Чайник	325	223	94	178	65	61
Кофеварка	456	127	93	83	183	79
СВЧ-печь	226	153	69	72	139	84
Мультиварка	143	157	84	199	98	84
Миксер	67	69	109	89	54	171

Таблица 4

Количество проданных товаров в екатеринбургском филиале торговой фирмы «Товары для дома»						
Наименование	**Январь**	**Февраль**	**Март**	**Апрель**	**Май**	**Июнь**
Чайник	100	126	34	80	65	43
Кофеварка	156	67	87	45	84	32
СВЧ-печь	23	56	34	7	43	49
Мультиварка	23	15	26	23	17	21
Миксер	32	56	45	65	76	56

2 В таблице 5 для каждого из спортсменов приведены оценки судей за его выступление. Для определения результата самая высокая и самая низкая из выставленных оценок отбрасываются, а для оставшихся вычисляется среднее арифметическое.

а) В столбец *Результат* запишите формулы, позволяющие для каждого спортсмена вычислить результат, показанный им в соревновании.

б) Расположите фамилии участников в порядке убывания их результатов. Определите, кто стал победителем и кто получил серебряную и бронзовую награды.

3 Правила подведения итогов соревнования изменились. Для определения результата для каждого спортсмена сначала определяется место, которое он занял по мнению одного судьи; если у данного судьи несколько спортс-

Таблица 5

№	Фамилия	1	2	3	4	5	Результат
1	Сотов	9,6	9,4	9,8	9,5	9,3	
2	Петров	9,7	9,2	9,4	9,3	9,6	
3	Минин	9,2	9,5	9,4	9,6	9,7	
4	Козлов	9,3	9,5	9,6	9,4	9,6	
5	Белов	9,6	9,9	9,2	9,3	9,5	
6	Жилин	9,1	9,5	9,4	9,3	9,6	
7	Котов	9,2	9,8	9,2	9,8	9,4	
8	Орлов	9,7	9,4	9,5	9,7	9,5	
9	Иванов	9,5	9,6	9,9	9,9	9,8	
10	Васин	9,1	9,3	9,4	9,1	9,1	
11	Громов	9,5	9,2	9,6	9,5	9,4	

менов имеют одинаковую оценку, то им выставляется среднее арифметическое тех номеров мест, которые они заняли вместе. Например, у судьи № 1 высший балл 9,7 имеют 2 спортсмена (т.е. они занимают 1-е и 2-е места), поэтому в качестве места им указывается число 1,5, а низший балл у того же судьи имеют 3 спортсмена (т.е. им выставляются 10, 11 и 12-е места), поэтому им всем указывается число 11. Затем места, полученные спортсменом у всех судей, суммируются и спортсмены ранжируются по возрастанию этой суммы.

Модифицируйте таблицу так, чтобы для каждого спортсмена вычислялось его место у каждого судьи (для этого для каждого судьи вставьте дополнительный столбец), а затем заполните формулами столбец результатов.

Таблица 6

№	Наименование товара	Цена, у.е.
1	Компьютер Hewlett-Packard 22	460
2	Компьютер Hewlett-Packard 460	520
3	Компьютер Lenovo V330	470
4	Ноутбук ASUS F405	320
5	Ноутбук HP 15ra	295
6	Монитор Samsung S24	115
7	Монитор LG 24MK	140
8	Монитор HP 24fw	155
9	Монитор Samsung S27	180

Таблица 7

Дата	Курс у.е., р.
27 сен.	67,374
28 сен.	67,206
29 сен.	67,252
30 сен.	66,876
1 окт.	66,759
2 окт.	66,778
3 окт.	66,859
4 окт.	66,93

4 В таблице 6 приведены цены на некоторые товары в интернет-магазине, а в таблице 7 — курс условной денежной единицы (у.е.) в рублях за несколько дней. С 1 октября фирма ввела скидку 15% на цену товара, если приобретается более 5 экземпляров этого товара, и скидку 2% с общей суммы покупки, если эта сумма превышает $1500. Создайте таблицы 6–7 на листе электронной таблицы. Листу дайте имя *Информация*.

а) Создайте новый лист электронной таблицы и присвойте ему имя *Прайс-лист*. На нём создайте прайс-лист на товары в зависимости от цены в у.е. и даты продаж. Для этого свяжите таблицу на листе *Прайс-лист* с таблицами на листе *Информация*. В прайс-листе учитывается скидка только на цену товара.

б) Создайте ещё один лист электронной таблицы и присвойте ему имя *Продажи*. На этом листе подсчитайте выручку магазина за каждый день и за 6 дней по каждому товару. Укажите, в какой день выручка была наибольшей, а в какой — наименьшей. Укажите, какой товар за 6 дней продаж принёс наибольший доход, а какой — наименьший.

в) На отдельном листе создайте счёт-фактуру по приведённому образцу (таблица 8).

Лабораторная работа завершена.

Таблица 8. Счёт-фактура

Наименование товара	Цена, р.	Количество, шт.	Стоимость без скидки, р.	Скидка	Стоимость со скидкой, р.
Компьютер Hewlett-Packard 22					
Компьютер Hewlett-Packard 460					
Компьютер Lenovo V330					
Ноутбук ASUS F405					
Ноутбук HP 15ra					
Монитор Samsung S24					
Монитор LG 24MK					
Монитор HP 24fw					
Монитор Samsung S27					
		Итого:		**Итого:**	

ЛАБОРАТОРНАЯ РАБОТА № 7. РАБОТА С ДАННЫМИ В РАЗНЫХ ФОРМАТАХ

В первых двух заданиях данные представлены в форматах дата/время.

1 В таблице приведено расписание авиарейсов из Новосибирска в различные города России и зарубежья. Согласно правилам время вылета и прилёта всегда указывается местное. Однако пункты прилёта располагаются в разных часовых поясах. Эти пояса относительно Москвы также приведены в таблице. Часовой пояс Новосибирска — 3. Требуется найти продолжительность каждого рейса.

Рейс	Направление	Время местное		Часовой пояс	Продолжительность рейса
		Вылет	Прилёт		
СУ 1307	Москва-Шрм	10:15	11:25	0	
S7 853	Урумчи	10:35	13:50	4	
ЯМ 525	Краснодар	11:30	13:00	0	
РД 9172	Красноярск	12:00	14:10	4	
РД 9172	Иркутск	12:00	17:40	5	
S7 3287	Алматы	13:10	14:50	2	
КС 218	Астана	13:40	14:35	2	
ОИ 722	Ош	15:20	17:15	2	
ГЛ 3261	Якутск	15:30	22.40	6	
ОИ 732	Бишкек	15:50	17:15	2	
СП 5024	Стрежевой	16:50	18:30	3	
ХИ 668	Ташкент	17:30	18:35	1	
У6 103	Москва-Дмд	18:40	19:50	0	
S7 823	Бургас	18:40	20:45	1	
ТИ 20	Красноярск	18:45	21:30	3	
ГЛ 176	Москва-Дмд	19:15	20:50	0	
С7 3337	С.Петербург	19:50	21:40	0	
С7 3273	Иркутск	21:25	1:55	5	
С7 3273	Владивосток	21:25	10:10	7	
ИК 756	Иркутск	21:30	1:30	5	
ГЛ 180	Москва-Дмд	21:40	23:15	0	
ДД 4691	Владивосток	21:55	7:35	7	
ЮТ 138	Сургут	22:15	23:45	2	
ДД 4643	Хабаровск	22:20	7:15	7	
S7 3295	Душанбе	22:50	0:30	1	
ЯМ 526	Мирный	0:15	6:05	6	
ЛА 162	Тюмень	0:25	1:30	2	
ЯК 755	Магадан	0:45	11:10	8	
С7 3507	Хабаровск	1:10	10:10	7	
ПЛ 544	С.Петербург	2:00	3:45	0	
S7 863	Пекин	2:10	7:35	4	
ЮТ 5053	Анталья	4:00	6:25	1	

ВЫ УЗНАЕТЕ:

- Как в электронной таблице работать с данными, которые записаны в форматах времени
- Как строить диаграммы с областями и накоплением
- Как построить лепестковую диаграмму

ВСПОМНИТЕ:

- Какие формы графического представления информации реализуемы с помощью стандартных инструментов электронной таблицы

2 Владелец автостоянки хочет автоматизировать расчёты с клиентами за услуги. Информация о приезде автомобиля на стоянку и его отъезде фиксируется автоматически (она представлена в таблице ниже).

а) Заполните расчётными формулами столбцы *Продолжительность стоянки* и *Стоимость*.

б) Найдите суммарную выручку за день.

Номер места	Время приезда	Время отъезда	Продолжительность стоянки, мин	Цена за час, р.	Стоимость, р.
3	6:15	8:25		25	
1	6:38	10:14			
4	7:52	13:10			
2	8:16	10:55			
3	9:28	18:04			
1	10:43	12:38			
2	11:17	15:34			
1	12:42	19:45			
4	14:26	15:09			
2	16:37	20:11			
4	17:12	20:20			
3	18:42	21:55			
1	20:10	22:26			
2	20:15	22:13			
4	20:59	22:50			
3	22:12	22:47			

3 В таблицах слева приведены данные переписей населения, проводившихся в России в 2002 и 2017 гг. Численность населения приведена округлённо в млн человек.

а) Постройте по данным 2002 г. столбчатую диаграмму, сгруппированную по три столбца по возрастам.

б) Постройте по данным 2002 г. три круговые диаграммы, показывающие долю каждой возрастной группы в общей численности.

в) Выполните задания пунктов *а* и *б* по данным 2017 г.

г) На кольцевой диаграмме покажите, как изменилась численность возрастных групп за прошедшие 15 лет.

Возраст	Число жителей, 2002 г.		
	всего	мужчин	женщин
0—9	13,3	6,8	6,5
10—19	23,2	11,8	11,4
20—29	22,1	11,1	11,0
30—39	20,0	9,9	10,1
40—49	24,2	11,6	12,6
50—59	15,4	7,0	8,4
60—69	14,3	5,7	8,6
70 и более	12,1	3,6	8,9

Возраст	Число жителей, 2017 г.		
	всего	мужчин	женщин
0—9	18,1	9,3	8,8
10—19	14,1	7,2	6,9
20—29	19,7	10,0	9,7
30—39	23,8	11,8	12,0
40—49	19,7	9,5	10,2
50—59	21,0	9,5	11,5
60—69	17,2	6,9	10,3
70 и более	13,2	3,9	9,3

4 Ниже в таблице приведены данные о сдаче спортивных нормативов в течение учебного года. Постройте по этим данным диаграмму с областями и накоплением.

Разряд	Начало	Четверть			
		1	2	3	4
1-й разряд	0	1	2	5	5
2-й разряд	1	3	3	2	5
3-й разряд	7	7	8	7	8
Ниже 3-го разряда	12	9	7	6	2

5 В таблице справа приведены данные метеонаблюдений за направлением ветра в течение июня в одном из городов России.

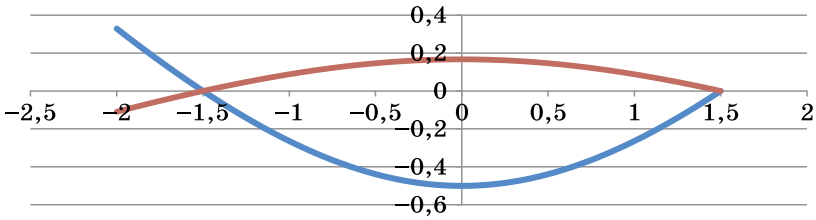
а) Для каждого из направлений подсчитайте, сколько раз в течение дня, ночи и суток дул ветер в данном направлении.

б) Для каждого из получившихся наборов данных постройте лепестковую диаграмму так, чтобы направления её осей соответствовали стандартным направлениям сторон света (север — вверх, восток — вправо и т.д.). Как вы знаете из географии, такая диаграмма называется «Роза ветров».

в) Взлётно-посадочная полоса аэропорта строится обычно так, чтобы взлёт и посадка осуществлялись вдоль преимущественного направления ветров. Используя полученные данные, определите, в каком направлении лучше всего строить взлётно-посадочную полосу.

Дата	День	Ночь
1	В	В
2	С	С
3	В	В
4	З	ЮЗ
5	ЮВ	ЮВ
6	В	С
7	СЗ	Ю
8	С	СЗ
9	СЗ	СЗ
10	СЗ	З
11	С	З
12	ЮЗ	ЮЗ
13	З	З
14	З	СЗ
15	З	З
16	З	З
17	СЗ	ЮВ
18	ЮВ	ЮВ
19	Ю	В
20	ЮВ	С
21	ЮВ	ЮВ
22	В	В
23	В	В
24	В	В
25	СВ	ЮВ
26	В	В
27	В	СВ
28	С	В
29	С	В
30	СВ	В

6 Постройте на отрезке от -2 до $1{,}5$ графики функций $y=\sqrt{x^2+4}-2{,}5$ и $y=\dfrac{2{,}5-\sqrt{x^2+4}}{3}$ (шаг изменения x возьмите равным $0{,}1$). У вас должен получиться рисунок, похожий на тот, который приведён ниже.



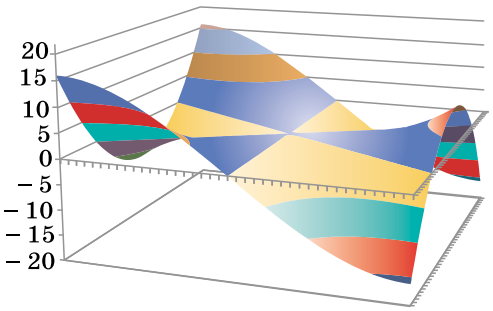
Не правда ли, он напоминает плывущую рыбку? Используя различные функции, вы можете попытаться построить и другие интересные рисунки.

7 Постройте какие-либо графики функций, формулы для которых приведены на с. 60 в задании 4. Пофантазируйте, какие из этих графиков можно было бы использовать для создания тех или иных рисунков. Попытайтесь, к примеру, создать рисунок бабочки.

8 а) Постройте поверхность, заданную формулой $z = y^2 - x^2$. Её изображение приведено на рисунке на с. 45, фрагмент заполнения таблицы приведён справа.

б) Постройте поверхность, заданную формулой $z = y^3 - 3yx^2$. Её изображение приведено на рисунке. Найдите название этой поверхности в Интернете.

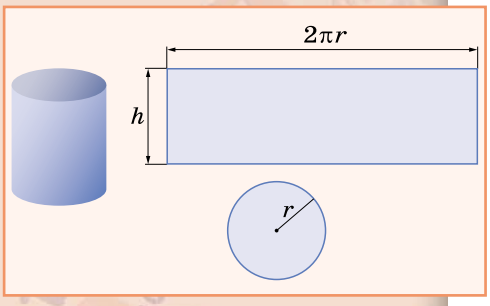
Лабораторная работа завершена.



x \ y	−1	−0,9	−0,8	−0,7
−1	0	−0,19	−0,4	−0,51
−0,9	0,2	0	−0,2	−0,32
−0,8	0,4	0,17	0	−0,15
−0,7	0,5	0,32	0,15	0

ВСПОМНИТЕ:

- Какие задачи называют задачами оптимизации
- В чём состоит метод наименьших квадратов



КОМПЬЮТЕРНЫЙ ПРАКТИКУМ

Лабораторная работа № 8 компьютерного практикума посвящена решению одного из важных классов задач, постоянно возникающих в разных сферах человеческой деятельности. Речь идёт о задачах поиска оптимального решения.

ЛАБОРАТОРНАЯ РАБОТА № 8. ПОИСК ОПТИМАЛЬНОГО РЕШЕНИЯ

Начнём с задачи, решение которой было разобрано на с. 46. Требуется изготовить ёмкость цилиндрической формы без крышки объёмом 10 л. Какими должны быть радиус основания и высота ёмкости, чтобы на её изготовление потребовалось как можно меньше материала?

Напомним, что через h обозначена высота цилиндра, а через r — радиус основания. Тогда при заданном объёме площадь поверхности выражается формулой

$$S = \pi r^2 + \frac{20\,000}{r},$$

и требуется найти положительное значение r, при котором величина S будет наименьшей.

1 А теперь, следуя объяснительному тексту на с. 46, заполните электронную таблицу и постройте график зависимости S от r. Убедитесь, глядя не только на график, но и на числа в столбце B, что минимум для S достигается, когда значение r заключено в пределах от 14 до 16.

Чтобы найти значение радиуса более точно, в столбце A в ячейку A2 введите число 14, а в ячейку A3 — число 14,1 и выполните автозаполнение столбца A до значения 16. При этом у вас автоматически перестроится график. Найдите требуемое значение r с точностью до десятых.

Найдите тем же способом значение радиуса r с точностью до сотых.

Поступая описанным способом, можно найти значение r с как угодно большой точностью. Но в электронных таблицах имеется инструмент, позволяющий сразу находить значения аргументов, при которых функция принимает наибольшее или наименьшее значение. Этот инструмент называется *Поиск решения*. Следуя объяснениям на с. 46, заполните требуемые поля в диалоговом окне *Поиск решения*. Найдите значение r.

Точное значение r, при котором достигается минимум значения S, равно $10\sqrt[3]{\frac{10}{\pi}}$. Вычислите это значение в отдельной ячейке электронной таблицы или на калькуляторе и сравните с тем, которое получилось у вас. Надеемся, что результаты различаются в пределах ошибки округления. Впрочем, вычислить радиус основания цилиндра точно невозможно, всё равно его значение

$$r = 10\sqrt[3]{\frac{10}{\pi}} \text{ см}$$

придётся взять приближённым.

Найдите, какой должна быть высота h этого цилиндра.

2 Три одноклассника собрались пойти в пятидневный поход. Посовещавшись, они составили список того, что нужно взять с собой (не считая личных вещей). Этот список представлен в таблице. Требуется разделить указанные вещи на три как можно более равные по весу части.

Что брать	Вес, кг	Количество
Палатка	7	1
Котелок	0,7	2
Топор	3	2
Консервы	0,4	15
Крупа	0,5	4
Сухари	0,4	5
Сахар	0,5	2
Чай	0,05	6
Сгущёнка	0,45	5

Заполните в электронной таблице столбцы A, B и C в соответствии с таблицей исходных данных. В следующих столбцах — D, E, F — естественно указывать, сколько предметов данного наименования берёт соответствующий турист. В ячейках D1, E1, F1 мы предлагаем написать имена этих туристов, чтобы в дальнейшем было удобно на них ссылаться. Пусть это будут Антон, Борис и Виктор. Заполните каким-либо образом блок ячеек D2:F10, указав тем самым ваше представление о наиболее равномерном распределении груза.

Чтобы увидеть результат такого распределения, в ячейку D11 запишите формулу подсчёта общего веса груза для Антона и скопируйте её в ячейки E11 и F11. (Мы советуем воспользоваться функцией СУММПРОИЗВ(), если вы работаете в Excel, или SUMPRODUCT(), если вы имеете дело с OpenOffice.org Calc.)

А теперь посмотрите, насколько равномерным получилось у вас распределение груза. Думаем, что результат получился не очень хорошим.

Ошибиться может каждый. Не исключены и ошибки при заполнении таблицы. Вполне может оказаться, что вы распределили всего 14 банок консервов вместо 15 или, наоборот, 16 банок, а про один топор забыли. Чтобы избежать этого, можно сразу учесть, что, например, Виктор берёт всё, что ещё не взяли Антон и Борис. Подумайте, какую формулу надо записать в ячейки столбца F, чтобы реализовать этот принцип. Запишите эту формулу. Теперь вы гарантированы от таких ошибок, когда будете варьировать значения в столбцах D и E.

Теперь, изменяя числа в столбцах D и E, постарайтесь подобрать наиболее равномерное распределение вещей. Можете ли вы обосновать, что найденное вами решение оптимальное, т.е. нельзя добиться более равномерного распределения?

Обдумайте и такой вопрос: можно ли было для решения этой задачи применить инструмент *Поиск решения*? Проведите компьютерный эксперимент и попытайтесь объяснить те эффекты, которые вы будете наблюдать.

3 Напомним экономическую задачу, решение которой мы обсуждали на с. 48:

«Какой должна быть цена услуги, чтобы доход был наибольшим?»

Количество потребителей услуги мы обозначили буквой x, а через $f(x)$ обозначили её цену. Тогда доход D, очевидно, будет вычисляться по формуле

$$D = xf(x).$$

Наша задача разбилась на две части: подобрать формулу для $f(x)$ и найти значение x, при котором функция D принимает наибольшее значение.

Саму функцию $f(x)$ предлагается искать в виде

$$f(x) = 100 - ax^2,$$

коэффициент a подбирать, используя экспериментальные данные. Эти данные приведены в таблице.

Цена, р.	50	80	110	140	170	200	230	260	290
Количество потребителей услуги, чел.	98	94	88	80	71	60	47	32	16

Заполнение электронной таблицы тогда можно предложить такое, как в таблице ниже (в ячейке L3 вычисляется некоторое значение параметра a; фактически мы положили его равным 0,01, но вы можете выбрать и другое значение, которое вам кажется более подходящим).

	A	B	C	D	E	F	G	H	I	J	K	L
1	Цена, xi	0	50	80	110	140	170	200	230	260	290	0
2	Количество потребителей, yi	100	98	94	88	80	71	60	47	32	16	0,01
3	f(xi)	=100–$L3*B$1^2										=L1+$L2
4	(yi — f(xi))2	=(B$2–B3)^2										

Скопируйте формулы из ячеек B3 и B4 в ячейки столбцов C—K, а в ячейку L4 запишите сумму блока B4:K4. Наша цель — минимизировать значение в ячейке L4, изменяя значение в ячейке L3.

Результат, который вы наблюдаете в ячейке L4, вас, наверно, огорчил. Да и взгляд на строку B3—K3 тоже удручает: вычисляемое количество потребителей становится отрицательным. Ясно, что значение коэффициента a надо уменьшать.

И вместо того чтобы огорчаться из-за постигшей вас неудачи, воспользуемся полученным результатом: в ячейку L1 запишем число 0,01, а в ячейку L2 отрицательное число –0,01. Улучшился ли после этого результат? Немного улучшился.

Скопируйте теперь блок ячеек B3:L4, начиная с ячейки B5. Содержимое ячейки L6 свидетельствует о дальнейшем улучшении результата. Продолжайте копировать, пока значение параметра a (в соответствующей ячейке столбца L с нечётным номером) не станет равным 0.

Рассматривая теперь значения, которые записаны в ячейках столбца L чётными номерами, вы обнаружите, что наименьшее значение записано в ячейке L20, а соответствующее значение параметра a равно 0,001. Если вы считаете, что найденная точность параметра a недостаточна, продолжите компьютерный эксперимент, записав в ячейку L1 число, чуть большее 0,001 (например, 0,0011), а в ячейку L2 — число –0,00001. Мы же ограничимся найденным значением для коэффициента a.

Приступим теперь к определению того значения цены, при котором доход окажется максимальным. Эта часть задачи похожа на задачу про цилиндрическую ёмкость, только искать надо будет не минимальное значение, а максимальное.

Перейдите на новый лист электронной таблицы. Заполните первые его две строки так, как показано в таблице.

	A	B	C
1	Цена	Количество посетителей	Доход
2	10	=100–0,001*A2^2	=A2*B2
3			

Примите решение, на какую величину вы будете увеличивать цену билетов, чтобы исследовать изменение числа посетителей и величины дохода. С помощью автозаполнения проставьте цены в столбце A и скопируйте формулы из ячеек B2 и C2 в ячейки других строк этих же столбцов. Постройте графики изменения количества посетителей и дохода.

На графике дохода отчётливо видна область максимума. Она располагается в ценовом диапазоне от 170 до 190 рублей. Уточните значение оптимальной цены, уменьшая шаг изменения цены. Определите оптимальное значение цены с точностью до рубля.

Для поиска максимума функции можно использовать режим *Поиск решения*. Если этот режим вам доступен, найдите с его помощью оптимальную цену. Окажется, что она равна 182 р. 57 к. А сколько тогда ожидается зрителей? Это число указано в столбце B — примерно 66—67 человек.

Возникает естественный вопрос: может быть, можно использовать меньший зал? Тогда и за аренду придётся платить меньше. Замените число 100 в формуле ячейки B2 на 70, не забудьте скопировать новую формулу в другие ячейки этого столбца и посмотрите, что получится.

Оптимальная цена билета уменьшилась почти на 20 рублей, доход упал более чем на 40%, а прогнозируемое количество зрителей теперь 46—47 человек. Как и для 100 зрителей, это количество составляет примерно 2/3 от числа мест в зале.

Может быть, это зависит от коэффициента a? Поменяйте значение коэффициента и снова проведите компьютерный эксперимент, когда количество мест 100 и 70. Цены получились другие, доход другой, а число посетителей при оптимальной цене осталось практически тем же.

Вооружившись математическими методами исследования функций, можно доказать, что максимальная выручка всегда будет получаться при заполнении зала на 2/3. Впрочем, проведённые расчёты годятся не только для кинотеатров, но и для самолётов, и для автотранспорта. Так что, господа будущие бизнесмены, запомните: максимальный доход получается тогда, когда предложение в полтора раза превышает спрос. Поэтому у успешного бизнесмена вы не увидите переполненный кинозал или забитый до отказа самолёт, в любой гостинице найдутся свободные номера, а в кафе — свободный столик.

Обдумайте и такой вопрос: какой смысл имеет коэффициент a с экономической точки зрения? Проведите вычисления, меняя значения коэффициента a и наблюдая за изменением оптимальной цены билетов.

Вы обнаружили, что с уменьшением этого коэффициента оптимальная цена возрастает. В жизни если человек готов платить за услугу или товар больше, то коэффициент a будет уменьшаться. Иными словами, он отражает уровень покупательной способности населения.

Лабораторная работа завершена.

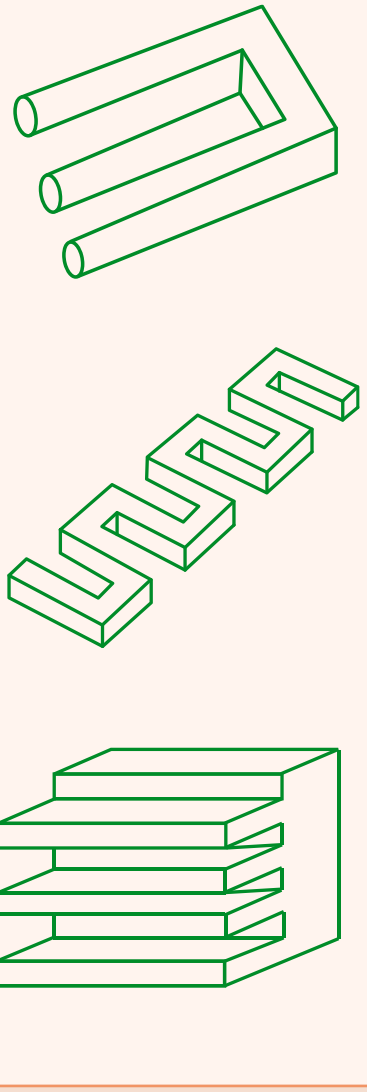
КОМПЬЮТЕРНЫЙ ПРАКТИКУМ

ВЫ УЗНАЕТЕ:

- Как выполняются основные операции в графических редакторах
- Какие эффекты могут быть реализованы в графическом редакторе

ВСПОМНИТЕ:

- Что такое примитивы графического редактора
- Каковы основные примитивы, используемые в графических редакторах



Мы не знаем, какой именно графический редактор имеется в вашем распоряжении. Возможно, это один из самых простых растровых редакторов Paint, входящий в число стандартных приложений ОС Windows, или более мощный CorelDRAW, также работающий под ОС Windows, или векторный редактор OpenOffice.org Draw, который можно использовать как под Windows, так и под Linux. Тем не менее графические инструменты и их использование в разных редакторах весьма схожи (их общее описание приведено на с. 51), так что наши задания формулируются одинаково, а в случае возникновения затруднений с конкретной реализацией пользуйтесь справочной системой, инструкцией и указаниями учителя.

Соблюдайте правила хорошего тона при работе с графическим редактором.

- Если вы случайно попали куда-нибудь не туда, позовите учителя, и он поможет вам выбраться из сложной ситуации. Не пытайтесь щёлкать клавишей мыши по всем подряд пиктограммам или панически нажимать клавиши на клавиатуре (тем более сразу по нескольку одновременно).
- Не выбирайте инструменты *сканер* и *принтер*, не получив предварительно разрешения учителя. Возможно, к вашему компьютеру не подключено ни то ни другое. Это может привести к зависанию машины, т.е. отказу компьютера работать с вами дальше.
- Не записывайте ничего на диск без разрешения учителя.
- Не заливайте всё поле яркой краской — не будет виден курсор!

ЛАБОРАТОРНАЯ РАБОТА № 9. ОСНОВНЫЕ ИНСТРУМЕНТЫ ГРАФИЧЕСКОГО РЕДАКТОРА В этой лабораторной работе вы познакомитесь с основными инструментами графического редактора (или вспомните их, если имели дело с графическим редактором раньше). Как правило, они изображены на пиктограммах и для их активации достаточно щёлкнуть по нужной пиктограмме.

А теперь выполните несколько заданий:

1 Перемещая курсор, попробуйте выяснить, где находится начало системы координат и куда направлены оси.

2 Изобразите домик. Используйте для этого инструменты *прямоугольник* и *отрезок*.

3 Пусть у домика будет труба, из которой идёт дым, а рядом растёт замечательное дерево (ель или тополь). Используйте инструменты *распылитель* и *овал*.

4 Раскрасьте дом и трубу, используя инструмент *узор* (для крыши — *черепицу*, для дома и трубы — *кирпичи*).

5 А теперь попробуйте нарисовать «мистический камертон», который изображён на верхнем рисунке.

6 Ниже представлены ещё три невозможные фигуры. Попробуйте нарисовать их.

Эти фигуры, совсем простые с точки зрения рисования, представили художникам основу для создания картин, заставляющих зрителей вглядываться в них и удивляться, а

иногда и задумываться. Взгляните на рисунок голландского художника М.К. Эшера, благодаря литографиям которого невозможные фигуры получили широкую известность.

7 Попытайтесь придумать ещё какие-нибудь невозможные фигуры и попробуйте нарисовать их. Организуйте среди одноклассников конкурс таких рисунков.

8 Вернитесь к рисунку дома с деревом. Постройте сказочный город, в котором все дома как будто похожи и в то же время не похожи друг на друга, а вернее, на ваш исходный дом. Для этого поместите изображение дома в буфер обмена. (Как это сделать конкретно в вашем графическом редакторе, вам расскажет учитель или вы можете прочитать самостоятельно, воспользовавшись встроенной справкой или инструкцией пользователя.) Затем вставьте его в выбранное вами место на рисунке, изменяя, если хотите, его размеры. Если дом сделать высоким и узким, то он станет похожим на башню. Если растянуть вширь, он скорее будет похож на амбар. Экспериментируйте! Украсьте ваш город деревьями, изменяя их размер и пользуясь правилами перспективы.

9 В вашем сказочном городе пошёл сказочный снег... Но у вас нет такой «снежной» заливки. Это новый *узор*. Как его создать, узнайте из инструкции пользователя. Как только «снег» готов, выберите инструмент *заливка*, подберите нужный цвет и устройте в городе снегопад. Можно выбрать, например, сиреневый цвет, тогда снег будет совсем необычным, а ваш рисунок, возможно, станет похожим на замечательную картину французского художника Клода Моне, передающую ощущение тумана.

10 Придумайте название вашему рисунку и напишите его в правом верхнем углу. Примените к буквам различные спецэффекты, которые допускает графический редактор. Изумительно смотрятся буквы с *расслоённым контуром*, особенно если потом внутренний контур закрасить другим цветом. Заливка может быть *градиентной*, т.е. постепенно меняющей свой оттенок. Не забудьте, что текст можно *наклонять, поворачивать*...

Лабораторная работа завершена.

К. Моне. «Мост Ватерлоо. Эффект тумана»

М.К. Эшер. «Бельведер» (Использована иллюстрация из книги: *Эшер М.К.* Графика. — М.: Taschen/Арт-родник, 2001. — Илл. 74.)

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ

ВЫ УЗНАЕТЕ:

- Как настраивать палитру цветов
- Как создавать цветовые эффекты
- Как использовать слои для создания изображений сложной структуры
- Как ретушировать фотографии

ВСПОМНИТЕ:

- Как в графическом редакторе обычно выполняются операции вставки и копирования изображений

Одним из популярных графических редакторов, позволяющих создавать многослойное изображение, является Adobe Photoshop. О его основных инструментах мы рассказали на с. 54–58.

ЛАБОРАТОРНАЯ РАБОТА № 10. ОСНОВНЫЕ ИНСТРУМЕНТЫ РЕДАКТОРА ADOBE PHOTOSHOP Взаимодействие любого программного продукта с пользователем определяется, как вы знаете, интерфейсом этого продукта. Поэтому освоение Adobe Photoshop мы предлагаем вам начать со знакомства с его интерфейсом, с тем, как осуществляется доступ к функциям программы посредством выбора команд в меню или нажатием комбинаций клавиш на клавиатуре.

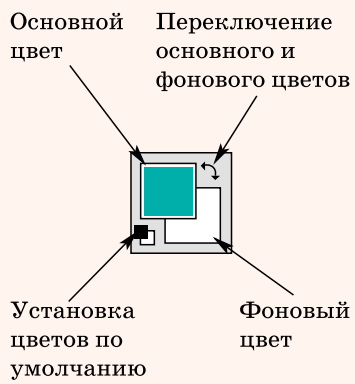
1 Запустите Adobe Photoshop.

Окно программы имеет типовое оформление.

Панель инструментов состоит из кнопок. Щелчок по кнопке включает соответствующий ей инструмент. Если на кнопке, имеющей внизу маленький треугольник, подольше подержать мышь с нажатой клавишей, то появится всплывающая панель с кнопками дополнительных родственных инструментов.

Панель свойств служит для настройки параметров используемого в данный момент инструмента.

На панели инструментов находятся образцы **основного** и **фонового** цветов (см. рисунок). По умолчанию это чёрный и белый цвета. Образец основного цвета на верхнем квадрате. Этим цветом закрашивают точки изображения команды заливки и инструменты рисования. Фоновым цветом закрашиваются точки после удаления фрагментов изображения. Двунаправленная изогнутая стрелка в верхнем правом углу около этих квадратов позволяет поменять образцы местами. В любой момент вы можете произвольно менять основной и фоновый цвета. Возврат к цветам по умолчанию — щелчок по кнопке слева в виде двух маленьких квадратиков чёрного и белого цвета.



На рабочем поле редактора (обычно в правой части) располагаются **палитры**. Палитры — это диалоговые окна, предназначенные для задания параметров и выбора режимов работы. В одном диалоговом окне может быть несколько вкладок — отдельных палитр. Переход к нужной вкладке осуществляется щелчком по ярлыку с названием. Вызов палитры или удаление с рабочего поля осуществляется командами из меню *Окно*.

Кнопки режимов окна программы в нижней части панели инструментов работают, если имеется открытый документ. Левая предназначена для возврата в стандартный вид окна программы. Две другие открывают окно программы на весь экран, эти режимы работы удобны для просмотра больших рисунков или рисунков в увеличенном масштабе.

Для создания рисунка на чистом листе выберите команду через меню *Файл*, *Новый*. В открывшемся диалоговом окне задайте размеры листа, разрешение, цвет фона. Чтобы для редактирования загрузить из файла уже существующий

рисунок, надо воспользоваться меню *Файл*, *Открыть*. Получение оригинала для редактирования со сканера или цифровой фотокамеры осуществляется командой меню *Файл*, *Импорт*.

В программе реализовано несколько способов выбора текущих цветов.

1-й способ. В палитре *Каталог* выбор основного цвета осуществляется щелчком левой клавиши по выбранному цвету, а выбор фонового цвета — при нажатой клавиши Alt на клавиатуре щелчком левой клавишей мыши.

2-й способ. В палитре *Цвет* сначала выберите редактируемый цвет (верхняя кнопка означает основной цвет, нижняя — фоновый цвет). Кнопка редактируемого цвета будет очерчена двойной рамкой. Грубый выбор активного цвета осуществляется щелчком левой клавиши в нижней цветовой полосе; точная настройка цвета производится затем с помощью движков.

3-й способ. Выбор цвета на рисунке выполняется непосредственно на изображении при помощи инструмента *Пипетка* . Щелчок левой клавишей по выбранному цвету в изображении меняет основной цвет на заданный, а щелчок левой клавишей при нажатой клавише Alt меняет цвет фона.

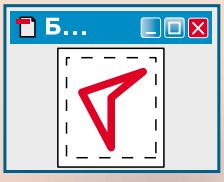
4-й способ. Щелчок по кнопке основного или фонового цвета на панели инструментов выводит на экран палитру соответственно для выбора основного или фонового цвета.

2 Опробуйте различные инструменты для рисования. Для этого создайте какой-либо несложный рисунок. Восемь пиктограмм рисующих инструментов расположены единой группой

Любой выбранный инструмент перед использованием необходимо настроить с помощью кнопок и раскрывающихся списков в *Панели свойств*, располагающихся в верхней части экрана. Например, вы можете выбирать размер и форму кисти в раскрывающемся списке *Кисть* (см. рисунок), в нём справа имеется кнопка вызова меню для настройки этой панели и формы кистей. При выборе режима *Обычный* (в других версиях — *Норма*) след от инструмента при рисовании будет сплошным, а при выборе режима *Растворение* — след в виде разбрызганных капель краски. Если включить переключатель *Мокрые края*, то при рисовании краска будет растекаться к границам мазка, имитируя эффект акварели. *Непрозрачность* 100% гарантирует при рисовании полную замену исходного цвета на основной или фоновый. Это замещение может быть частичным, если выбрать непрозрачность меньше 100 %. В раскрывающемся списке *Режим* вы можете выбирать другие способы взаимодействия имеющегося цвета точек и цвета, накладываемого инструментом.

Поэкспериментируйте с различными режимами наложения цветов. При затруднениях поищите ответы в справочной системе программы.

3 Как было сказано выше, форму кисти можно выбирать из некоторого списка, который раскрывается, если щёлкнуть по кнопке вызова меню в панели *Форма кисти*. Но можно создать собственную кисть. Вот как это делается.



Первый шаг. Нарисуйте кистью или карандашом простую фигуру.

Второй шаг. Создайте вокруг неё прямоугольное выделение (см. рисунок).

Третий шаг. Выполните команду меню *Редактирование*, *Определить кисть*. В открывшемся диалоговом окне можно задать произвольное имя кисти и оценить создаваемый элемент слева в поле просмотра. Число в поле просмотра означает размер кисти.

Созданная кисть выбрана. Порисуйте созданной кистью.

Можете сохранить созданную кисть в составе существующего комплекта кистей или создать новый комплект из ваших собственных кистей, открыв список кистей и выбрав в меню команду *Сохранить кисти*.

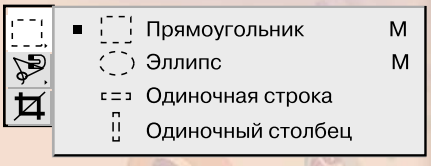
Восстановите стандартный набор кистей командой *Сбросить кисти*.

4 Попробуем создать что-нибудь полезное.

Начнём с чистого листа. Обратившись к меню *Файл*, *Новый*, задайте его характеристики: *Размер*: 10 см × 10 см, *Разрешение*: 28 пикс/см (или, что то же самое, 72 пикс/дюйм), *Цветовой режим*: RGB цвет, *Содержимое фона*: Белый.

Выберите основной цвет зелёный.

А теперь поговорим об инструментах выделения, которыми вы будете дальше пользоваться неоднократно. В Adobe Photoshop имеется целый набор инструментов и функций, которые позволяют не только выделять фрагменты, но и выполнять определённые действия над ними, рисовать геометрические фигуры.



При удержании мыши на верхней левой кнопке в панели инструментов открывается панель с простейшими инструментами выделения. Однократный щелчок по ней выбирает (активизирует) инструмент.

Если выбрать инструмент выделения *Прямоугольник* и задать на панели *Параметры* для параметра *Стиль* значение *Нормальный*, а параметру *Растушёвка* значение 0 px, то можно при нажатой клавише мыши выделить фрагмент произвольного размера, протягивая мышь по диагонали выделяемой области. Созданную область можно затем перемещать по листу, установив курсор мыши внутри этой области и перемещая его при нажатой левой клавише мыши. Сказанное справедливо, если в панели *Параметры* включён режим *Новая выделенная область*.

Значение параметра *Растушёвка* означает чёткость граничных линий, отсутствие размытия границ. При увеличении значения этого параметра границы прямоугольной области будут скруглены, а при её закрашивании края будут размыты. Ненулевое значение параметра *Растушёвка* позволяет смягчить границы выделенной области с тем, чтобы сделать менее резким и заметным переход между изображением в выделенной области и окружающим рисунком. Особенно это важно при вклеивании изображения из другого рисунка.

Если для параметра *Стиль* выбрать значение *Фиксированный размер*, то необходимо указать ширину и высоту инструмента выделения в пикселях. Каждый раз

область выделения указанного размера будет строиться автоматически.

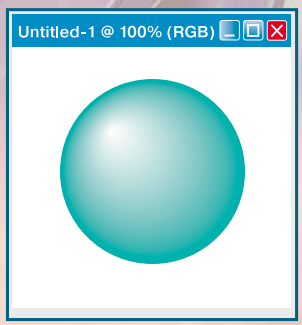
Задайте основной цвет зелёный, а фоновый белый.

Выберите инструмент выделения *Овальная область* и задайте для него следующие значения параметров: *Растушёвка:* 0 px; *Стиль:* Нормальный; для параметра *Сглаживать* поставьте галочку.

Нарисуйте область выделения в виде окружности, для этого во время рисования нужно держать нажатой клавишу Shift.

Выберите инструмент *Градиент*. Задайте для него следующие параметры: *Вид градиента*: Радиальный градиент; *Режим*: Нормальный; *Непрозрачность*: 100%, для параметра *Инверсия* поставьте галочку.

Проведите курсором мыши изнутри круга к его границе с нажатой левой клавишей мыши. У вас должен получиться зелёный шарик (см. рисунок).



Если для параметра *Инверсия* галочка не поставлена, то нужно в качестве основного цвета выбрать белый, а в качестве фонового — зелёный.

Уменьшите размер рисунка, используя команду меню *Изображение / Размер изображения...* В открывшемся диалоговом окне *Размер изображения* поставьте ширину и высоту 1 см.

Получился хороший элемент для оформления, например, списка или кнопки. Сохраните его.

Лабораторная работа завершена.

ЛАБОРАТОРНАЯ РАБОТА № 11. РАБОТА СО СЛОЯМИ Для выполнения действий со слоями имеется специальная палитра, вызов которой осуществляется через меню: *Окно / Слои*. В палитре имеется список слоёв с их уменьшенными изображениями. Справа от изображения — название слоя. Редактировать можно только один активный слой. Его строка в палитре подсвечена синим цветом. Для активизации другого слоя следует щёлкнуть по строке с его названием в палитре.

Вы будете редактировать уже готовые изображения. Поэтому надо иметь два-три файла с рисунками или фотографиями. Начнём, как всегда, с простого задания — создадим вокруг имеющегося рисунка романтический фон.

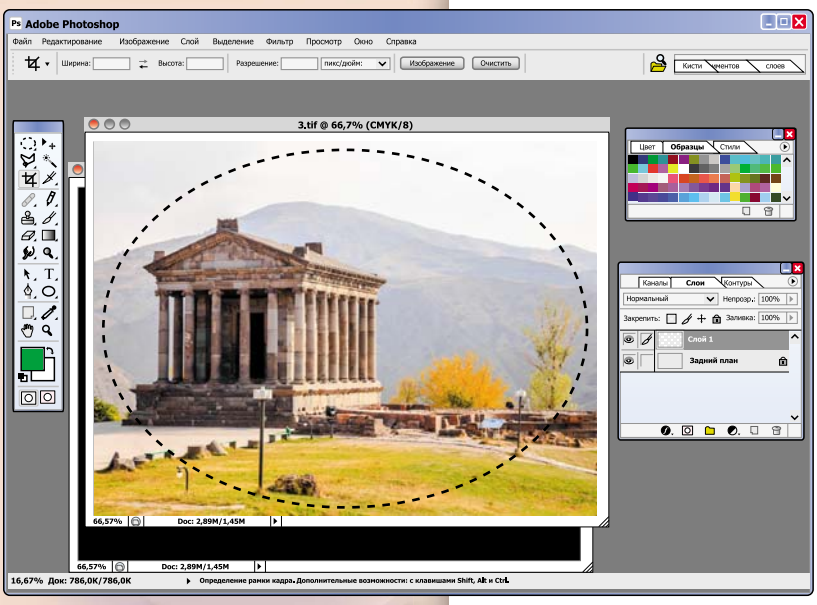
1 Откройте файл с каким-либо рисунком или фотографией.

Выберите инструмент выделения *Овальная область* и задайте для него следующие значения параметров: *Растушёвка*: 15 px; *Стиль*: Нормальный; для параметра *Сглаживать* поставьте галочку.

Постройте эллиптическую область выделения на фотографии, перемещая мышь при нажатой левой клавише по диагонали воображаемого описанного прямоугольника (см. рисунок на с. 84).

Скопируйте выделение в буфер через меню *Редактирование / Копировать*.

2 Создайте новый рисунок размером не менее размера исходной фотографии. Истинный размер фотографии можно посмотреть, используя меню *Изображения / Размер хол-*



ста..., при создании задайте следующие характеристики: *Разрешение*: 72 пик/дюйм, *Цветовой режим*: RGB цвет, *Содержимое фона*: Белый.

Выберите основной цвет (например, голубой).

Выберите инструмент *Заливка* и закрасьте лист.

Вызовите на экран палитру слоёв: *Окно / Показать слои*.

Выполните команду *Редактирование / Вклеить*.

Получилось изображение сложной структуры (см. рисунок). Объедините все его элементы на одном листе — одном слое командой меню *Слой / Объединить видимые*.

Сохраните рисунок, используя команду меню *Файл / Сохранить как...*, в открывшемся диалоговом окне задайте *Имя файла*. В поле *Тип файлов* выберите формат сохранения jpeg (*.jpg, *.jpe).

В открывшемся диалоговом окне *Параметры JPEG* задайте характеристики сохранения: *Качество*: 5 (среднее); *Разновидность формата*: Baseline Optimized (Оптимизированный).

Определите размер получившегося файла.

3 А теперь, используя механизм слоёв, попытайтесь сделать так, чтобы получившийся у вас рисунок воспринимался как вид из окна. Для этого, как вы понимаете, именно контуры окна должны быть созданы в отдельном слое.

Лабораторная работа завершена.

ЛАБОРАТОРНАЯ РАБОТА № 12. РЕДАКТИРОВАНИЕ ФОТОГРАФИЙ

Поработаем теперь над улучшением цифровой фотографии.

1 Загрузите файл с какой-либо фотографией.

Используя диалоговое окно, которое вызывается в меню *Изображение / Гистограмма*, понаблюдайте за распределением тонов.

Измените яркость рисунка командой *Изображение / Коррекция / Уровни*. Посмотрите, как изменилась ваша фотография. Снова посмотрите распределение тонов на гистограмме.

Давайте ещё улучшим фотографию, применив средства ретуширования. Для этого можно использовать несколько способов. Вот два из них.

Ретушью называют исправление дефектов изображения.

1-й способ. Выберите инструмент *Палец* , настройте его параметры: *Кисть*: 13 px с размытыми краями, *Режим*: Нормальный, *Интенсивность*: 50%. Перемещением мыши по отдельным участкам определённого цвета уберите чрезмерную резкость рисунка.

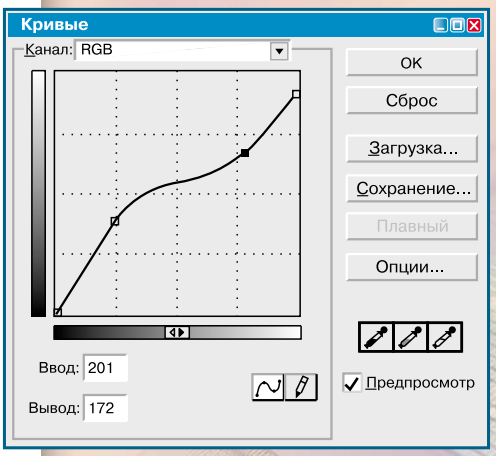
2-й способ. Выберите инструмент *Многоугольное лассо* .

Перемещая мышь вдоль граничной линии и щёлкая левой клавишей в точках изгиба, получите выделение области неба. Примените к выделенной области фильтр (меню *Фильтр / Резкость*).

Отретушируйте фотографию.

2 Возможно, вас всё ещё не устраивает качество созданного изображения. В этом случае можно воспользоваться гамма-коррекцией. Она служит для настройки тонального изображения на отдельных участках диапазона, а не в целом. Гамма-коррекцию осуществляют в диалоговом окне, вызываемом командой *Изображение / Коррекция / Кривые...*

В появившемся диалоговом окне *Кривые* изображён график, на котором по горизонтальной оси отображаются входные значения яркости, а по вертикальной — выходные. Первоначально это диагональная прямая, отображающая линейную зависимость между входными и выходными значениями яркостей. Коррекция осуществляется установкой нелинейной зависимости, в результате которой некоторые участки входного диапазона яркостей сужаются в выходном диапазоне и за счёт этого происходит расширение других участков. Проблема состоит в определении этих участков.



Чтобы определить расположение участка, который следует корректировать, нужно при открытом окне *Кривые* установить курсор мыши на исправляемый участок, перемещать по нему мышь при нажатой левой клавише. При этом на графике по линии будет перемещаться кружочек, показывающий тональный диапазон этого участка рисунка.

Щелчками на линии устанавливаются предельные точки изменяемого диапазона. Можно зафиксировать до 15 точек. Лишние точки удаляются перемещением их по линии за пределы графика. Для корректировки диапазона нужно щёлкнуть в центре диапазона на линии, зацепиться за точку и при нажатой левой клавише переместить её в нужном направлении до достижения приемлемого результата. У вас может получиться, например, такая кривая, какая показана на рисунке.

Выделите редактируемый участок фотографии при помощи любого инструмента выделения. Настройте тональность изображения, используя гамма-коррекцию.

Сохраните отредактированную фотографию в формате jpeg.

Если у вас осталось время, задайте в режиме гамма-коррекции какой-либо экзотический график, например с несколькими разрывами. Посмотрите, как это отразилось на выделенном фрагменте изображения. Такие парадоксальные соединения красок тем не менее могут вполне использоваться для создания декоративных полотен.

Лабораторная работа завершена.

ПОДВЕДЁМ ИТОГИ

- **Компьютерные информационные технологии** — средства создания и преобразования информационных объектов с помощью компьютера.
- **Редактирование текста** — обработка с целью сделать его более доступным для понимания и удобным для восприятия.
- **Текстовый редактор** — программа, предназначенная для создания, форматирования и печати текстовых документов.
- **Гипертекст** — текст, в котором организованы гиперссылки на фрагменты или объекты того же текста или другого документа.
- **Электронная таблица** — программа, с помощью которой можно производить вычисления с данными, представленными в табличной форме.
- Изображение на экране создаётся как **растр**, состоящий из большого числа отдельных элементов — **пикселей**. Способ, при котором задаются цвет и яркость для каждого отдельно взятого пикселя, называется **растровым**; **векторным** называется способ задания рисунка с помощью формул, описывающих объекты, из которых рисунок составлен.
- Практически любой графический редактор — и векторный, и растровый — позволяет рисовать с помощью графических примитивов, а также изменять характеристики любого пикселя изображения.
- Для представления результатов работы над проектом используют **электронные презентации**. Они позволяют передать слушателем информацию, представленную в разнообразной форме, т.е. являются мультимедийным продуктом.

ПОДРОБНЕЕ...

Васильев А. В., Богомолова О. Б. Работа в электронных таблицах: Практикум. — М.: Бином, 2007. — 160 с.

Гейн А. Г. Информатика и ИКТ. Задачник-практикум: 10—11 классы. — М.: Просвещение, 2019. — 157 с.

Крылов С. С. Готовимся к ЕГЭ. Информационные и телекоммуникационные технологии. — М.: Интеллект-центр, 2010. — 48 с.

Лещинер В.Р., Крылов С. С., Ушаков Д. М. Я сдам ЕГЭ! Информатика. Типовые задания. — М.: Просвещение, 2019. — 240 с.

Энциклопедия школьной информатики. — М.: БИНОМ: Лаборатория знаний, 2014. — 400 с.

ВОПРОСЫ ДЛЯ ОБСУЖДЕНИЯ:

- К XX в. изготовление документов на бумаге полностью вытеснило их изготовление на других носителях – камне, обожжённой глине, шёлке и др. Заменят ли в будущем электронные документы полностью документы на бумаге?
- В чём вы видите преимущества и в чём недостатки представления числовой информации в виде диаграмм по сравнению с табличной формой?
- Какова роль устного выступления, если результаты вашей работы оформлены в виде презентации?

Федеральный центр информационно-образовательных ресурсов (ФЦИОР): http://fcior.edu.ru/

Глава 3

АЛГОРИТМЫ И ИСПОЛНИТЕЛИ. ОСНОВЫ ПРОГРАММИРОВАНИЯ

САМОЕ... САМОЕ...

Компьютер может написать текст, провести расчёты и даже создать рисунок. Но остаётся тайна: как он справляется со всеми поручениями, которые ему даются? Как создаются, на каких принципах основаны те чудесные программы, которые управляют его работой?

Скажем сразу: эти принципы относятся не только к компьютерам, но и к любой микропроцессорной технике, к любым роботам, какими бы сложными и человекоподобными они ни казались. В основе управления всеми такими устройствами лежит важное понятие — «алгоритм».

19

ФОРМАЛЬНЫЕ ИСПОЛНИТЕЛИ

ВЫ УЗНАЕТЕ:

- Что такое формальный исполнитель
- Чем характеризуется формальный исполнитель

ВСПОМНИТЕ:

- Какой язык называется формализованным
- Какой язык называется формальным.
- Что такое синтаксис и семантика языка

Обсуждая процесс обработки информации, мы акцентировали внимание на том, что для такой обработки нужен исполнитель — объект или субъект, осуществляющий преобразование информации.

ОБРАБОТКА ИНФОРМАЦИИ Обработкой информации следует признать и вычисление суммы двух чисел — ведь из двух известных данных получается новое, до того неизвестное. Обработкой информации является и, например, перевод предложения с русского языка на иностранный.

Вот несколько примеров обработки информации. Детектив собрал улики и указал, кто совершил преступление. Математик сопоставил известные ему утверждения и доказал новую теорему. Д. И. Менделеев на основе информации о химических свойствах элементов сформулировал периодический закон. Во всех этих примерах в результате обработки информации появилась новая информация.

На первый взгляд между процессами обработки информации, указанными в двух предыдущих абзацах, большая разница. Главное отличие здесь в том, что для розыска преступника или доказательства новой теоремы нет и не может существовать жёстких правил, как должна обрабатываться исходная информация. Как говорят, человек в этих случаях действует **эвристически**.

Складывая два числа, мы уже руководствуемся жёстко указанными правилами. Такую работу можно поручить техническому устройству, которое способно понимать и исполнять предписанную ему инструкцию. Устройства, управляемые с помощью инструкций и выполняющие свою работу автоматически, называют программируемыми и говорят, что свою работу они исполняют **формально**. В частности, можно говорить и о формальной обработке информации. Исполнитель, производящий такую обработку, не должен вникать в смысл выполняемых им действий; поэтому формальная обработка информации, как правило, касается изменения формы её представления, а не содержания.

ФОРМАЛЬНЫЕ ИСПОЛНИТЕЛИ Вспомнив предыдущие уроки информатики, вы наверняка поймёте, что все те средства компьютерных технологий, которые вами изучались, являются не чем иным, как формальными исполнителями, готовыми по одному движению вашего пальца обрабатывать электронные документы. И если электронная таблица почему-то отказывается пересчитывать данные, то объясняется это либо неисправностью оборудования, либо (что бывает гораздо чаще) вашими собственными ошибками.

Разнообразных устройств, которыми человек управляет как формальными исполнителями, становится всё больше и больше. К ним относятся, в частности, **роботы** — автоматические устройства, которые частично или полностью заменяют человека при выполнении работ в опасных для жизни условиях, при относительной недоступности объекта, а также во многих других случаях.

Под обработкой информации понимают получение новой иформации из уже имеющейся или преобразование формы представления информации.

Слово «робот» придумано чешским писателем Карелом Чапеком и долго использовалось только писателями фантастами. Тем не менее, первыми прообразами роботов можно считать, например, механические фигуры, созданные арабским учёным и изобретателем Аль-Джазари (1136–1206). Так, он создал лодку с четырьмя механическими музыкантами, которые играли на бубнах, арфе и флейте.

Всерьёз созданием роботов начали заниматься в 50-х годах прошлого века.

В Технологическом университете города Харбин в Китае в июне 2010 года прошли первые в истории Олимпийские игры среди человекоподобных роботов.

СВОЙСТВА И ХАРАКТЕРИСТИКИ ФОРМАЛЬНЫХ ИСПОЛНИТЕЛЕЙ

Для того чтобы формальный исполнитель мог выполнить ту или иную работу, ему нужна инструкция. В инструкции указывается, какие действия и в каком порядке требуется исполнить, чтобы преобразовать исходные данные в результат. Мы уже обсуждали, что формальный исполнитель не вникает в смысл исполняемых им действий, не задаётся вопросами, зачем он это делает и нельзя ли сделать лучше. Поэтому каждое действие должно быть «преподнесено» исполнителю так, чтобы он понял его однозначно и мог исполнить. Эти условия означают, что в инструкции должны указываться только допустимые действия данного исполнителя. Сама инструкция пишется на формализованном языке.

В некоторых случаях требуется, чтобы не только смысл каждого слова был определён однозначно, но и сами слова должны строиться по строгим формальным правилам, т.е. грамматика языка тоже должна быть формализована. Такой язык, как вы, наверно, помните, называют формальным.

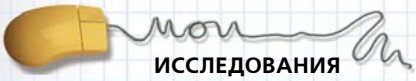
К формальным языкам относятся языки программирования и, вообще, все языки управления автоматическими устройствами. Но и человек в своей работе нередко использует формальные языки, например язык записи натуральных чисел в десятичной системе счисления.

Совокупность всех команд, которые понимает конкретный исполнитель, называется **системой команд** этого исполнителя. А совокупность всех действий, которые он может выполнить в ответ на эти команды, называется **системой допустимых действий** исполнителя.

Когда мы выбираем исполнителя, то надеемся с его помощью решить нужную нам задачу. Но тогда сразу встаёт вопрос, может ли исполнитель с помощью своих допустимых действий получить требуемый результат? Совокупность тех результатов, которые можно получить с помощью данного исполнителя, называется его **достижимыми целями**.

Если в вашем распоряжении оказалось два разных исполнителя, у которых множество достижимых целей одинаково, то для решения задачи можно взять любого из них. Таких исполнителей естественно назвать **равносильными** или, по-другому, **эквивалентными**. Из двух эквивалентных исполнителей можно выбрать того, который лучше по каким-то другим характеристикам — проще в изготовлении, экономичнее, легче в управлении и т.п. Например, один исполнитель понимает команды только на английском языке, а другой — на русском, хотя набор допустимых действий у них одинаков. Какого исполнителя вы предпочтёте? Впрочем, может оказаться, что и наборы допустимых действий у них различны, а достижимые цели одинаковы. Такие исполнители всё равно эквивалентны — нам ведь в первую очередь важно то, можно ли с помощью данного исполнителя решить нужную задачу.

В дальнейшем, говоря «исполнитель», мы всегда будем иметь в виду формального исполнителя. Поэтому для краткости слово «формальный» мы, как правило, будем опускать.



ИССЛЕДОВАНИЯ

Подсчитайте, сколько раз в течение дня (или недели) вы ведёте себя как формальный исполнитель, т.е. исполняете какие-либо инструкции, не анализируя их.

ВОПРОСЫ:

- Чем формальный исполнитель отличается от других исполнителей?
- Что такое система допустимых действий данного исполнителя?
- Что такое система команд исполнителя?
- Что называется достижимыми целями исполнителя?
- Может ли существовать Всемогущий Исполнитель, для которого любая цель достижима?

20

АЛГОРИТМЫ И ПРОГРАММЫ

ВЫ УЗНАЕТЕ:

- Что понимают под терминами «алгоритм» и «программа»
- Чем программа отличается от алгоритма
- Какими свойствами обычно обладает алгоритм
- Какие ошибки называют синтаксическими, а какие семантическими
- Что такое отладка программы

ВСПОМНИТЕ:

- Чем руководствуется формальный исполнитель
- Что должно быть указано в инструкции к формальному исполнителю

Алгоритм — последовательность допустимых действий исполнителя, направленная на достижение заданной цели.
Программа — алгоритм, записанный на формальном языке исполнителя.

Происхождение слова «алгоритм» обычно связывают с именем арабского учёного Мухаммада ибн Мусы аль-Хорезми, что в переводе на современный русский означает «Мухаммад Мусаевич из Хорезма». Латинизированный вариант написания аль-Хорезми и принято считать термином «алгоритм». Так что слово «алгоритм» — это фактически название города. Тем не менее именно этот учёный написал математический трактат с описанием правил, т.е. алгоритмов, решения линейных и квадратных уравнений.

Формальному исполнителю для его работы нужна инструкция. Обсудим, как создаются такие инструкции и каким требованиям они должны удовлетворять.

ПОНЯТИЕ АЛГОРИТМА И ПРОГРАММЫ Инструкция, указывающая исполнителю, какие действия и в каком порядке он должен выполнить, и записанная на формальном языке этого исполнителя, называется **программой**. Однако написанию программы обычно предшествует разработка **алгоритма**, т.е. описание на формализованном языке приводящей к определённому результату последовательности действий, допустимых для некоторого исполнителя.

Различие между понятиями алгоритма и программы весьма тонкое, однако для поиска ошибок в программе даже есть специальный термин — **отладка**. Отладка в качестве составной части обязательно включает в себя поиск ошибок, допущенных именно в *записи* команд. Такие ошибки принято называть **синтаксическими**, поскольку они вызваны нарушением грамматики языка, понимаемого исполнителем. Ошибки другого вида могут вызвать реакцию исполнителя «Не могу исполнить»; такие ошибки называются **семантическими** — исполнитель понимает смысл отданной ему команды, но исполнение её по каким-то причинам невозможно. Например, вы предлагаете исполнителю разделить одно число на другое, а делитель оказался нулём.

Обычно алгоритмы пишутся так, чтобы они были понятны человеку. И мы будем записывать алгоритмы на обычном русском языке. При этом каждый из вас может применять какие-либо сокращения слов и вообще заменять одни слова другими. Важно только, чтобы за этими словами стояли действия, допустимые для данного исполнителя. Самого исполнителя удобно представлять себе как устройство управления, соединённое с набором инструментов. Устройство управления воспринимает и анализирует алгоритм, а затем организует его выполнение, командуя соответствующими инструментами. От исполнителя требуется лишь чёткое выполнение каждого действия, входящего в алгоритм. Мы не должны объяснять ему, для каких целей предназначается алгоритм.

Алгоритмы нередко обладают теми или иными свойствами. Перечислим основные из них.

Прежде всего назовём **дискретность**. Под дискретностью понимается то, что алгоритм состоит из описания последовательности тактов обработки, организованной таким образом, что в начальный момент задаётся исходная ситуация, а в каждый следующий момент ситуация преобразуется на основе данных, полученных в предшествующие такты обработки. Дискретность алгоритма означает, что он исполняется по шагам: каждое действие, предусмотренное алгоритмом, исполняется только после того, как закончилось исполнение предыдущего.

Другое свойство принято называть **детерминированностью**. Оно означает, что на каждом шаге однозначно опреде-

лено преобразование объектов среды исполнителя, полученных на предшествующих шагах алгоритма.

Третье свойство — **результативность** алгоритма. Это свойство подразумевает, что каждый шаг (и алгоритм в целом) после своего завершения создаёт ситуацию, в которой все имеющиеся объекты однозначно определены. Если это по каким-либо причинам невозможно, то алгоритм должен сообщать, что решение задачи не существует.

Свойство результативности алгоритма в целом обычно одновременно подразумевает и **конечность** алгоритма, т.е. завершение его работы за конечное число шагов (при этом количество шагов может быть заранее неизвестным и различным для разных начальных ситуаций). Если процесс исполнения алгоритма за конечное число шагов не заканчивается, то говорят, что **алгоритм зациклился**.

Необходимым признаком алгоритма является использование в нём только допустимых действий исполнителя. Это свойство нередко называют **понятностью** алгоритма. Термин понятность довольно естествен, поскольку реакция исполнителя на записанное в алгоритме действие, не являющееся допустимым, всегда одна: «Не понимаю».

С тем, как эти свойства реализуются в алгоритмах, вы неоднократно встретитесь при дальнейшем изучении алгоритмизации. Но надо сказать, что вовсе не каждый алгоритм обязан обладать всеми указанными свойствами. Представим, к примеру, что любой алгоритм должен быть конечным, т.е. завершать работу за конечное число шагов. А если это алгоритм управления космической станцией или термоядерным реактором? Момент прекращения работы алгоритма — это момент катастрофы. Впрочем, не надо обращаться к столь далёким от вашей практики примерам. Каждая операционная система на вашем компьютере — это бесконечно исполняемый алгоритм. В тот момент, когда операционная система прекратит работу, ваш компьютер станет бесполезен. Можно сказать, что все алгоритмы управления объектами в режиме реального времени не могут быть конечными.

Но в вашей школьной практике вам вряд ли потребуются бесконечно исполняемые алгоритмы. Поэтому, если по составленной вами программе компьютер не собирается заканчивать работу, ищите ошибку в программе.

Теория алгоритмов постоянно развивается. Всё большую роль играет параллельное программирование, т.е. создание алгоритмов, в которых те или иные действия над обрабатываемыми данными выполняются одновременно, а не последовательно. Время, нужное для получения результата, сокращается в разы. Но для этого требуется разрабатывать и применять на практике методы синхронизации действий, выполняемых в алгоритме.

Конечно, для каждого исполнителя алгоритм записывается на том формальном языке, который понятен этому исполнителю. Так что программы, реализующие один и тот же алгоритм, могут выглядеть по-разному. Но важно также понимать, что один и тот же алгоритм, исполняемый разными исполнителями, может давать различные результаты — это зависит от того, как для него реализованы соответствующие допустимые действия.

Мухáммад аль-Хорезмú (783—850)

ИССЛЕДОВАНИЯ

Проанализируйте, с какими алгоритмами вы познакомились на уроках русского языка, иностранного языка, математики, химии, физкультуры, информатики. Какой из предметов оказался для вас наиболее, а какой наименее алгоритмизированным?

ВОПРОСЫ:

- Любые ли действия могут присутствовать в алгоритме, предназначенном для данного исполнителя?
- Какие общие свойства алгоритмов вы можете назвать?
- В чём различие между алгоритмом и программой?
- Что такое отладка программы?
- Почему исполнение одного и того же алгоритма разными исполнителями может давать разные результаты?

21

ЛИНЕЙНЫЕ И РАЗВЕТВЛЯЮЩИЕСЯ АЛГОРИТМЫ

ВЫ УЗНАЕТЕ:

- Как может быть организована последовательность действий в алгоритмах
- Что такое ветвление в алгоритме
- Какие формы условного оператора применяются при составлении алгоритмов

ВСПОМНИТЕ:

- Что называют алгоритмом
- Какие действия могут использоваться в алгоритме

Алгоритм — это всегда организованная некоторым образом последовательность действий. А как может быть такая последовательность организована? Об этом рассказывается в данном разделе.

ПОСЛЕДОВАТЕЛЬНОЕ ИСПОЛНЕНИЕ ДЕЙСТВИЙ. ЛИНЕЙНЫЙ АЛГОРИТМ

Самый простой способ указать, какие действия должен выполнить исполнитель, — записать их одно за другим в виде списка. В первых языках программирования такой список нередко был нумерованным, в современных языках от нумерации давно отказались. Алгоритм, в котором последовательность исполняемых действий задана списком, называется **линейным**.

ВЕТВЛЕНИЯ В АЛГОРИТМЕ. УСЛОВНЫЙ ОПЕРАТОР Жизнь и деятельность любого исполнителя редко удаётся описать линейным алгоритмом. Как правило, в инструкции исполнителю требуется предусмотреть вариативность действий в зависимости от складывающихся условий. Так, подъезжая к перекрёстку, регулируемому светофором, водитель проследует его либо без остановки, если на светофоре зелёный сигнал, либо остановится в противном случае. По крайней мере так он должен поступить как дисциплинированный исполнитель правил дорожного движения.

Правила дорожного движения — это, конечно, инструкция водителю или пешеходу, но не алгоритм его действий. В алгоритме, как вы помните, на формализованном языке должна быть записана последовательность допустимых действий формального исполнителя. Значит, надо договориться о формализованной записи выбора исполняемых действий. Вот эта запись:

```
Если (условие) то
{   действие;
    действие;
    ...
    действие;
}
Иначе
{   действие;
    действие;
    ...
    действие;
}
```

Здесь представлен именно шаблон формализованной записи выбора той или иной группы действий в зависимости от того, выполнено заданное условие или нет. В реальном алгоритме вместо слова «условие» должно быть записано именно то условие, выполнение которого исполнитель про-

ИССЛЕДОВАНИЯ

Проанализируйте, как часто среди алгоритмов, которые вам приходится исполнять при изучении различных предметов или в быту, вам приходится делать выбор исполняемых действий.

Схема линейного алгоритма

веряет *до того*, как приступить к исполнению последовательности действий, записанных единым блоком в первых фигурных скобках. Если условие выполнено, то действия указанного блока исполняются; если условие не выполнено, то исполняются действия из блока, записанного после слова **Иначе**.

Фигурные скобки показывают границы блока действий, которые исполняются в зависимости от выполнения или невыполнения условия. Их называют **операторными скобками**. В языке программирования, на котором записывается алгоритм для конкретного исполнителя, операторные скобки могут выглядеть по-другому.

В этой записи слова **Если**, **то**, **иначе** являются неотъемлемыми элементами формализованного языка записи алгоритмов. Обычно их называют **служебными** или **зарезервированными** словами алгоритмического языка. В будущем появятся и другие служебные слова алгоритмического языка. В записи алгоритмов служебные слова мы будем выделять жирным шрифтом. И снова скажем: в языке программирования, на котором алгоритм записывается для конкретного исполнителя, служебные слова могут выглядеть по-другому.

Ожидается, что в скором времени вместо шофёра-человека автомобилем будет управлять робот. Тогда для него алгоритм проезда через перекрёсток будет выглядеть так:

```
Получить сигнал от светофора;
Если (зелёный) то
{  Продолжить движение;
}
Иначе
{  Остановиться;
}
```

В рассматриваемом нами способе организации действий происходит как бы разветвление на два варианта действий. Поэтому такая конструкция называется **ветвлением**, а её запись в форме **«Если ... то ... Иначе»** называется **условным оператором** или **оператором ветвления**.

В условном операторе может отсутствовать «веточка», начинающаяся словом **Иначе** — ведь вполне возможно, что если условие не выполняется, то и делать ничего не требуется. Тогда говорят, что имеет место **ветвление в неполной форме**.

Хотя набор допустимых действий у каждого исполнителя свой, способы организации действий для всех исполнителей одинаковы. Чтобы наглядно представлять формы организации действий, полезны так называемые **схемы алгоритмов**. В схеме каждое действие алгоритма, кроме проверки условия, помещают в прямоугольник, а вопрос о том, выполняется ли некоторое условие, — в ромб. Блоки соединяют отрезками или отрезками со стрелками, показывая очерёдность выполнения действий. Если на таком отрезке стрелка отсутствует, то блок, расположенный ниже, должен выполняться позже. На рисунке изображены схемы линейного алгоритма, ветвлений в полной и неполной формах.

Все операторы, к которым относится условие, необходимо заключать в операторные скобки.

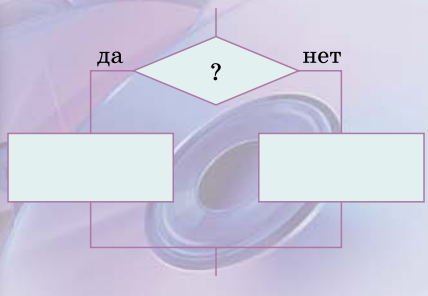


Схема ветвления в полной форме

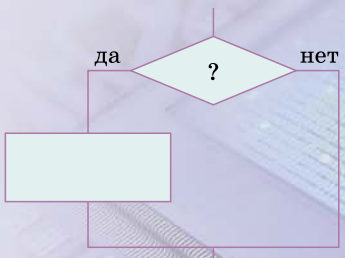


Схема ветвления в неполной форме

ВОПРОСЫ:

- Что такое условный оператор?
- Какую форму условного оператора называют полной? А какую неполной?
- Почему целесообразно рассматривать две формы условного оператора — полную и неполную?

22

ЦИКЛИЧЕСКОЕ ИСПОЛНЕНИЕ АЛГОРИТМА

ВЫ УЗНАЕТЕ:

- Как в алгоритмах записывается повторение действий
- Что такое циклическая форма организации действий в алгоритме
- Как организуется цикл, если количество повторений тела цикла заранее не известно
- Какое свойство алгоритма называют массовостью

ВСПОМНИТЕ:

- Какую роль играет условие в конструкции ветвления
- Для чего используются операторные скобки в конструкции ветвления

Циклом (повтором) называется такая форма организации действий в алгоритме, при которой одна и та же последовательность действий совершается несколько раз (или ни разу) до тех пор, пока выполняется заданное условие.

Нередко бывает так, что тот или иной блок действий требуется исполнить несколько раз. Эффективность применения компьютера была бы слишком низкой, если бы пришлось в алгоритме каждый раз писать такой блок столько раз, сколько его нужно повторять. Для преодоления этой ситуации в способах организации действий предусмотрено специальное средство — цикл.

ЦИКЛ В ФОРМЕ «ДЕЛАТЬ ПОКА...» **Есть несколько способов организовать повторение действий. Вот как записывается один из них:**

```
Делать пока (условие)
{  Действие;
   Действие;
   ...
} (* Конец цикла *)
```

Запись **Делать пока** называется **заголовком цикла.** Идущий в операторных скобках блок действий называется **телом цикла.** Исполнение этого блока действий будет происходить только в том случае, если выполняется условие, записанное в заголовке цикла. Закончив в очередной раз исполнение тела цикла, исполнитель алгоритма проверяет, выполнено ли условие, указанное в заголовке цикла. Если оно выполнено, то снова исполняются все действия, записанные в теле цикла.

Условие, которое нужно проверить для того, чтобы принять решение, исполнять ли тело цикла в очередной раз, называется **условием продолжения цикла.**

В круглых скобках со звёздочками стоит **комментарий.** Он совершенно не нужен формальному исполнителю, но полезен тому человеку, который пишет алгоритм. Хорошо написанные комментарии помогают понять алгоритм даже тому, кто его не составлял. Кроме того, в алгоритмах с комментариями обычно оказывается гораздо меньше ошибок, чем в алгоритмах без них.

Слова **Делать пока** служебные и потому выделены в соответствии с нашей договорённостью.

По-другому цикл в форме **Делать пока** называют **циклом с предусловием**, его схема изображена на рисунке.

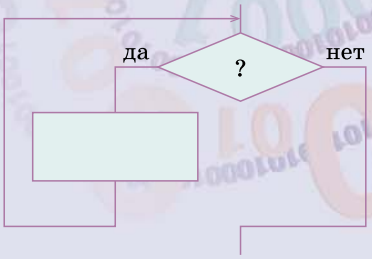


Схема цикла в форме **Делать пока**

ВЛОЖЕННЫЕ КОНСТРУКЦИИ В теле цикла вместо действия может быть записан оператор ветвления или оператор другого цикла. Аналогично и в конструкции ветвления вместо действия может быть записан снова оператор ветвления или оператор цикла. В этом случае говорят о **вложенных** циклах и ветвлениях. Вложенный цикл или ветвление в свою очередь может содержать собственные вложенные циклы и ветвления. Цикл (ветвление) по отношению к своему вложенному циклу (ветвлению) называется **внешним**, а вложенный цикл (ветвление) — **внутренним.**

СВОЙСТВО МАССОВОСТИ АЛГОРИТМА Алгоритмы в жизни людей появились намного раньше компьютеров. Некоторые из них навсегда ушли из человеческой практики. Никто, например, сейчас не производит действия с дробями по тем алгоритмам, которые применялись древними египтянами. Другие алгоритмы, столь же давно изобретённые, и сегодня играют важную роль. Вот один из таких алгоритмов, который называется алгоритмом Евклида в честь описавшего его знаменитого математика древности. Он предназначен для вычисления наибольшего общего делителя двух натуральных чисел. Мы представим его так, как если бы пред вами был лист папируса, на котором только и можно, что записывать процесс исполнения алгоритма — никаких компьютеров и калькуляторов ещё никто не придумал!

Евклид
(ок. 300 г. до н. э. — до 265 г. до н. э.)

Алгоритм НОД
{ Записать первое натуральное число;
Записать второе натуральное число;
Делать пока (первое из чисел не равно второму)
{ **Если** (первое число больше второго) **то**
{ Вместо первого числа записать разность между первым и вторым числом;
}
Иначе
{ Вместо второго числа записать разность между вторым и первым числом;
}
}
}

После исполнения этого алгоритма на листе будут записаны два равных числа. Это и есть наибольший общий делитель исходных чисел.

Этот алгоритм позволяет решить не одну конкретную задачу — скажем, найти НОД чисел 178 357 и 1 049 254, — а позволяет решать любую задачу такого типа. Свойство алгоритма, состоящее в возможности его применения к решению различных однотипных задач, называется **массовостью**.

В теории алгоритмов, говоря более строго, массовым принято называть алгоритм, применимый к бесконечному множеству исходных данных.

Надо сказать, что вовсе не каждый алгоритм обязан обладать свойством массовости. Иногда бывает нужен алгоритм для решения одной, но очень трудоёмкой задачи. К примеру, нужно вычислить первые 100 000 знаков после запятой для числа π. Без компьютера здесь не обойтись, а значит, нужен и алгоритм. Такой алгоритм был составлен, задача была решена. Но вряд ли этот алгоритм пригодится для решения какой-нибудь другой задачи. Иными словами, массовость алгоритма определяется множеством тех исходных данных, к которым данный алгоритм может быть применён. Областью применимости алгоритма Евклида служит множество всевозможных пар натуральных чисел. А для нахождения 100 000 знаков числа π нам вообще не нужны какие бы то ни было исходные данные.

ВОПРОСЫ:

- Когда целесообразно применять оператор цикла?
- Что такое заголовок цикла? Что такое тело цикла?
- Какую роль играет условие в конструкции оператора цикла?
- Нужны ли комментарии формальному исполнителю?
- Какие циклы называются вложенными?
- В результате исполнения некоторого действия, записанного не последним в теле цикла, условие в заголовке цикла перестало быть истинным. Будут ли исполнены оставшиеся действия тела цикла?
- В чём состоит свойство массовости алгоритма?

23

ПЕРЕМЕННЫЕ В АЛГОРИТМАХ

ВЫ УЗНАЕТЕ:

- Как хранится информация, используемая в алгоритмах
- Каковы основные атрибуты переменных
- Что такое тип переменной и почему нужны переменные разных типов

ВСПОМНИТЕ:

- Виды информации, с которыми может работать компьютер
- Известные вам формы операторов цикла
- Какими свойствами обычно обладают алгоритмы

Важным способом сохранения информации для обработки её в алгоритмах является запись этой информации в **переменную**. Каждая переменная имеет имя, содержащаяся в переменной информация, называется её **значением**. Вид хранимой в переменной информации характеризуется **типом** переменной.

Чтобы исполнить какой-либо алгоритм, компьютер должен иметь возможность извлекать из своей памяти и сохранять в ней информацию о значениях величин, фигурирующих в алгоритме. Для этого информация специальным образом структурируется, а формы, посредством которых хранится информация, называют **структурами данных**.

ПЕРЕМЕННЫЕ И ИХ АТРИБУТЫ Для хранения информации, обрабатываемой с помощью какого-либо алгоритма, используются **переменные**. У каждой переменной обязательно есть **имя**. Имена позволяют одни переменные отличать от других и указывать исполнителю, из какой переменной ему нужно взять информацию для обработки и в какую поместить результат.

Какие имена можно употреблять, зависит от исполнителя. Большинство компьютерных исполнителей понимает имена, состоящие из букв и цифр или только из букв (так что никаких фамилий и отчеств у переменных нет). Информация, которая хранится в переменной, называется её **значением**.

Самое важное действие над переменной состоит в том, что ей можно присвоить то или иное значение, например 1. Чтобы это сделать, используют **оператор присваивания**. Его мы будем обозначать символом :=. Например, если S — имя некоторой переменной, то указанное действие записывается так:

$S := 1$

Слева от знака присваивания всегда указывается имя переменной, а справа пишется выражение, состоящее из констант и имён переменных, связанных знаками операций, допустимых для данного исполнителя. В выражении могут присутствовать скобки, указывающие на изменение естественного порядка действий.

Если переменная уже имеет какое-то значение, то его можно изменить. Для этого есть несколько путей. Можно просто присвоить переменной новое значение, скажем:

$S := 7$

Можно указать, на сколько надо изменить исходное значение:

$S := S + 4$

Последняя запись означает, что к прежнему значению переменной надо прибавить 4 и результат снова присвоить переменной S. Естественно, после нескольких подобных операций будет трудно установить, что же изначально было в переменной S. Вообще, после выполнения очередной операции присваивания предыдущее значение переменной утрачивается безвозвратно! Поэтому, если нам хочется, чтобы переменная S помнила своё прежнее значение, то надо взять другую переменную, к примеру с именем T:

$T := S + 4$

Тогда в S сохранится её исходное значение, а переменная T будет иметь значение на 4 большее, чем значение переменной S.

Кроме того, среди допустимых действий исполнителя всегда есть сравнение значений переменных (больше, меньше, равно, не равно, не больше, не меньше). Наконец, если потребуется, исполнитель сообщит значение интересующей нас переменной. Это действие договоримся записывать так:

Сообщить <имя переменной>

Можно попросить исполнителя сообщить и некоторый текст. Соответствующее действие записывается так:

Сообщить "<текст>"

(кавычки указывают исполнителю, что здесь записано не имя переменной, а некоторое сообщение).

Получение информации формальным исполнителем от пользователя мы будем обозначать допустимым действием

Запросить <имя переменной>

По этой команде переменной с указанным именем будет присвоено значение, вводимое пользователем.

Алгоритмы выполнения действий над числами зависят от того, какие это числа. Например, десятичные дроби складываются совсем не так, как обыкновенные. Поэтому формальный исполнитель должен знать, с числами какого вида он будет иметь дело. Правда, в алгоритме редко присутствуют сами числа. Обычно в нём фигурируют переменные, которые, словно пустая тара, всегда готовы принять в себя числа на временное хранение. Но никто ведь не хранит подсолнечное масло в бумажных пакетах или сахар в сетках. Каждому виду продукта — свой тип тары. Также и с переменными: каждому виду хранимой в ней информации — свой **тип переменных**. Поэтому и переменные бывают нескольких типов. Прежде всего используются переменные для хранения **целых** и **вещественных** чисел. Для обработки текстовой информации используются **символьные** переменные (иногда их называют **строковыми**), а для выполнения логических операций — **логические**, или **булевы**.

Приступая к написанию программы, вы обязаны сообщить исполнителю, каков тип используемых в ней переменных. Эту информацию записывают с помощью специальной конструкции, называемой **объявлением переменных**. В записи алгоритмов мы будем применять следующие формы записи объявления переменной:

вещ: < список имён переменных через запятую> — объявление переменных вещественного типа,

цел: < список имён переменных через запятую> — объявление целочисленных переменных,

сим: < список имён переменных через запятую> — объявление переменных символьного (строкового) типа,

лог: < список имён переменных через запятую> — объявление переменных логического (булева) типа.

В таблице на с. 98 приведены обычно используемые обозначения операций, а также типы операндов и результата этих операций.

Операция	Выражение	Типы операндов	Тип результата
Сложение	A + B	Оба целые	Целый
		Хотя бы один вещественный	Вещественный
Умножение	A * B	Оба целые	Целый
		Хотя бы один вещественный	Вещественный
Вычитание	A — B	Оба целые	Целый
		Хотя бы один вещественный	Вещественный
Деление	A / B	Целый или вещественный	Вещественный
Целочисленное деление	A div B	Оба целые	Целый
Остаток при целочисленном делении	A mod B	Оба целые	Целый
Целая часть	int(A)	Целый или вещественный	Целый
Корень квадратный	sqrt(A)	Целый или вещественный	Вещественный

При действиях с переменными символьного типа операнды и результат могут иметь как символьный, так и целый тип. Например, оператор *Длина* для символьной переменной вычисляет количество символов, которыми записано значение этой переменной. Поэтому у этого оператора тип операнда символьный, а тип результата целый. Для операций над булевыми переменными тип результата — булев.

Обратите внимание, что при записи алгоритма объявление переменных записывается до начала всех действий этого алгоритма, а совокупность всех действий алгоритма заключается в операторные скобки. Она называется **телом алгоритма**. Запись алгоритма (вместе с объявлением переменных) предваряется заголовком **Алгоритм**.

ЦИКЛ СО СЧЁТЧИКОМ Стремление переложить на плечи автоматических устройств как можно больше рутинной работы естественно и похвально. Пусть, например, у нас имеется автоматизированная теплица, в которой на десяти грядках выращивается клубника. Для успешного созревания её надо поливать. Конечно, в алгоритме для автоматического поливального устройства можно воспользоваться циклом **Делать пока**. Вот как мог бы выглядеть такой алгоритм:

```
Алгоритм
{  Полить первую грядку клубники;
   Делать пока (есть неполитая грядка)
   {  Перейти к следующей грядке клубники;
      Полить грядку клубники;
   }  (* Конец цикла *)
}
```

Но про исполнение данного цикла мы знаем гораздо больше, чем обычно: нам известно, сколько раз будет исполняться тело цикла и какая по счёту грядка будет поливаться при очередном исполнении цикла. Это знание отнюдь не лишнее. Оказывается, тогда можно воспользоваться другой конструкцией цикла:

```
Алгоритм
{  Делать от n := 1 до 10
   {  Подойти к n-й грядке клубники;
      Полить эту грядку клубники;
   }   (* Конец цикла *)
}
```

Здесь буквой n обозначена переменная, к значению которой при каждом выполнении тела цикла прибавляется 1. Эта переменная называется **счётчиком цикла**. Как только значение n станет равным 10, тело цикла исполнится последний раз. Если по каким-то причинам первую грядку поливать не нужно, то переменной n вы можете присвоить другое начальное значение (оно записывается в заголовке цикла сразу после знака операции присваивания).

Бывает, что счётчик цикла после каждого исполнения тела цикла нужно увеличивать не на 1, а на 2 или даже больше. Величину, на которую изменяется счётчик цикла после очередного исполнения тела цикла, называют **шагом**. Вот как эта информация записывается в заголовке цикла:

Делать от $n := 2$ **до** 10 **с шагом** 2.

Описанная алгоритмическая конструкция называется **циклом со счётчиком**. Схема цикла со счётчиком изображена на рисунке. В ней через **нз** и **кз** обозначены начальное и конечное значения счётчика.

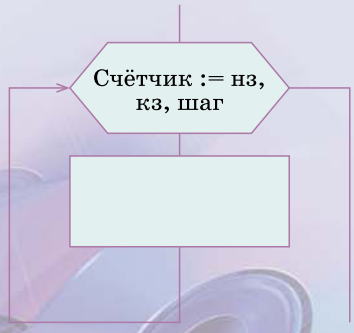


Схема цикла со счётчиком

Конструкция цикла со счётчиком используется во многих вычислительных задачах. Пусть, к примеру, нужно найти сумму

$$1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\frac{1}{6}+\frac{1}{7}+\ldots+\frac{1}{100}.$$

Как бы действовал человек, решая такую задачу на калькуляторе? Он бы к числу 1 прибавил число $\frac{1}{2}$, к полученному числу прибавил $\frac{1}{3}$, к получившемуся числу прибавил $\frac{1}{4}$, и т.д. Иными словами, если уже вычислена сумма S первых k слагаемых, то для вычисления суммы $k+1$ слагаемых надо к S прибавить $\frac{1}{k+1}$.

Таким образом, такое сложение нужно выполнить всего 99 раз. Поняв способ вычисления, мы можем теперь легко составить нужный алгоритм:

```
Алгоритм
цел: k;
вещ: S;
{  S := 1;
   Делать от k := 2 до 100
   {  S := S + 1/k;
   }
   Сообщить S;
}
```

ВОПРОСЫ:

- Для чего применяют переменные в алгоритмах?
- Для чего нужно имя переменной? Что называют значением переменной?
- С помощью какой операции можно присвоить или изменить значение переменной?
- Для чего нужен тип переменной? Как исполнитель узнаёт, с переменными какого типа ему придётся иметь дело при исполнении данного алгоритма?
- Когда целесообразно использовать цикл со счётчиком?
- Может ли тело цикла со счётчиком не исполняться ни разу?

24 ОТ ПЕРЕМЕННОЙ К МАССИВУ

ВЫ УЗНАЕТЕ:

- Как исполнители работают с таблицами
- Как поступают, если двумерной таблицы недостаточно

ВСПОМНИТЕ:

- Что такое таблица
- Какими параметрами характеризуются ячейки электронной таблицы

Мы уже обсуждали, что нередко ту или иную информацию удобно представлять и обрабатывать в табличной форме. И компьютер в этом всегда готов помочь человеку. Одним из популярных формальных исполнителей, реализованных в виде компьютерной программы, является уже хорошо знакомая вам электронная таблица. Однако этого исполнителя вы использовали, можно сказать, в режиме непосредственного управления — записали формулу в ячейку, нажали клавишу «Ввод» и компьютер воспринял это как команду на вычисление значения по данной формуле. Но, конечно, можно обрабатывать информацию, представленную в табличной форме, и в режиме программного управления. В этом случае подобные таблицы принято называть массивами. Можно, например, говорить о массиве, в котором хранятся дни рождения или имена учеников класса. В массив можно поместить таблицу турнира по теннису или расстояния между аэропортами какой-либо страны.

МАССИВ И ЕГО ХАРАКТЕРИСТИКИ Нетрудно понять, что каждая ячейка электронной таблицы — это отдельная переменная со своим именем (там оно называется адресом) и значением вполне определённого типа. В массиве каждый **элемент** (аналог ячейки электронной таблицы) представляет собой переменную, у которой есть имя, тип и значение. Из простейших переменных, как вещество из атомов, могут строиться сложные структуры данных. Массив — одна из таких структур.

Было бы, однако, очень неудобно составлять алгоритмы обработки данных в массиве, если у каждого его элемента было бы индивидуальное имя, никак не связанное с именами других элементов, и индивидуальный тип. Поэтому обычно все ячейки массива имеют один и тот же тип. Он называется **типом массива**. И имя у ячеек тоже в некотором смысле общее (как фамилии детей из одной семьи).

Обычно таблица имеет несколько строк и несколько столбцов. Массивы, соответствующие таким таблицам, называют **двумерными**. Номер строки и номер столбца в таблице называют **индексами** того элемента, который стоит в клетке на пересечении данной строки и данного столбца.

Если массив представляет собой таблицу, состоящую из одной строки или одного столбца, то такой массив называют **линейным**, или, по-другому, **одномерным**. В его описании будет использоваться только одно множество индексов.

Впрочем, массивы могут быть и трёхмерными, и четырёхмерными, и ещё большей размерности — всё зависит от задачи, которую приходится решать. Количество наборов используемых индексов называют **размерностью** массива.

Чтобы задать массив, надо указать его **имя**, **тип**, **диапазон** изменения каждого индекса, например:

вещ: M[1:14, 3:25, 5:70].

Здесь M — имя массива, размерность его равна трём, и у каждого из трёх индексов свой диапазон: для первого — от 1 до 14, для второго — от 3 до 25, для третьего — от 5 до 70. Границы диапазона входят в множество значений индекса.

Массив — это набор однотипных данных, снабжённых системой из одного или нескольких индексов, каждый из которых принимает последовательные целые значения.

При создании алгоритмов каждый элемент массива можно рассматривать как самостоятельную переменную. Её именем является имя таблицы с указанием значений индексов данного элемента. Значения индексов элемента будем указывать после имени массива в квадратных скобках через запятую. На рисунке показано, как располагаются элементы двумерного массива $A[1{:}5,\ 1{:}5]$. Например, в массиве $A[1{:}5,\ 1{:}5]$ элемент, стоящий во второй строке и третьем столбце — это переменная с именем $A[2,\ 3]$. Тип каждой переменной, представляющей элемент массива, совпадает с типом всех остальных элементов в таблице.

$A[1,\ 1]$	$A[1,\ 2]$	$A[1,\ 3]$	$A[1,\ 4]$	$A[1,\ 5]$
$A[2,\ 1]$	$A[2,\ 2]$	$A[2,\ 3]$	$A[2,\ 4]$	$A[2,\ 5]$
$A[3,\ 1]$	$A[3,\ 2]$	$A[3,\ 3]$	$A[3,\ 4]$	$A[3,\ 5]$
$A[4,\ 1]$	$A[4,\ 2]$	$A[4,\ 3]$	$A[4,\ 4]$	$A[4,\ 5]$
$A[5,\ 1]$	$A[5,\ 2]$	$A[5,\ 3]$	$A[5,\ 4]$	$A[5,\ 5]$

Чтобы пользователь мог заполнить массив нужными ему значениями, обычно применяют вложенные циклы. Например, для заполнения массива $A[1{:}3,\ 1{:}4]$ пишут:

```
Делать от i := 1 до 3
{  Делать от j := 1 до 4
   { Запросить A(i, j);
   }
 }
```

Чтобы каждый раз в алгоритмах не писать такой двойной цикл, договоримся сокращённо его записывать одним оператором

Запросить $A[1{:}3,\ 1{:}4]$

Конечно, если речь идёт о заполнении другого массива, то после слова **Запросить** надо указать имя и размеры того массива, который нужен.

Если же заполнение массива происходит в результате вычислений, то для этого применяют операцию присваивания. Например:

```
Делать от i := 1 до 3
{  Делать от j := 1 до 4
   {  A[i, j] := i*i − j +1;
   }
}
```

Наконец, если массив уже загружен в память компьютера, то работа с ним тоже нередко требует использования вложенных циклов. Вот, к примеру, алгоритм поиска в числовом массиве $A[1{:}3,\ 1{:}4]$ максимального числа (оно будет получено в переменной с именем MAX):

```
Алгоритм
цел: i, j, MAX;
цел:  A[1:3, 1:4];
{  Запросить A[1:3, 1:4];
   MAX := A[1, 1];
   Делать от i := 1 до 3
   {  Делать от j := 1 до 4
      {  Если (A[i, j] > MAX) то
         {  MAX := A[i, j];  }
      }   (*Конец цикла по j*)
  }   (*Конец цикла по i*)
  Сообщить MAX;
}
```

ВОПРОСЫ:

- Что называют массивом в программировании?
- В чём отличия массивов от таблиц?
- Что такое тип массива? Что такое размерность массива?
- Как связаны имена элементов массива с именем самого массива?

25

ЗНАКОМСТВО С ЯЗЫКАМИ ПРОГРАММИРОВАНИЯ

ВЫ УЗНАЕТЕ:

- На каком языке пишут программы для компьютера
- С чего начать изучение языка программирования

ВСПОМНИТЕ:

- Основные алгоритмические конструкции
- Чем характеризуются переменные
- Что такое массив и каковы его характеристики

Подавляющее большинство компьютеров имеют совершенно нечеловеческую систему команд. И общаться с ними на их родном языке весьма затруднительно. Поэтому для составления программ используется какой-либо промежуточный язык, более понятный человеку. Таких языков много, но мы расскажем только о двух. Это языки КуМир и Паскаль (Pascal). Их выбор обусловлен двумя причинами. Во-первых, это языки, ориентированные как на профессиональное программирование, так и на обучение основам этой профессии. Во-вторых, тем, кто собирается сдавать ЕГЭ, будут предложены задания на этих языках.

ЯЗЫК ПРОГРАММИРОВАНИЯ: ОСНОВНЫЕ ОСОБЕННОСТИ Что значит изучить язык? Для обычного иностранного языка это значит выучить перевод слов и правила их соединения в предложения. То же самое и для языков программирования. К счастью, в языках программирования используется очень мало слов (как правило, чуть больше двух десятков), а правил их соединения ещё меньше. Правила эти, как вы, наверно, догадались, в точности те самые алгоритмические конструкции ветвлений и циклов, с которыми вы уже познакомились.

Удобно себе представлять каждый из языков программирования как язык некоторого формального исполнителя. Тогда, как обычно, для него есть среда, состоящая из тех объектов, над которыми он может выполнять допустимые действия. В первую очередь такими объектами выступают числа, а допустимые действия — это действия над числами. Арифметические операции в языках программирования обозначаются, как правило, стандартными математическими знаками. Арифметические операции выполняются с числами, а также с различными переменными числовых типов.

Вычисления — это всего лишь одна из многих разновидностей обработки информации. Поэтому языки программирования рассчитаны отнюдь не только на вычисления. Чтобы описать все возможности какого-либо языка программирования, пишут специальное и довольно большое руководство. Мы же ограничимся лишь простейшими «способностями» двух указанных языков.

Конечно, компьютерные программы обрабатывают не только числа. Значит, языки программирования должны допускать и другие типы переменных. Один из них — логический (по-другому, булевый). Переменная такого типа принимает только два значения: Истина и Ложь. Над булевыми переменными можно выполнять логические операции: соединять их союзами И и ИЛИ, а ещё ставить перед ними частицу НЕ. Результат этих операций определяется так называемыми таблицами истинности. В таблице приведены обозначения логических операций и логических значений в языках программирования КуМир и Паскаль.

Инициатором создания языков программирования стала знаменитая фирма IBM (International Business Machine Company). Было крайне заманчиво предложить компьютер, доступный не только профессиональным программистам, но и научным работникам и инженерам, привыкшим к алгебраическим формулам. В 1953 году Джон Бэкус предложил язык, названный FORTRAN (FORmula TRANslator — переводчик формул). Он позволял не только «переводить формулы» на машинный язык, но и автоматизировал организацию циклов.

Операция или логическое значение	КуМир	Паскаль
И	и	and
ИЛИ	или	or
НЕ	не	not
Истина	истина	true
Ложь	ложь	false

Логические значения получаются при сравнении значений переменных. К примеру, по команде (записанной по правилам языка Паскаль)

q := (x > −2) **and** (x < 6)

логической переменной q будет присвоено значение true, если числовая переменная x имеет, например, значение 4. Если же x имеет значение −3, то q получит значение false.

Как переводятся на языки КуМир и Паскаль основные алгоритмические конструкции, можно посмотреть в таблицах на оборотах обложки. А мы в качестве примера приведём запись одного и того же алгоритма на трёх языках. Это вычисление члена ряда Фибоначчи по его номеру, который задаётся пользователем программы. Вот этот алгоритм:

```
Алгоритм
цел: a1, a2, Раб, Номер, Счётчик;
{  Запросить Номер;
   a1:=0;
   a2:=1;
    Делать от Счётчик := 1 до Номер
    {  Раб := a1 + a2;
       a1 := a2;    a2 := Раб;
     }
  Сообщить Номер,"-й член равен ", a2
  Сообщить “равен”, a2;
}
```

КуМир:

```
алг Фиб
нач цел a1, a2, Раб, Номер, Счётчик
   ввод Номер
   a1 := 0; a2 := 1
   нц для Счётчик от 1 до Номер
    Раб := a1 + a2;
    a1 := a2;    a2 := Раб
   кц
   вывод Номер,"-й член равен ", a2
кон
```

Pascal:

```
program fib;
var a1, a2, n, r, i : integer;
begin
     readln(n);
     a1 := 0; a2 := 1;
     for i := 1  to  n do
     begin
       r := a1 + a2;    a1 := a2;     a2 := r;
     end;
     write ( n ,'-й член равен ', a2)
end.
```

Ни один из языков программирования в написании своих операторов не различает маленькие и большие буквы, поэтому пишите так, чтобы программа для вас выглядела удобно и красиво.

Итальянский математик Леонардо Пизанский (Фибоначчи), живший в XII в. в г. Пизе, был инициатором широкого использования позиционной системы счисления и арабских цифр вместо римской нумерации. В своих математических работах он исследовал последовательность, начинающуюся с двух единиц, а далее каждый член последовательности равен сумме двух предыдущих. Эта последовательность оказалась полезной при исследовании различных процессов в химии, биологии и других науках. Числа этой последовательности получили название чисел Фибоначчи.

В мире сейчас используется более 1000 различных языков программирования. Некоторые из них универсальные, т.е. могут использоваться для решения широкого круга задач, другие специализированы под задачи определённой области, например управления роботами. Попытайтесь выяснить, какие бывают специализированные языки программирования.

ВОПРОСЫ:

- Что значит изучить язык программирования?
- Какие значения принимают переменные логического типа?
- Какие операции можно выполнять над переменными логического типа?

26

МЕТОД ПОШАГОВОЙ ДЕТАЛИЗАЦИИ, ВСПОМОГАТЕЛЬНЫЕ АЛГОРИТМЫ И ПОДПРОГРАММЫ

ВЫ УЗНАЕТЕ:

- Как решают сложную задачу
- Для чего составляют вспомогательные алгоритмы
- Чем вспомогательный алгоритм отличается от основного
- Что такое подпрограмма
- Как оформить вспомогательный алгоритм
- Как «научить» исполнителя новым допустимым действиям

ВСПОМНИТЕ:

- Что такое тип переменной
- Каковы основные типы переменных, используемых формальными исполнителями

Чтобы быстрее добраться из одного пункта в другой, человек изобрёл поезда, автомобили, самолёты. Чтобы быстрее выполнить ту или иную работу, человек придумал разные хитроумные устройства. Чтобы быстрее решать задачи, человек создал компьютеры. Борьба за экономию времени и ресурсов пронизывает всю историю человеческой деятельности. Для этого придумывались самые разнообразные вспомогательные средства. Есть такое средство и для разработки алгоритмов. Это **вспомогательные алгоритмы**. Конечно, циклов и ветвлений вполне достаточно, чтобы записать любой алгоритм. Поэтому вспомогательными алгоритмами пользоваться не обязательно (ведь и в Москву из Владивостока можно прийти пешком, не пользуясь транспортом). Но они экономят время и силы, уменьшают количество ошибок. Прежде чем обсуждать те или иные особенности вспомогательных алгоритмов, мы познакомим вас с одним общим подходом к решению задач.

МЕТОД ПОШАГОВОЙ ДЕТАЛИЗАЦИИ Человеку нередко приходится исполнять различные алгоритмы. Но гораздо приятнее поручить такую работу какому-нибудь автоматическому устройству, т.е. формальному исполнителю. Более того, нередко формальный исполнитель справится с алгоритмом гораздо лучше человека — человек может отвлечься и неточно выполнить предписанное действие, может вообще задуматься, стоит ли исполнять данный алгоритм, а может начать его модифицировать, и тогда есть риск, что исполнение алгоритма вообще никогда не закончится. А всё потому, что человек, напомним, является не формальным, а эвристическим исполнителем.

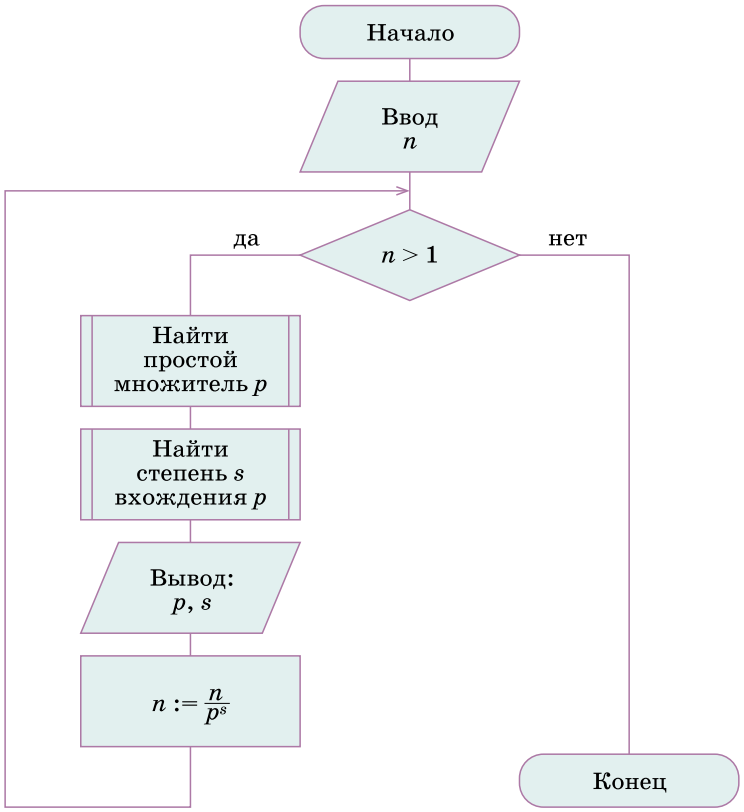
Эффективным способом решения сложной задачи является сведение её к решению системы более простых задач. Такой подход называют **методом нисходящего проектирования** или, по-другому, методом **пошаговой детализации**.

Зато умение составлять алгоритмы на сегодняшний день не доступно никакому формальному исполнителю. Точнее эту мысль надо выразить так: конечно, есть алгоритмы, позволяющие для некоторых классов задач автоматически строить алгоритмы их решения, но эти классы задач весьма узкие. Однако люди придумали методы, облегчающие создание алгоритмов. Одним из таких методов является **метод нисходящего проектирования**. По-другому его называют **методом пошаговой детализации**. Продемонстрируем этот метод на следующей задаче.

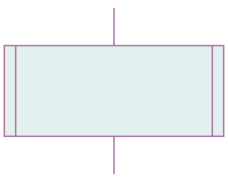
Вы знаете, что любое натуральное число, большее 1, можно представить в виде произведения степеней различных простых чисел. Скажем, $720 = 2^4 \cdot 3^2 \cdot 5^1$, $32 = 2^5$, $71 = 71^1$. Требуется составить алгоритм, который для каждого натурального $n > 1$ указывал бы все простые множители, входящие в такое разложение, и показатели степеней, с которыми они присутствуют в этом разложении. Недолгие размышления подсказывают, что первым шагом, по-видимому, должно

быть определение хотя бы одного простого множителя, делящего данное число n. Дальше, однако, путь раздваивается: можно попытаться определить показатель степени, в которой уже найденный простой множитель входит в разложение числа n, или попытаться найти следующий простой множитель и т.д., а только потом для каждого такого множителя найти показатель степени, с которым он входит в разложение числа n. Справа приведены схемы этих двух вариантов пошаговой детализации.

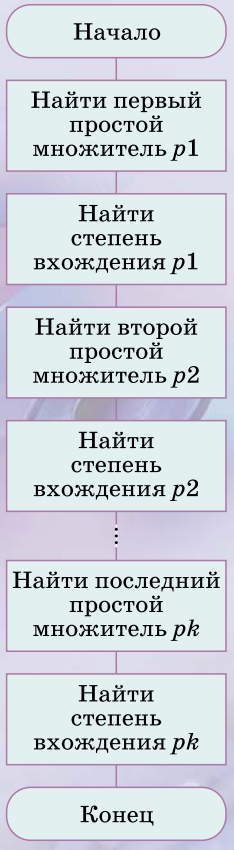
Проведенная нами пошаговая детализация показывает, что для решения исходной задачи нужно научиться решать две другие задачи: находить простой множитель в разложении натурального числа n и вычислять показатель степени, с которой он входит в разложение числа n. А поскольку количество различных простых множителей заранее неизвестно, то, по-видимому, придётся воспользоваться конструкцией цикла в форме **Делать пока**. Будем далее рассматривать первый вариант пошаговой детализации и представим решение исходной задачи в виде следующей схемы.



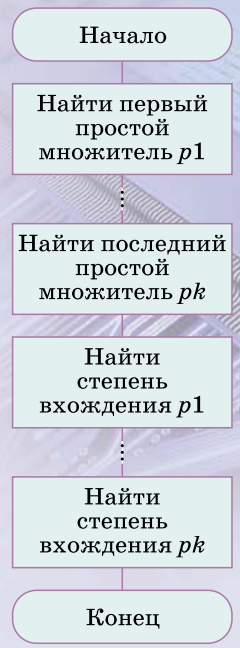
В этой схеме дважды употреблён блок:



а)



б)



Два алгоритма для нахождения разложения натурального числа n на простые множители

В нём записано не действие, которое должен выполнить исполнитель, а новая задача, для которой опять-таки требуется составить алгоритм. Такой алгоритм, который нужен для решения вспомогательной задачи, называют **вспомогательным алгоритмом**.

ВСПОМОГАТЕЛЬНЫЙ АЛГОРИТМ

В рассматриваемой нами задаче нам на самом деле требуется составить два вспомогательных алгоритма. И это дело можно поручить двум разным людям. Без сомнения, при таком распараллеливании работы на составление всего алгоритма потребуется меньше времени. Тому, кто будет заниматься основным алгоритмом, вряд ли надо вникать, как составлен вспомогательный алгоритм, но он, как минимум должен знать, какие исходные данные предстоит обрабатывать вспомогательному алгоритму. Для первого из вспомогательных алгоритмов это натуральное число n, для второго — не только n, но и простой множитель p, найденный в результате работы первого вспомогательного алгоритма. Значит, всякий вспомогательный алгоритм характеризуется исходными данными, которые он будет обрабатывать, и результатами, которые с его помощью будут получены. Кроме того, вспомогательных алгоритмов может быть много, их надо различать, поэтому естественно каждому из них дать собственное имя. Всё это записывается в виде заголовка вспомогательного алгоритма:

Алгоритм Имя **(арг:** <тип>: <список переменных>; **рез:** <тип>: <список переменных>**)**

В переменных, которые указаны в списке после служебного слова **арг** (сокращение слова «аргументы»), будут содержаться исходные данные, подлежащие обработке данным алгоритмом, а в переменных, записанных после служебного слова **рез** (сокращение слова «результаты»), будут сохранены результаты работы алгоритма. Например, для первого вспомогательного алгоритма заголовок может выглядеть так:

Алгоритм Простой множитель **(арг: цел:** n; **рез: цел:** p**).**

А для второго вспомогательного алгоритма заголовок может быть таким:

Алгоритм Степень множителя **(арг: цел:** n, p; **рез: цел:** s**).**

Разумеется, имена алгоритмов и переменных могут быть другими. Например, вместо длинного «Простой множитель» можно написать сокращение «ПМ». С именами переменных в списке аргументов и результатов ситуация несколько сложнее. Во-первых, поскольку вспомогательные алгоритмы создаются разными людьми, то было бы сложно заранее договариваться об именах используемых переменных в разных вспомогательных алгоритмах. Во-вторых, тот или иной вспомогательный алгоритм может оказаться востребованным в совершенно разных задачах. Не писать же каждый раз алгоритм заново только потому, что переменные в основном или других вспомогательных алгоритмах обозначены по-другому. Поэтому существуют правила описания переменных для вспомогательных алгоритмов.

Вспомогательный алгоритм обязательно имеет имя. Он может иметь переменные, которые называются аргументами — в них заносятся исходные данные, предназначенный для обработки этим алгоритмом, и переменные, называемые результатами, — в них записываются результаты обработки исходных данных этим алгоритмом.

Переменные, которые записаны в заголовке вспомогательного алгоритма, называются **формальными параметрами** этого алгоритма — они всего лишь сообщают пользователю вспомогательным алгоритмом, сколько переменных требуется для передачи исходных данных и результатов. Конечно, культурный разработчик вспомогательного алгоритма обязательно прокомментирует, какая именно информация должна быть передана в каждую из формальных переменных или из неё получена. Например, для вспомогательного алгоритма *Простой множитель* такой комментарий мог бы выглядеть так:

Алгоритм Простой множитель (**арг**: *n*; **рез**: *p*)
(**n — исходное число, p —простой делитель числа n**)

Переменные, с которыми будет исполняться алгоритм, указываются в специальной команде, которая называется **вызовом вспомогательного алгоритма**. Вместо переменных могут указываться конкретные значения. Вот как записывается такая команда:

Вызвать <имя вспомогательного алгоритма>
(<список значений или имён переменных>)

Переменные или значения, фигурирующие в операторе **Вызвать**, называются **фактическими параметрами вспомогательного алгоритма**.

Правила перегрузки значений из фактических параметров в формальные очень просты:

- если во вспомогательном алгоритме есть аргументы, то берётся первая переменная в операторе **Вызвать**;
- её содержимое (значение) перегружается в первую переменную после слова **арг** в заголовке вспомогательного алгоритма;
- если во вспомогательном алгоритме есть второй аргумент, то берётся вторая переменная в операторе **Вызвать**;
- её содержимое перегружается во вторую переменную после слова **арг** в заголовке вспомогательного алгоритма;
- если во вспомогательном алгоритме есть ещё третий аргумент, то третья переменная перегружается в третью и т.д.

После окончания работы вспомогательного алгоритма:

- берётся первая переменная после слова **рез** в заголовке вспомогательного алгоритма;
- её содержимое перегружается в первую переменную, которая осталась после перезагрузки аргументов в операторе **Вызвать**;
- содержимое второй — во вторую, третьей — в третью...

Понятно, что количество переменных и их тип в операторе **Вызвать** должны совпадать с количеством переменных и их типом в заголовке вызываемого вспомогательного алгоритма.

В языках программирования вместо слов «алгоритм» и «вспомогательный алгоритм» употребляются слова «программа» и «подпрограмма».

Имя вспомогательного алгоритма. Фактические параметры

Блок вызова вспомогательного алгоритма

Имя вспомогательного алгоритма. Формальные параметры

Блок начала вспомогательного алгоритма

В схеме фактические параметры указываются в блоке-заголовке вызова вспомогательного алгоритма, а формальные параметры — в блоке вспомогательного алгоритма.

Вот как может выглядеть запись основного и вспомогательных алгоритмов, позволяющих решить задачу разложения натурального числа на простые множители.

```
Алгоритм Разложение на простые множители
цел: n, p, s;
(* n — исходное число, p — простой делитель числа n,
s — степень множителя p *)
{ Запросить n;
   Если (n = 1) то
   {  Сообщить "Разложения не существует";
   }
   Делать пока (n > 1)
   {  Вызвать Простой множитель (n, p);
      Вызвать Степень множителя (n, p, s);
      Сообщить "Простой множитель", p;
      Сообщить "Его степень", s;
      n := n / (p ^ s);
    }      (* Конец цикла *)
}     (* Конец алгоритма *)

Алгоритм Простой множитель (арг: цел: n; рез: цел: p)
(* n — исходное число, p — простой делитель числа n *)
{  p := 2;
   Делать пока (n mod p ≠ 0)
   {  p := p + 1;
   }
 }     (* Конец вспомогательного алгоритма *)

Алгоритм Степень множителя (арг: цел: n, p; рез: цел: s).
(* n — исходное число, p — простой делитель числа n,
s — степень простого множителя p *)
{  s := 1;
   n := n / p;
   Делать пока (n mod p = 0)
   {  s := s + 1;
   }
} (* Конец вспомогательного алгоритма *)
```

Мы надеемся, что сам алгоритм и оба вспомогательных алгоритма вам понятны. Но у вас может возникнуть недоумённый вопрос — как это возможно, что в основном и вспомогательных алгоритмах исполнитель будет манипулировать с одними и теми же переменными?

Дело в том, что разработчик вспомогательного алгоритма не обязан знать, какими именами переменных будет пользоваться разработчик основного алгоритма. Поэтому имена переменных, фигурирующих на «территории» вспомогательного алгоритма, не имеют никакого отношения к именам тех переменных, которые участвуют в основном алгоритме. Про людей мы бы сказали, что это просто однофамильцы. Поэтому переменные, которые используются во вспомогательном алгоритме, называются **локальными**.

Переменные, используемые только в пределах вспомогательного алгоритма, никак не связаны с переменными, которые используются вне вспомогательного алгоритма. Поэтому они называются локальными переменными и для них могут использоваться те же имена, что и для переменных в основном алгоритме или других вспомогательных алгоритмах.

Обратите внимание, что все действия, относящиеся к основному или вспомогательному алгоритму, заключены в операторные скобки. За пределами скобок, кроме заголовка алгоритма, ещё объявляются переменные. Это можно понимать так, что вспомогательный алгоритм — это как бы новое допустимое действие исполнителя, которому он обучился благодаря нашим интеллектуальным усилиям. Это похоже на обучение младшего брата самостоятельно ходить в магазин за хлебом: первый раз вы подробно объясните дорогу и действия в магазине, а в следующие разы будете просто говорить: «Сходи за хлебом». Разумеется, операторные скобки играют и важную синтаксическую роль — они указывают, какие именно действия относятся к данной алгоритмической конструкции.

АЛГОРИТМ-ФУНКЦИЯ Вспомогательный алгоритм — мощное средство, облегчающее решение задач. Но всегда хочется иметь более удобный способ использования вспомогательных алгоритмов. Вспомните: когда нам требуется найти значение функции $\sqrt{x}$, мы пишем y := sqrt(x), а не вызываем подпрограмму вычисления этой функции. То есть подпрограмма, конечно, вызывается (ведь компьютер умеет выполнять только арифметические операции; всё остальное — это набор стандартных подпрограмм, встроенных в язык программирования), но мы этого не ощущаем и воспринимаем вычисление такой функции как ещё одно допустимое действие исполнителя. Инструментом для создания подобной функции служит алгоритмическая конструкция, которую называют **алгоритмом-функцией**.

Алгоритм-функция, как и любой алгоритм, имеет имя и набор аргументов. А вот переменной для результата у неё нет, поскольку вычисляемое ею значение всегда тут же присваивается переменной, указанной пользователем. Договоримся объявлять алгоритм-функцию следующим образом:

```
Функция Имя: <тип>
(арг: <тип>: <список переменных>)
<тип>: <список локальных переменных>;
{  <действие>;
    ...
   <действие>;
   знач := <результат>;
}
```

Возможно, вызовет недоумение употреблённое в описании алгоритма-функции слово **знач**. Так как **знач** стоит слева от знака операции присваивания, это имя некоторой переменной. Данная переменная служит сугубо внутренним целям алгоритма-функции — именно её тип указывается после имени функции.

Чтобы воспользоваться алгоритмом-функцией, достаточно написать команду

z := Имя (a_1, a_2, …, a_n);

Здесь a_1, a_2, …, a_n — значения аргументов, для которых требуется вычислить значение функции.

Алгоритм-функция — частный случай вспомогательного алгоритма. Для неё именем переменной, куда записывается результат, служит имя самой функции.

ВОПРОСЫ:

- Что такое вспомогательный алгоритм?
- Какая информация указывается в заголовке вспомогательного алгоритма?
- Какая информация указывается в команде вызова вспомогательного алгоритма?
- Что такое формальные и фактические параметры?
- Какие переменные называются локальными?
- Чем алгоритм-функция отличается от просто вспомогательного алгоритма?
- Когда можно вспомогательный алгоритм оформить в виде алгоритма-функции?

ВСПОМНИТЕ:

- Какого исполнителя называют формальным
- Что такое алгоритм
- Каковы основные алгоритмические конструкции
- В каких случаях в алгоритме приходится использовать оператор ветвления
- В чём различия оператора ветвления в полной и неполной формах
- Что такое оператор цикла с предусловием
- Как исполняется оператор цикла со счётчиком
- Чем характеризуется переменная
- Какие основные типы переменных используются в языках программирования
- Какие операции можно выполнять над логическими переменными
- Что называют массивом в теории алгоритмов
- Чем характеризуется массив
- Как описывается каждый элемент массива
- В чём состоит метод пошаговой детализации
- В чём польза применения вспомогательных алгоритмов
- Как оформляется вспомогательный алгоритм
- Как оформляется вызов вспомогательного алгоритма
- Что такое алгоритм-функция

ТЕОРЕТИЧЕСКИЙ ПРАКТИКУМ

Чтобы вы могли проверить, хорошо ли усвоен изученный материал, предлагаем выполнить следующие задания. Более широкий комплекс заданий вы найдёте в задачнике-практикуме.

ПОНЯТИЕ АЛГОРИТМА И ИСПОЛНИТЕЛЯ

1 Что, на ваш взгляд, ответит человек и что ответит формальный исполнитель, если к ним обратиться с вопросами:
а) Не можете ли вы сказать, который час?
б) Не знаете ли вы, который сейчас час?
в) Не будете ли вы любезны сказать, который сейчас час?
г) Который сейчас час?

2 Имеется два кувшина ёмкостью 3 л и 8 л. Исполнитель Джинн может набирать воду из реки в каждый кувшин, выливать из кувшина воду и определять, налита ли вода в кувшин доверху.

а) Составьте алгоритм, выполнив который Джинн наберёт из реки 7 л воды.
б) Пусть Джинну подменили трёхлитровый кувшин двухлитровым. Существует ли теперь алгоритм для этого исполнителя, позволяющий набрать из реки 7 л воды?

3 Злоумышленник выдал следующий алгоритм за алгоритм получения кипятка:

Налить в чайник воду.
Открыть кран газовой горелки.
Поставить чайник на плиту.
Ждать, пока вода не закипит.
Поднести спичку к горелке.
Зажечь спичку.
Выключить газ.

Исправьте алгоритм, чтобы предотвратить несчастный случай.

4 Пусть дан отрезок AB. Определите, для решения какой задачи предназначен следующий алгоритм:

Поставить ножку циркуля в точку A.
Установить раствор циркуля равным длине отрезка AB.
Провести окружность.
Поставить ножку циркуля в точку B.
Провести окружность.
Провести прямую через точки пересечения окружностей.

5 У исполнителя Увеличитель две команды, которым присвоены номера:

1. Прибавь 1.
2. Умножь на 2.

Первая из них увеличивает число на экране на 1, вторая умножает это число на 2.
Увеличитель понимает только линейные алгоритмы, и программа для него — это последовательность номеров указанных команд. Так, по программе **2112** из числа 2 получится число 12. Запишите все программы, содержащие не более 8 команд, с помощью которых из числа 3 можно получить число 13.

6 Петя и Коля в роли формальных исполнителей выполняют с помощью калькулятора следующий алгоритм:

Сложить 83,2438 и 57,6847.
Полученный результат умножить на 10.

У Пети на табло калькулятора помещается 6 цифр, а у Коли — 8. Будут ли у них одинаковыми результаты выполнения этого алгоритма?
Какой вывод можно сделать из результата выполнения этого задания?

УСЛОВНЫЙ ОПЕРАТОР

7 а) Проснувшись утром, Петя почувствовал недомогание. Находившийся рядом злоумышленник тут же предложил ему следующий алгоритм:

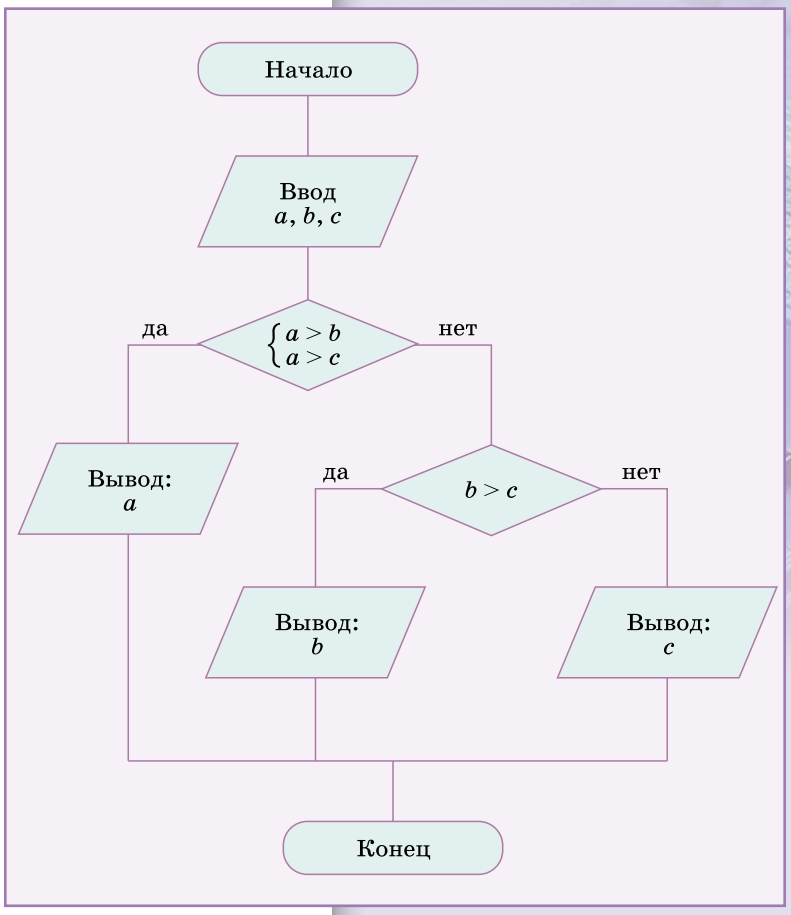
Измерить температуру;
Если (температура выше 37°) **то**
Вызвать врача;
Пойти в школу.

Исправьте этот алгоритм, чтобы при его исполнении Пете не стало хуже.
б) Однажды Петя решил позвонить своему приятелю. Злоумышленник предложил ему воспользоваться следующим алгоритмом:

Включить сотовый телефон;
Набрать номер;
Если (приятель ответил) **то**
{ Сказать: «Здравствуй!»;
Сообщить последние новости;
Узнать, что нового и как жизнь;
Сказать: «До свидания!»;
Выключить телефон;
}

Как исправить алгоритм, чтобы батарея не разрядилась до того, как телефон потребуется в следующий раз?

8 Исполните алгоритм, изображённый схемой при разных значениях a, b, c.



9 Дан четырёхугольник $ABCD$. Определите, для решения какой задачи предназначен следующий алгоритм:

Алгоритм
{ **Если** ($\angle A + \angle C = \angle B + \angle D$) **то**
{ Построить серединный перпендикуляр к стороне AB;
Построить серединный перпендикуляр к стороне BC;
Найти точку пересечения построенных перпендикуляров и обозначить её буквой O;
Построить окружность с центром в точке O радиусом OA;
}
}

Составьте схему этого алгоритма.

ОПЕРАТОР ЦИКЛА

10 Используя циклическую форму организации действий, запишите следующий алгоритм перевода текста с иностранного языка:

Прочитать первое предложение.
Перевести предложение.
Записать перевод в тетрадь.
Прочитать следующее предложение.
Перевести предложение.
Записать перевод в тетрадь.
Прочитать следующее предложение.
Перевести предложение.
Записать перевод в тетрадь.
...
Прочитать последнее предложение.
Перевести предложение.
Записать перевод в тетрадь.

Изобразите составленный алгоритм схемой.

11 а) Во время перемены Петя зашёл в столовую поесть пирожков. Для этого он составил следующий алгоритм:

Алгоритм
{ **Делать пока (не** исчезло чувство голода)
{ Купить пирожок; }
Съесть пирожок;
}

Сумеет ли Петя поесть пирожков? Исправьте алгоритм так, чтобы Петя ушёл из столовой сытым.
б) Однажды Пете задали на дом несколько задач по математике. Придя домой, он решил сначала сделать домашнее задание, а потом пойти гулять. Петя составил такой алгоритм:

Алгоритм
{ **Делать пока (не** решены все задачи)
{ Решить очередную задачу;
Пойти гулять до ужина;
}
}

Петя исполнил этот алгоритм в роли формального исполнителя и на следующий день получил двойку. Объясните почему. Исправьте Петин алгоритм, чтобы поставленная цель была достигнута.

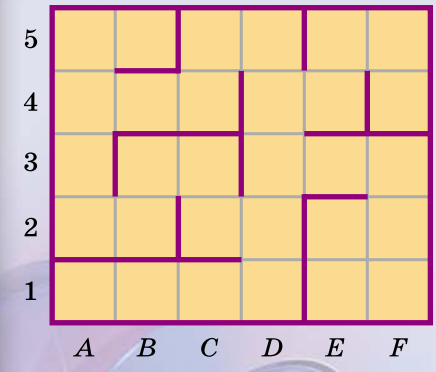
12 Исполнитель Робот перемещается по клетчатому полю, на котором между клетками могут стоять перегородки. Выходить за края поля и проходить сквозь перегородки Робот не может. Пусть стрелки означают команды:

↑ — переместиться на одну клетку вверх;
↓ — переместиться на одну клетку вниз;
← — переместиться на одну клетку влево;
→ — переместиться на одну клетку вправо.

Других команд Робот не понимает. Кроме того, он умеет проверять условия «слева стена», «справа стена», «сверху стена» и «снизу стена».

Дан алгоритм:

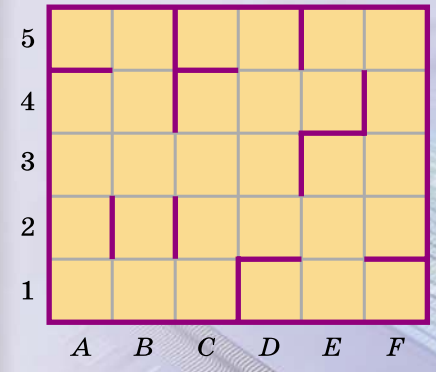
Алгоритм
{ **Делать пока** (сверху стена)
{ →; }
Делать пока (справа стена)
{ ↑; }
Делать пока (слева стена)
{ →; }
Делать пока (снизу стена)
{ ←; }
}



а) Исполните этот алгоритм для Робота на поле, изображённом на верхнем рисунке, если первоначально Робот стоит на клетке *A*1. Укажите, в какой клетке он остановится после исполнения этого алгоритма.
б) Первоначально Робот стоит на клетке *C*1. Какое диагностическое сообщение вынужден будет передать Робот при исполнении этого алгоритма? В какой клетке он будет при этом находиться?

13 Тот же Робот, что и в задании 12, исполняет следующий алгоритм на поле, изображённом на нижнем рисунке.

Алгоритм
{ **Делать пока** (**не** сверху стена)
{ ↑;
Делать пока (**не** справа стена)
{ →;
Делать пока (**не** снизу стена)
{ ↓;
Делать пока (**не** слева стена)
{ ←; }
}
}
}
}



а) Первоначально Робот стоит на клетке *B*1. Укажите, в какой клетке он остановится после исполнения этого алгоритма.
б) Первоначально Робот стоит на клетке *A*1. С какой проблемой столкнётся Робот при исполнении этого алгоритма?

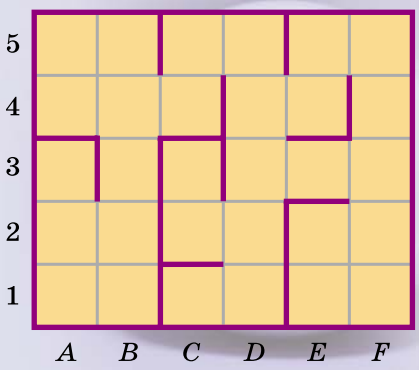
14 Чему будут равны a и b после выполнения каждого из следующих алгоритмов? Сколько раз будет выполняться тело цикла в каждом из них?

а) **Алгоритм**
цел: a, b;
{ $a := -5$; $b := -2$;
Делать пока ($a + b < a*b$)
{ $a := 2*a$; $b := b + 1$; }
} (* *Конец алгоритма* *)

б) **Алгоритм**
цел: a, b;
{ $a := 25$; $b := 1$;
Делать пока ($a - b > a*b$)
{ $a := a + 10$; $b := -3*b$; }
} (* *Конец алгоритма* *)

в) **Алгоритм**
цел: a, b;
{ $a := -3$; $b := -1$;
Делать пока ($a < b$)
{ **Если** ($b < 3$) **то** { $a := a - b$; $b := b + 2$; }
Иначе { $a := a + 2$; $b := b - a$; }
} (* *Конец цикла* *)
} (* *Конец алгоритма* *)

ВЛОЖЕННЫЕ КОНСТРУКЦИИ



15 Робот, описанный в задании 12, исполняет следующий алгоритм на поле, изображённом на рисунке слева.

Алгоритм
{ **Делать пока** (**не** сверху стена)
{ ↑;
Если (**не** справа стена) **то** { →; }
}
Делать пока (**не** снизу стена)
{ ↓;
Если (**не** слева стена) **то** { ←; }
}
}

а) Первоначально Робот стоит на клетке $A1$. Укажите, в какой клетке он остановится после исполнения алгоритма.
б) Первоначально Робот стоит на клетке $C2$. Укажите, в какой клетке он остановится после исполнения алгоритма.
в) Клетка $C2$ не единственная, начиная с которой Робот после исполнения алгоритма вернётся на ту же клетку, с которой начинал движение. Обследуйте оставшиеся 28 клеток этого поля (т.е. кроме клеток $A1$ и $C2$) и установите, какие из них обладают тем же свойством: Робот возвращается на ту же клетку, с которой начал движение.

16 Исполнитель Черепашка перемещается, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя есть две команды:

Вперёд n, вызывающая передвижение Черепашки на n шагов в направлении движения;

Направо m, вызывающая изменение направления движения на m градусов по часовой стрелке.

В командах вместо n и m должны стоять целые числа.
Черепашке был дан для исполнения следующий алгоритм:

Алгоритм
{ **Повторить** 5 **раз**
{ **Повторить** 4 **раза**
{ Вперёд 40;
Направо 72;
}
Направо 108;
}
}

Какая фигура будет начерчена Черепашкой?

РАБОТА С МАССИВАМИ

17 Какие значения переменных A и B сообщит компьютер, выполнив следующий алгоритм для двумерного массива M, изображённого таблицами *а*—*в*?

```
Алгоритм
цел A, B, I, J, M[1:5; 1:4];
{ Запросить M[1:5; 1:4];
  A := 0;
  B := 0;
  Делать от I := 1 до 5
  { Делать от J := 1 до 4
      { Если (M[I, J] = 1 то {  A := A + 1;  }
      Если M[I, J] = −1 то {  B := B + 1;  }
      }     (*Конец цикла*)
  }     (*Конец цикла*)
  Если (A > B) то {  B := A + B;  }
  Если (A < B) то {  B := A + B;  }
  Сообщить A;
  Сообщить B;
}
```

а)

0	1	0	−1
1	0	−1	1
2	−1	2	1
−1	0	−1	2
0	1	1	1

б)

−1	0	2	1
1	-1	0	2
0	1	−1	2
2	0	1	−1
−1	0	3	1

в)

1	0	1	2
3	−1	−1	−1
1	−1	1	−1
0	1	2	−1
−1	0	1	2

18 Требуется заполнить массив $A[1:6, 1:6]$ из целых чисел по правилу $A[k, n] = k^n - n^k$. Изобразите схемой алгоритм, позволяющий заполнить массив нужным образом.

АЛГОРИТМ-ФУНКЦИЯ

19 Ниже приведены алгоритм T и алгоритм-функция S.

```
Алгоритм T
цел: n, k, m, t, s;
{ Запросить n;  (* n ⩾ 1 *)
  Запросить k; (* k ⩾ 1 *)
  Если (k > 9*n) то
  { Сообщить "Таких чисел нет"; }
  Иначе
  { t := 0;
      Делать от m := 10^(n — 1) до 10^n − 1
      { s := S(m);
        Если (s = k) то { t :=  t + 1;  }
      }
      Сообщить t;
  }
}

Функция S: цел (арг: цел: n)
цел:  m;
{ m := n mod 10;
  Делать пока (n > 9)
  {  n := n div 10;
     m := m + n mod 10;
  }
    знач := m;
}
```

а) Каким будет результат исполнения алгоритма T при $n = 2$ и $k = 8$? A при $n = 3$ и $k = 13$?
б) Для решения какой задачи составлены эти алгоритмы?

20 Ниже приведён алгоритм вычисления функции F.

```
Функция F: цел (арг: цел: n)
{ Запросить n;
  Если (n ⩽ 2) то
  {  знач := n;
  }
  Иначе
  {  знач := F(n − 1) + 2*F(n − 2);
  }
}
```

Какое значение получит переменная z после выполнения команды

$z := F(6)$?

ЯЗЫКИ ПРОГРАММИРОВАНИЯ

21 Определите число, которое будет напечатано в результате выполнения программы, записанной на двух языках программирования.

КуМир

```
алг
нач
  цел  n, s
  n := 0
  s := 0
  нц
  пока s <= 325
        s := s + 10
        n := n + 3
  кц
  вывод  n
кон
```

Паскаль

```
var n, s: integer;
begin
  n := 0;
  s := 0;
  while (s <= 325) do
  begin
    s := s + 10;
    n := n + 3;
  end;
  write(n)
end.
```

22 Ниже на двух языках записан алгоритм. Получив на вход число x, этот алгоритм печатает два числа: a и b. Укажите наименьшее из таких чисел x, при вводе которых алгоритм печатает сначала 13, а потом 5.

КуМир

```
алг
нач
  цел x, a, b, c
  ввод x
  a := 0;
  b := 10
  нц
  пока x > 0
    c := mod(x, 10)
    a := a + c
    если c < b
      то b := c
    все
    x := div(x, 10)
  кц
  вывод  a, b
кон
```

Паскаль

```
var x, a, b, c: integer;
begin
  readln(x);
  a := 0;
  b := 10;
  while (x > 0) do
  begin
    c := x mod 10;
    a := a + c;
    if (c < b) then b := c;
    x := x div 10;
  end;
  writeln(a);    write(b);
end.
```

23 Требовалось написать программу, которая решает неравенство $\frac{x+a}{bx} < 0$ относительно x для любого ненулевого числа b и любого неотрицательного числа a, введённых с клавиатуры. Все числа считаются действительными. Программист торопился и написал программу неправильно. Ниже она приведена на двух языках.

КуМир

```
алг
нач
вещ x, a, b
  ввод x, a, b
  если  a = 0 то
если  b > 0 то
  вывод "нет решений"
иначе
  вывод "x > 0 или x < 0"
всё
иначе
вывод −a, "< x < 0"
всё
кон
```

Паскаль

```
var x, a, b: real;
begin
 readln(x, a, b);
 if (a := 0) then
 if (b > 0) then
  write ('нет решений')
 else
  write('x > 0 или x < 0')
 else
  write(−a, '< x < 0');
end.
```

а) Приведите пример таких чисел a, b, x, при которых программа неверно решает поставленную задачу.
б) Укажите, какая часть программы является лишней.
в) Укажите, как нужно доработать программу, чтобы не было случаев её неправильной работы. (Это можно сделать несколькими способами.)

24 Значения двух массивов $A[1..100]$ и $B[1..100]$ задаются с помощью следующей программы, записанной на двух языках программирования.

КуМир

```
алг Масс
нач
 цел n
 цел таб A[1:100], B[1:100]
 A[1] :=1;
 нц для n от 2 до 100
   A[n] := n + 1 − 2 * A[n − 1]
 кц
 B[1] := A[1]
 нц для n от 2 до 100
   B[n] := B[n − 1] * A[n]
 вывод B[n]
 кц
кон
```

Паскаль

```
program mass
var n: integer;
var A: array [1..100] of
integer,
var B: array [1..100] of
integer,
begin
  A[1] :=1;
  for n:=2 to 100 do
  A[n] := n + 1 − 2 * A[n − 1];
  B[1] := A[1];
  for n:=2 to 100 do
  begin
    B[n] := B[n − 1] * A[n];
    writeln(B[n]);
  end;
end.
```

а) Определите, сколько отрицательных чисел будет напечатано в результате исполнения этой программы.

б) Измените программу так, чтобы в ней подсчитывалось количество отрицательных чисел в массиве B и это число выводилось на печать (элементы самого массива B по-прежнему тоже печатаются).

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ

Выполнять задания лабораторных работ, помещённых в этом разделе главы, вы будете в компьютерном классе. А использовать вы будете тот язык программирования, транслятор с которого имеется на ваших компьютерах. Если это КуМир или Паскаль, то помощь в освоении языка программирования вам окажут таблицы записи основных алгоритмических конструкций, которые приведены на оборотах обложки. Если вы знакомились с каким-либо другим языком (Си, Питон и др.), то воспользуйтесь справочной системой, которая есть для каждого языка программирования. И конечно, вам всегда придёт на помощь учитель. Мы советуем при решении задач (по крайней мере на первых порах) сначала записывать алгоритм на алгоритмическом языке.

ВСПОМНИТЕ:

- Каковы основные характеристики переменных
- Как описываются переменные в языках программирования
- Какие формы оператора ветвления применяются в алгоритмизации
- Как в языках программирования записывается оператор ветвления

ЛАБОРАТОРНАЯ РАБОТА № 13. ОСВОЕНИЕ ЯЗЫКА ПРОГРАММИРОВАНИЯ Начнём с совсем простого.

1 Напишите программу сложения двух чисел. Числа запрашиваются у пользователя (т.е. вводятся с клавиатуры), результаты вычисления выводятся на экран. Протестируйте работу программы на нескольких парах чисел (целых, дробных, положительных, отрицательных).

2 Напишите программу решения квадратного уравнения $ax^2 + bx + c = 0$. Коэффициенты a, b и c запрашиваются у пользователя, сообщения и результаты расчётов выводятся на экран. Протестируйте составленную вами программу для следующих начальных данных:
а) $a = 2$, $b = -3$ и $c = 1$;
б) $a = 0{,}5$, $b = 2$ и $c = 2$;
в) $a = -1$, $b = 1{,}9$ и $c = -2{,}1$;
г) $a = 0$, $b = 2$ и $c = -3$ (в этом случае на экран должно выводиться сообщение «Ваше уравнение не квадратное»).

3 Составьте программы нахождения:
а) наименьшего из двух чисел;
б) наибольшего из трёх чисел (совет: воспользуйтесь схемой алгоритма, изображённой в задании 8 на с. 111);
в) числа, расположенного между двумя из трёх заданных чисел.
Числа запрашиваются у пользователя, результат выводится на экран. Протестируйте составленные программы для различных наборов начальных данных.

4 Напишите программу, которая для трёх заданных отрезков определяет, могут ли они быть сторонами треугольника. Длины отрезков запрашиваются у пользователя, результат выводится на экран.

Протестируйте вашу программу для следующих начальных данных (здесь через a, b и c обозначены длины отрезков):
а) $a = 2{,}2$, $b = 6{,}9$ и $c = 5{,}4$;
б) $a = 7$, $b = 2$ и $c = 4$;
в) $a = 3{,}3$, $b = 1{,}9$ и $c = 5{,}2$.

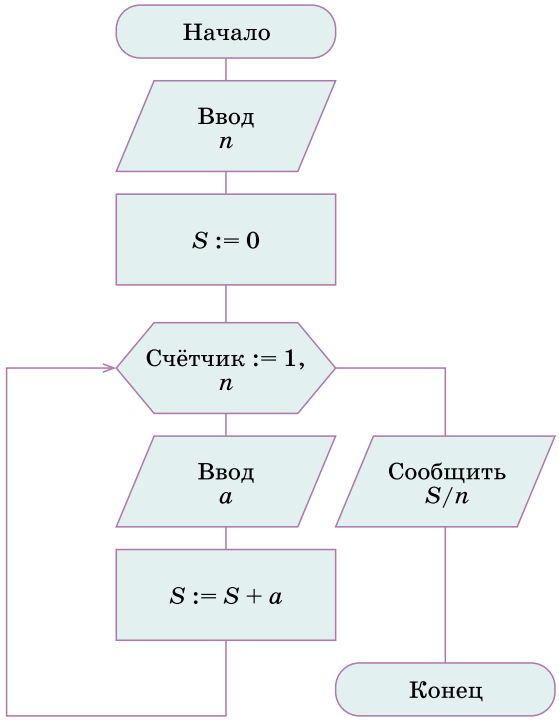
Лабораторная работа завершена.

ЛАБОРАТОРНАЯ РАБОТА № 14. ЦИКЛЫ В ЯЗЫКАХ ПРОГРАММИРОВАНИЯ

Начнём с уже знакомого.

1 На с. 103 приведены алгоритмы и программы вычисления чисел Фибоначчи. Если вы изучаете другой язык программирования, перепишите этот алгоритм на соответствующем языке. Используя данную программу, найдите десятый член ряда Фибоначчи.

2 Нередко приходится вычислять среднее арифметическое нескольких чисел. Полезно иметь программу, которая может такое сделать. Вот схема соответствующего алгоритма:



а) Исполните этот алгоритм, чтобы найти среднее арифметическое чисел 5, −3, 2, 6.

б) Напишите программу, реализующую данный алгоритм. Протестируйте её для различных наборов чисел (например, при $n = 10$).

3 В задании 2 показан приём, позволяющий в переменной S накапливать сумму чисел. Используя этот приём, напишите программу, вычисляющую сумму чисел, обратных первым n натуральным числам. Число n запрашивается у пользователя, результат выводится на экран. Протестируйте программу для $n = 1; 2; 3; 4; 5$. Найдите сумму указанных чисел для $n = 50$.

4 Напишите программу вычисления следующей алгебраической суммы:

$$1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} - \frac{1}{6} + \frac{1}{7} - \frac{1}{8} + \ldots + \frac{(-1)^{n-1}}{n}.$$

ВСПОМНИТЕ:

- Основные виды циклических конструкций

ИССЛЕДОВАНИЯ

Мы уже говорили, что числа Фибоначчи обладают многими замечательными свойствами и находят применение в самых разных областях человеческой деятельности. Мы предлагаем вам в компьютерном эксперименте обнаружить одно из таких свойств.

- Дополните программу вычисления n-го числа a_n вычислением отношения $\frac{a_n}{a_{n-1}}$ при $n > 2$.
- Найдите значения этих отношений при $n = 10; 30; 50; 70; 90$.
- Вычислите разность между соседними значениями отношений. Обратите внимание, что она уменьшается с увеличением n.

Можно доказать, что эта разность стремится к 0 с увеличением номера n. При этом само отношение $\frac{a_n}{a_{n-1}}$ стремится к числу $\frac{1+\sqrt{5}}{2}$, которое называют золотым сечением.

- Найдите отклонение отношения $\frac{a_n}{a_{n-1}}$ от золотого сечения при $n = 100; 200; 500$.
- Золотое сечение встречается в архитектуре, живописи, биологии. Найдите в Интернете более подробную информацию об использовании золотого сечения.

Само число $\frac{1+\sqrt{5}}{2}$ иррационально, поэтому отношение $\frac{a_n}{a_{n-1}}$ используется как рациональное приближение к этому числу.

Число n запрашивается у пользователя, результат выводится на экран. Протестируйте программу для $n = 1; 2; 3; 4; 5$. Найдите указанную сумму для $n = 50$.

5 Вычисление квадратного корня — это некоторый вспомогательный алгоритм, который исполняется каждый раз, когда вы даёте команду x := sqrt (a) — ведь компьютер умеет выполнять только арифметические действия. Вычисление квадратного корня осуществляется при помощи построения последовательности приближённых значений, вычисляемых по формуле $x_n = \frac{1}{2}\left(x_{n-1} + \frac{a}{x_{n-1}}\right)$ при $n > 1$, а в качестве x_1 можно взять любое положительное число, например 1. Про последовательности, которые заданы правилом вычисления значения n-го члена по значениям предыдущих членов последовательности, говорят, что они заданы **рекуррентно**. В частности, как вы видели, рекуррентно задаётся последовательность чисел Фибоначчи.
Вот алгоритм приближённого вычисления квадратного корня из числа a с помощью данного рекуррентного соотношения.

```
Алгоритм
вещ A, X;
цел N;
{ Запросить A;
  Если (A < 0) то
  { Сообщить "Корень не извлекается";  }
  Иначе
  { X := 1; (* Начальное приближённое значение корня *)
    N = 0;
    Делать пока ( |A – X*X| > 0,0001)
    { X := 0,5*(X + A/X);
      N := N + 1;
      Сообщить N;
      Сообщить X; (* N-е приближение корня *)
    }
  }
}
```

а) Напишите программу, реализующую данный алгоритм. Протестируйте её для различных A (например, для $A = 4; 9; 25$). На каком шаге для каждого из указанных значений A программа заканчивала работу?

б) Найдите значения $\sqrt{2}$, $\sqrt{5}$, $\sqrt{13}$.

ЛАБОРАТОРНАЯ РАБОТА № 15. МАССИВЫ В ЯЗЫКАХ ПРОГРАММИРОВАНИЯ На с. 101 приведён алгоритм поиска наибольшего элемента в двумерном массиве. Обдумайте его и приступайте к решению задач.

1 а) Напишите и отладьте программу заполнения трёхмерного массива $A[1{:}6,\ 1{:}10,\ 1{:}6]$ по правилу

$$A[k,\ m,\ n] = (k^n - n^k)\ (m - k - n).$$

б) Напишите и отладьте программу поиска максимального элемента в массиве, который создан при помощи программы, созданной вами в результате выполнения задания пункта *а*. На экран должны выводиться значения максимального элемента и его индексов.

ВСПОМНИТЕ:

- Каковы основные характеристики массива
- Как описываются массивы в языке программирования
- Как задаются элементы в массиве

в) Напишите и отладьте программу, позволяющую определить количество положительных и отрицательных элементов в массиве A.

2 Дан одномерный массив, заполненный целыми числами. Требуется построить массив, содержащий только чётные числа из исходного массива, или сообщить, что все числа в исходном массиве нечётны. Напишите и отладьте соответствующую программу.

3 Даны 100 точек на плоскости. Будем считать, что они пронумерованы числами от 1 до 100, а их координаты размещены в двумерном массиве $M[1:2; 1:100]$, где $M[1; j]$ — абсцисса j-й точки, а $M[2; j]$ — ордината той же точки. Напишите и отладьте программу, после исполнения которой будут сообщены номера точек, наиболее удалённых друг от друга.

4 В одномерном массиве требуется найти два элемента, сумма которых максимальна. Напишите и отладьте программу, которая позволяет это сделать.

Это задание является незначительной модификацией задания ЕГЭ.

ЛАБОРАТОРНАЯ РАБОТА № 16. МЕТОД ПОШАГОВОЙ ДЕТАЛИЗАЦИИ И ПОДПРОГРАММЫ

1 На с. 104–106 проведена пошаговая детализация задачи разложения натурального числа на простые множители. Напишите подпрограммы:
а) вычисления наименьшего простого множителя в разложении натурального числа n;
б) определения степени, с которой множитель p входит в разложение натурального числа n.
Отладьте эти подпрограммы. С использованием подпрограмм, написанных в пунктах *а* и *б*, напишите программу разложения натурального числа n на простые множители.

2 Используя подпрограмму вычисления наименьшего простого множителя (см. задание 1, *а*), составьте программу подсчёта числа различных простых делителей натурального числа n.

3 Выполнив задание 4 лабораторной работы № 13, вы написали программу, определяющую, можно ли из заданных трёх отрезков построить треугольник. Оформите эту программу как функцию от трёх аргументов, результатом которой является логическое значение «Истина», если треугольник построить можно, и «Ложь» в противном случае. Используя эту функцию, напишите и отладьте программу, определяющую, можно ли из четырёх отрезков выбрать три таких, что из них можно построить треугольник. Длины отрезков вводятся с клавиатуры, ответ выводится на экран.

4 В задании 19 теоретического практикума приведены алгоритм T и алгоритм-функция S. Как вы, наверно, догадались, алгоритм-функция S вычисляет сумму цифр натурального числа, а алгоритм T подсчитывает количество n-значных натуральных чисел, имеющих заданную сумму цифр k. Напишите и отладьте соответствующие программы и функции. Совпадают ли результаты, полученные с помощью составленных вами программ, с результатами задания 19, *а*?

ВСПОМНИТЕ:

- В чём состоит метод пошаговой детализации
- Как записываются подпрограммы и функции на алгоритмическом языке и на языке программирования

ИССЛЕДОВАНИЯ

Число 11 — простое, числа 111 и 1111 — составные. А число 11111 — простое или составное? Какие числа, в записи которых используется только цифра 1, являются простыми? Предлагаем вам попытаться найти ответ на этот вопрос.

- Число простое, если оно совпадает со своим наименьшим простым делителем. Напишите программу, которая, используя подпрограмму вычисления наименьшего простого делителя, позволит обследовать числа, имеющие не более 10 цифр.
- Если количество единиц составное, то и само число будет составным. Воспользуйтесь этой информацией, чтобы оптимизировать поиск нужных чисел.
- Если подумать ещё немного, то можно понять, что у составного числа n наименьший простой делитель не превосходит $\sqrt{n}$. Воспользуйтесь этим, чтобы ещё ускорить работу программы. Попытайтесь обследовать числа, в которых от 10 до 30 цифр.

ПОДВЕДЁМ ИТОГИ

- Каждый формальный исполнитель полностью определён совокупностью объектов, которые он может обрабатывать, и системой своих допустимых действий над этими объектами. Важной характеристикой исполнителя является совокупность достижимых с его помощью целей.
- Для управления формальным исполнителем требуется алгоритм. Алгоритм, записанный на языке команд, понятных исполнителю, называется программой.
- Для составления алгоритмов используют линейный способ организации действий и конструкции ветвление и цикл.
- Для работы исполнителя с информацией используют переменные. Каждая переменная имеет имя и значение. Для работы с данными разного вида используют переменные разных типов: целые, вещественные, символьные и т.д.
- Для обработки информации, представленной в табличном виде, используются массивы. Массив — это набор однотипных данных, снабжённых некоторым количеством индексов. Количество индексов называют размерностью массива.
- Для решения сложных задач используют метод пошаговой детализации. Он состоит в том, что исходная задача разбивается на несколько более простых задач или задач, решение которых уже известно. Решение каждой задачи, возникшей в ходе пошаговой детализации, удобно оформлять в виде вспомогательного алгоритма или алгоритма функции.
- Язык программирования — это формальный язык записи алгоритмов, связующий язык записи алгоритмов для человека с внутренним языком команд компьютера.

ПОДРОБНЕЕ...

Аникеева Л. А., Гусева О. Б. Программирование на алгоритмическом языке КуМир. — М.: Солон-Пресс, 2011. — 48 с.

Гейн А. Г. Информатика и ИКТ. Задачник-практикум. 10—11 классы. — М.: Просвещение, 2019. — 157 с.

Лещинер В. Р., Крылов С. С., Ушаков Д. М. Я сдам ЕГЭ! Информатика. Типовые задания. — М.: Просвещение, 2019. — 240 с.

Филиппов С. А. Робототехника для детей и родителей. — СПб.: Наука, 2013. — 319 с.

Энциклопедия школьной информатики. — М.: БИНОМ: Лаборатория знаний, 2014. — 400 с.

ВОПРОСЫ ДЛЯ ОБСУЖДЕНИЯ:

- Верно ли, что только человек не является формальным исполнителем?
- Есть точка зрения, что в любой своей деятельности человек хотя бы частично ведёт себя как формальный исполнитель, т.е. следуя определённым инструкциям и правилам. Согласны ли вы с ней? Если не согласны, то приведите подходящие примеры. Если согласны, то в чём состоят положительные стороны этого явления и в чём отрицательные?
- Представляют ли роботы опасность для человека?

Язык КуМир:

https:/www.niisi.ru/kumir/

Федеральный центр информационно-образовательных ресурсов (ФЦИОР):

http://fcior.edu.ru/

ТОЛКОВЫЙ СЛОВАРЬ ТЕРМИНОВ И ПОНЯТИЙ ИНФОРМАТИКИ

А

Алгоритм — организованная последовательность действий, допустимых для некоторого исполнителя

Алгоритмическая конструкция — способ организации действий в алгоритме. Основными алгоритмическими конструкциями являются ветвление, цикл и вспомогательный алгоритм

Б

Байт — последовательность восьми битов

Бит — 1) двоичный знак; 2) количество информации, вдвое уменьшающее неопределённость ситуации

Браузер — программа просмотра гипертекстовых страниц

В

Ветвление — алгоритмическая конструкция, предусматривающая выполнение одного блока действий, если выполнено заданное условие, и другого блока действий, если условие не выполнено. Второй блок действий может отсутствовать; тогда говорят, что имеет место ветвление в неполной форме

Вспомогательный алгоритм — алгоритм, позволяющий решить часть более общей задачи.

Г

Гиперссылка — связь между различными фрагментами текста, позволяющая осуществлять автоматический переход между этими фрагментами

Гипертекст — текст, в котором реализована возможность автоматического перехода от одного фрагмента текста к другому

Графический редактор — программа, позволяющая создавать и редактировать графические объекты на экране компьютера.

Д

Данные — информация, представленная в формализованном виде, предназначенная для обработки человеком или каким-либо автоматизированным устройством

Детерминированность алгоритма — однозначная определённость преобразования объектов исполнителем, полученных на предшествующих шагах алгоритма

Дискретизация — замена непрерывно меняющейся величины её отдельными значениями

З

Знания — информация в обобщённом и систематизированном виде, готовая к применению в схожих ситуациях

И

Информатика — наука о закономерностях информационных процессов и средствах получения, сохранения, передачи и преобразования информации, а также сфера человеческой деятельности, связанной с информационными процессами

Информационный объект — информация, зафиксированная тем или иным способом на некотором носителе

Информационный процесс — это процесс, в ходе которого изменяется содержание информации или форма ее представления. Основные информационные процессы: получение, сохранение, передача и преобразование (обработка) информации

Исполнитель — объект или субъект, предназначенный для исполнения алгоритма. Исполнитель может быть формальным, т.е. исполняющим алгоритм без осмысления действий, входящих в алгоритм, и эвристическим, т.е. осмысливающим выполняемые действия и принимающим решения по их исполнению. Исполнитель характеризуется набором допустимых для него действий и достижимых целей

К

Конечность алгоритма — свойство алгоритма, состоящее в том, что для любых допустимых исходных данных алгоритм заканчивает работу за конечное число шагов, вообще говоря, зависящее от исходных данных

М

Массив — совокупность однотипных переменных, снабжённых одной или несколькими системами индексов

Массовость алгоритма — свойство алгоритма, заключающееся в возможности использования алгоритма для решения однотипных задач при разных исходных данных

О

Операторные скобки — скобки, указывающие совокупность действий, которые требуется исполнить в операторе ветвления, операторе цикла или во вспомогательном алгоритме. В ряде языков программирования операторные скобки записываются служебными словами (например, begin, end в языке Паскаль)
Отладка программы — поиск ошибок в программе с помощью тестов
Ошибка синтаксическая — ошибочная запись команды в программе
Ошибка семантическая — попытка применения действия в алгоритме к недопустимым для этого действия данным

П

Пиксель — наименьший элемент изображения на экране компьютера
Подпрограмма — вспомогательный алгоритм, записанный на формальном языке исполнителя
Программа — алгоритм, записанный на формальном языке исполнителя

Р

Размерность массива — количество индексов, которые имеет каждый элемент массива
Разрешение графического режима — количество пикселей на экране
Растр — прямоугольная сетка пикселей на экране компьютера
Робот — автоматическое устройство, имеющее датчики для получения информации из окружающей среды и программируемое устройство управления, позволяющее на основании получаемой от датчиков информации вырабатывать управляющие воздействия для своего поведения

С

Система счисления — один из способов записи чисел с помощью конечного количества цифр
Слайд-фильм — последовательность слайдов, для которой реализована возможность автоматического показа

Т

Текстовый редактор — компьютерная программа, позволяющая автоматизировать работу по созданию, редактированию и форматированию текстов, вставлять в него графические и другие информационные объекты
Тестирование компьютерной программы — запуск программы на исполнение для специально подготовленных исходных данных (тестах), для которых известно, какой результат должен быть получен. Тестирование — один из важных инструментов отладки программы
Тетрада — четыре записанные подряд цифры числа, представленного в двоичной системе счисления
Тип массива — единый тип всех переменных, составляющих данный массив

Ф

Форматирование текста — задание или изменение параметров текста, определяющих его расположение на странице
Формализация — закрепление единственного значения за используемым словом

Ц

Цикл — алгоритмическая конструкция, позволяющая организовать повторение исполнения блока действий, пока выполнено заданное условие

Я

Язык программирования — формальный язык записи алгоритмов

ASCII — Американский стандартный код информационного обмена, восьмибитный код для кодирования символов

HSB-модель — описание цвета тремя параметрами: оттенок, насыщенность, яркость

HTML — язык гипертекстовой разметки, редактор создания гипертекстовых страниц

RGB-кодирование — кодирование цвета значениями трёх параметров, показывающих доли основных цветов — красного, зелёного и синего — в кодируемом цвете

UNICODE — универсальный код кодирования символов, обычно используется двухбайтовая версия этого кода

ПРЕДМЕТНО-ТЕМАТИЧЕСКИЙ УКАЗАТЕЛЬ

Адрес ячейки 40
адресация абсолютная 42
– относительная 42
алгоритм 90
– вспомогательный 104
– линейный 92
алгоритм-функция 109
алфавит 10
анимация 59
аргумент вспомогательного алгоритма 107

Байт 12
бит 12
блок ячеек 42
браузер 39

Ветвление 92
– в неполной форме 93
– в полной форме 93
Выравнивание по левому краю 33
– по правому краю 33
– по центру 33
– по ширине 33
вызов вспомогательного алгоритма 107

Гарнитура 34
гигабайт 12
гиперссылка 38
гистограмма 44
грамматика языка 11

Детерминированность алгоритма 90
диаграмма круговая 44
– столбчатая 44
диапазон изменения индексов 101
дискретизация 14
дискретность алгоритма 90
длина слова 10

Ёмкость символа информационная 23

Заголовок цикла 94
знания 7
значение переменной 96

Имя вспомогательного алгоритма 106
– массива 100
– переменной 96
индекс элемента массива 100
интервал межстрочный 33
информация актуальная 7
– достоверная 7
– избыточная 7
– мультимедийная 8, 58
– объективная 7
– полезная 7
– полная 7
– символьная 8
исполнитель 9, 88
исполнители равносильные 89
– эквивалентные 89

Квантование звука 15
кегль 34
комментарий 94
килобайт 12
колонтитул 33
конечность алгоритма 91
контраст 57
круг Манселла 57

Логотип 58

Маркер 34
массив 100
– двумерный 100
– линейный 100
– одномерный 100
массовость алгоритма 95
мегабайт 12
метод дихотомии 21
– наименьших квадратов 49
– нисходящего проектирования 104
– пошаговой детализации 104

Насыщенность 57

Обработка информации 9
– – формальная 88
– – эвристическая 88
объект информационный 8
объявление переменной 97
оператор ветвления 92
– присваивания 96
– условный 91
– цикла 94
операция логическая 102
– И 102
– ИЛИ 102
– НЕ 102
отладка 90
ошибка семантическая 90
– синтаксическая 90

Параметры фактические 107
– формальные 107
переменная 96
– локальная 108
пиксель 50
понятность алгоритма 91
приемник информации 22
примитив графический 52
программа 90
процесс информационный 8
пункт 33

Размерность массива 100
растр 50
редактирование текста 32
редактор графический 50
результат вспомогательного алгоритма 107
результативность алгоритма 91
робот 88

Семантика языка 11
сигнал аналоговый 14
синтаксис языка 11
система допустимых действий 89
– команд 89
– счисления 16
– двоичная 16
– непозиционная 16
– позиционная 16
скобки операторные 93
слайд-фильм 58
слой изобразительный 55
– корректирующий 55
слова зарезервированные 93
– служебные 93
слово 10
строка висячая 33
– красная 33
схема алгоритма 93
счётчик цикла 99

Тело алгоритма 98
– цикла 94
терабайт 12
тетрада 19
тип булев 97
– вещественный 97
– логический 97
– массива 100
– переменной 97
– символьный 97
– целый 97

Формализация 11
форматирование 34

Цель достижимая 89
цвет дополнительный 57
цикл 94
– вложенный 94
– внешний 94
– внутренний 94
– с предисловием 94
– со счётчиком 99

Частота дискретизации 15

Шкала яркости 14
шрифт 34
– наклонный 34
– подчёркнутый 34
– полужирный 34
– рубленный 34
– с засечками 34

Язык 10
– естественный 10
– коммуникативный 10
– КуМир 102
– Паскаль 102
– программирования 102
– формализованный 11
– формальный 11
яркость 56
ячейка электронной таблицы 40

ActiveX 37
ASCII 12

HSB-модель 57
HTML 39

RGB-кодирование 14

UNICODE 12

СОДЕРЖАНИЕ

Предисловие 3
Работаем с учебником 4

ГЛАВА 1. ИНФОРМАЦИЯ И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ
1. Информация и её свойства 6
2. Информационные процессы 8
3. Язык как средство сохранения и передачи информации 10
4. Универсальность двоичного кодирования 12
5. Кодирование числовой информации 16
6. Перевод целых чисел из одной системы счисления в другую 18
7. Метод дихотомии. Измерение количества информации 20
Теоретический практикум 24
Компьютерный практикум 26
Подведём итоги 30

ГЛАВА 2. ОСНОВНЫЕ ИНФОРМАЦИОННЫЕ ОБЪЕКТЫ. ИХ СОЗДАНИЕ И КОМПЬЮТЕРНАЯ ОБРАБОТКА
8. Компьютерная обработка текста 32
9. Вставка объектов в текст документа 36
10. Гипертекст 38
11. Организация вычислений при помощи компьютера 40
12. Построение графиков и диаграмм с помощью электронной таблицы 44
13. Применение электронной таблицы в задачах проектирования 46
14. Применение электронной таблицы в экономических задачах 48
15. Создание и компьютерная обработка визуальных информационных объектов 50
16. Тонкая обработка изображений 54
17. Цветовые модели 56
18. Мультимедийные презентации 58
Теоретический практикум 60
Компьютерный практикум 62
Подведём итоги 86

ГЛАВА 3. АЛГОРИТМЫ И ИСПОЛНИТЕЛИ. ОСНОВЫ ПРОГРАММИРОВАНИЯ
19. Формальные исполнители 88
20. Алгоритмы и программы 90
21. Линейные и разветвляющиеся алгоритмы 92
22. Циклическое исполнение алгоритма 94
23. Переменные в алгоритмах 96
24. От переменной к массиву 100
25. Знакомство с языками программирования 102
26. Метод пошаговой детализации, вспомогательные алгоритмы и подпрограммы 104
Теоретический практикум 110
Компьютерный практикум 118
Подведём итоги 122

Толковый словарь терминов и понятий информатики ... 123
Предметно-тематический указатель 125

Учебное издание

Серия «Сферы»

Гейн Александр Георгиевич
Юнерман Нина Ароновна

ИНФОРМАТИКА

10 класс

Базовый уровень
Учебник

Центр развития углублённого и профильного образования,
функциональной грамотности, технологии и ИКТ-компетенций

Ответственный за выпуск *Л. В. Кузнецова*
Редактор *Л. В. Кузнецова*
Художественный редактор *Т. В. Глушкова*
Фотографии: *Фотобанк Лори*, *DIOMEDIA*, © «*РИА Новости*»
Компьютерная графика *Н. А. Артемьевой*
Компьютерная вёрстка *Д. В. Шейкиной*
Корректоры *Н. В. Белозёрова*, *Г. Н. Смирнова*, *И. В. Чернова*

Подписано в печать 16.02.2022. Формат 84×108/16. Гарнитура Школьная.
Усл. печ. л. 13,44. Уч.-изд. л. 11,34. Тираж экз. Заказ № .

Акционерное общество «Издательство «Просвещение».
Российская Федерация, 127473, г. Москва, ул. Краснопролетарская, д. 16,
стр. 3, этаж 4, помещение I.

Адрес электронной почты «Горячей линии» — **vopros@prosv.ru.**

Язык записи алгоритмов	Pascal
1. Заголовок программы	
Алгоритм {...}	**program** <имя>**;**
2. Описание переменных	
	var *(*Обязательно перед описанием переменных*)*
вещественные переменные	
вещ: <список переменных>;	<список переменных>**: real;**
целочисленные переменные	
цел: <список переменных>;	<список переменных>**: integer;**
символьные переменные	
сим: <список переменных>;	<список переменных>**: string;**
массивы	
вещ: <имя массива> [1:100; 1:100];	<имя массива>**: array** [1..100, 1..100] **of real;**
3. Оператор условия	
Если (<условие>) **то** {<оператор>; ...} **иначе** {<оператор>; ...}	**if** <условие> **then begin** <оператор>; ... **end** **else begin** <оператор>; ... **end;**
Конструкция **иначе** может отсутствовать	
4. Оператор цикла (else) по условию	
Делать пока (<условие>) {<оператор>; ...}	**while** <условие> **do** **begin** <оператор>; ... **end;**
5. Оператор цикла со счётчиком	
Делать от <переменная> := <выражение> **до** <переменная> **с шагом** <выражение> {<оператор>; ...}	**for** <переменная> := <выражение> **to** <выражение> **do** **begin** <оператор>; ... **end;** Если нужен шаг –1, то вместо **to** пишут **downto**
6. Операторы ввода и вывода информации	
Запросить <имя переменной>;	**readln** (<список переменных>);
Сообщить <имя переменной или/и сообщение в кавычках>;	**writeln** ('<сообщение>',<список переменных>);
7. Подпрограммы и функции	
Подпрограмма <имя подпрограммы> (**арг:** <список параметров>, **рез:** <список параметров>) {<оператор>; ...}	**Procedure** <имя> (<параметры>**:** <тип>; var <результаты>: <тип>); **begin** <оператор>; ... **end**;
Функция <имя>: (**арг:** <типы аргументов>: <список переменных>): <тип> { <действие>; ... **знач** := <результат>; }	**function** <имя> (<параметры>**:** <тип>)**:** <тип>; **begin** <оператор>; ... <имя функции> := <результат>; **end**;
8. Обращение к подпрограмме	
Вызвать <имя подпрограммы> (**арг**: <список параметров>; **рез**: <список параметров>);	<имя> (<список параметров>);
9. Обращение к функции	
<переменная> := <имя функции> (<список фактических параметров>);	<переменная> := <имя функции> (<список фактических параметров>);

Язык записи алгоритмов	КуМир
1. Заголовок программы	
Алгоритм {...}	**алг** <имя> **нач** ... **кон**
2. Описание переменных	
вещественные переменные	
вещ: <список переменных>;	**вещ**: <список переменных>
целочисленные переменные	
цел: <список переменных>;	**цел**: <список переменных>
символьные переменные	
сим: <список переменных>;	**сим**: <список переменных>
массивы	
вещ: <имя массива> [1:100; 1:100];	**вещ таб** <имя массива> [1:100, 1:100]
3. Оператор условия	
Если (<условие>) **то** { <оператор>; ... } **иначе** { <оператор>; ... }	**Если** <условие> **то** <оператор> ... **иначе** <оператор> ... **все**
Конструкция **иначе** может отсутствовать	
4. Оператор цикла по условию	
Делать пока (<условие>) { <оператор>; ... }	**нц** **пока** <условие> <оператор> ... **кц**
5. Оператор цикла со счётчиком	
Делать от <переменная> := <выражение> **до** <переменная> **с шагом** <выражение> { <оператор>; ... }	**нц для** <переменная> **от** <выражение> **до** <выражение> <оператор> ... **кц**
6. Операторы ввода и вывода информации	
Запросить <имя переменной>;	**Ввод** <список переменных через запятую >
Сообщить <имя переменной или/и сообщение в кавычках>;	**Вывод** <список переменных или/и сообщение в кавычках>
7. Подпрограммы и функции	
Подпрограмма <имя подпрограммы>; (**арг:** <список параметров>, **рез:** <список параметров>) { <оператор>; ... }	**алг** <имя> **(арг**: <параметры>, **рез**: <параметры>**)** **нач** <описание переменных> <оператор> ... **кон**
Функция <имя>: (**арг:** <типы аргументов>: <список переменных>): <тип> { <действие>; ... **знач** := <результат>; }	**алг** <тип> <имя> **(арг**: <тип> <параметры>**)** **нач** <описание переменных> <оператор> ... **знач** := <результат> **кон**
8. Обращение к подпрограмме	
Вызвать <имя подпрограммы> (**арг**: <список параметров>; **рез**: <список параметров>);	<имя алгоритма> **(**<список параметров и результатов>**)**
9. Обращение к функции	
<переменная> := <имя функции> (<список фактических параметров>);	<переменная> := <имя функции> **(**<список параметров>**)**;