Н. Культин

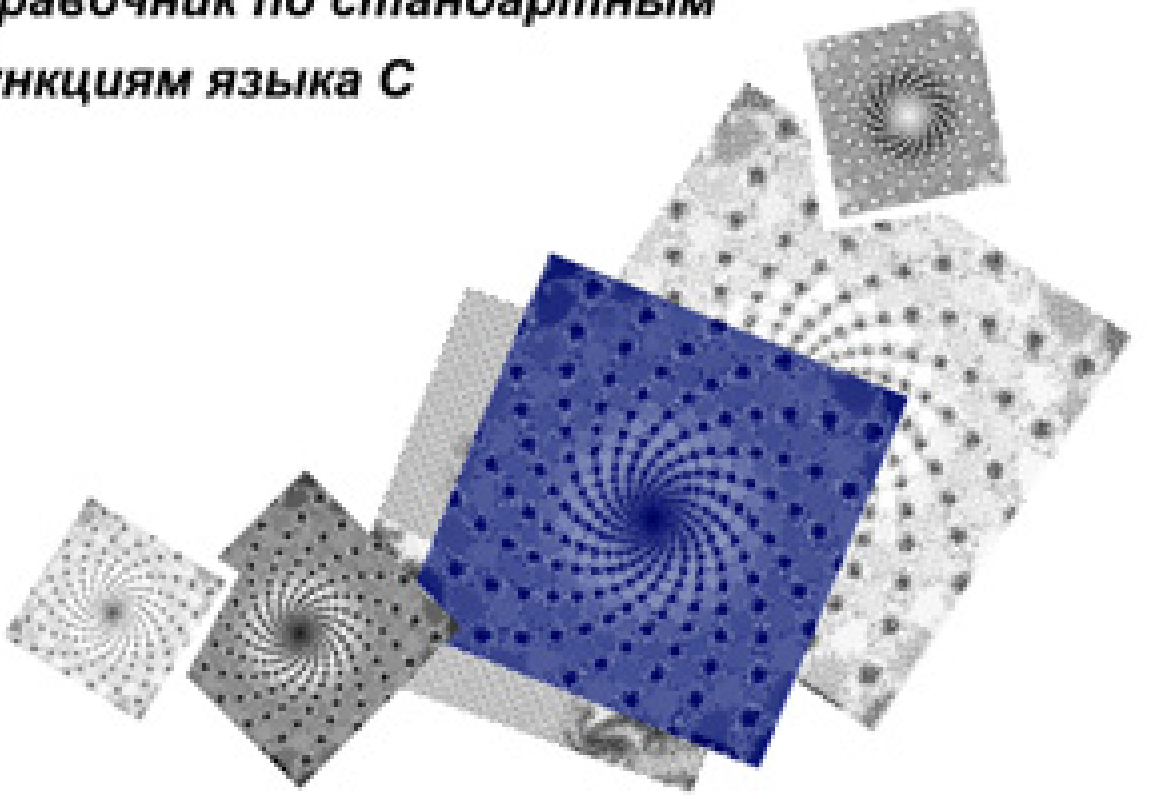
bhv
www.bhv.ru
www.bhv.kiev.ua

# C/C++ в задачах и примерах

Более 250 задач

*Примеры*
*и исходные тексты*
*программ с комментариями*

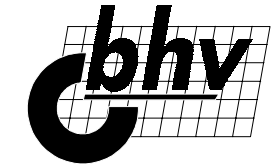
*Справочник по стандартным*
*функциям языка С*



**Никита Культин**

# C/C++

## в задачах и примерах




Дюссельдорф ◆ Киев ◆ Москва ◆ Санкт-Петербург

УДК 681.3.06

Сборник задач по программированию на языке C/C++, как типовых — ввод-вывод, управление вычислительным процессом, работа с массивами, поиск и сортировка, так и тех, которые чаще всего не входят в традиционные курсы — работа со строками и файлами, вывод на принтер, деловая графика, рекурсия. Для большинства задач приведены решения, представляющие собой документированные исходные тексты программ. Книга содержит также справочник по наиболее часто используемым функциям языка C/C++ и может служить задачником для студентов и школьников, изучающих программирование.

*Для начинающих программистов*

**Группа подготовки издания:**

| | |
|---|---|
| Главный редактор | *Екатерина Кондукова* |
| Зав. редакцией | *Наталья Таркова* |
| Редактор | *Владимир Овчинников* |
| Компьютерная верстка | *Натальи Смирновой* |
| Корректор | *Наталия Першакова* |
| Дизайн обложки | *Ангелины Лужиной* |
| Зав. производством | *Николай Тверских* |

**Культин Н. Б.**
C/C++ в задачах и примерах. — СПб.: БХВ-Петербург, 2001. — 288 с.: ил.

ISBN 5-94157-029-5



Лицензия ИД № 02429 от 24.07.00. Подписано в печать 05.02.01.
Формат 60×90 $^{1}/_{16}$. Печать офсетная. Усл. печ. л. 18.
Тираж 5000 экз. Заказ №
"БХВ-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар, № 77.99.1.953.П.950.3.99 от 01.03.1999 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с диапозитивов
в Академической типографии "Наука" РАН.
199034, Санкт-Петербург, 9-я линия, 12.

# Содержание

# Предисловие

Чтобы научиться программировать, недостаточно прочитать книгу, посвященную языку программирования, надо писать программы, решать конкретные проблемы. Но где их найти? В учебниках, как правило, приводятся типовые, стандартные задачи, в основе которых лежит расчет по формулам. Это, несомненно, полезно, но не всегда интересно.

В книге, которую вы держите в руках, начинающему программисту предлагаются задачи, которые, с одной стороны, ему по плечу, с другой — полезны и занимательны.

Состоит книга из трех частей и приложения.

В первой части собраны задачи. Они сгруппированы по темам и охватывают практически все разделы базового курса программирования: от объявления переменных и программ с линейной структурой до работы с графикой и файлами.

Вторая часть содержит решения, представленные в виде хорошо документированных текстов программ, изучение которых, несомненно, будет полезно для начинающего программиста даже в том случае, если задача решена им самостоятельно.

Третья часть представляет собой справочник по языку программирования C++ и содержит описание наиболее часто используемых функций.

О компиляторе. Разрабатывая программу, программист ориентируется на ту или иную среду разработки, компилятор. Программировать на C++ можно как в среде Windows, так и в DOS, причем для каждой из операционных систем существует довольно большое количество средств разработки: от компилятора, работающего в режиме командной строки DOS, до мощной интерактивной интегрированной среды разработки. На каком инструменте остановить свой выбор? Если вы опытный программист (хотя это маловероятно, тогда вы не держали бы в руках эту книгу) и собираетесь разрабатывать программу по Windows, то безусловно выберите Microsoft Visual C++ или Borland C++ Builder. Если вы только осваиваете язык программирования, то следует сосредоточиться именно на языке, его возможностях, особенностях. Для решения этой задачи наилучшим образом подходит предназначенная для работы в DOS интегрированная среда разработки Bor-

land C++ Version 3.1, которая, безусловно, может быть запущена и из Windows. Следует особо обратить внимание на то, что хотя Borland C++ и был создан довольно давно, но это профессиональная, высокоэффективная среда разработки, которая наилучшим образом подходит для изучения C++.

Еще раз повторю, что научиться программировать можно только программируя, решая конкретные задачи. Поэтому, чтобы получить максимальную пользу от книги, вы должны работать с ней активно. Решайте задачи. Изучайте приведенные решения. Вводите их в свой компьютер. Не бойтесь экспериментировать — вносите изменения в программы. Чем больше вы сделаете самостоятельно, тем большему вы научитесь!

## Как работать с книгой

Группы задач следуют в книге в том порядке, в котором традиционно изучаются соответствующие разделы в курсе программирования. Перед тем как приступить к решению задач, нужно изучить соответствующую тему — прочитать раздел учебника. Если сразу решить задачу не получается, то можно посмотреть решение и затем еще раз попытаться решить задачу самостоятельно. Писать программу лучше сначала на бумаге, а уже затем вводить программу в компьютер.

Задача считается решенной, если написанная программа работает так, как сказано в условии задачи.

## Оформление решений

Важно, чтобы решенная задача была правильно оформлена. Это предполагает:

- использование несущих смысловую нагрузку имен переменных, констант и функций;
- применение отступов при записи инструкций программы;
- использование комментариев.

Правильно оформленную программу легче отлаживать, кроме того, она производит хорошее впечатление.

Приведенные в книге решения задач можно рассматривать как образцы правильного оформления.

# ЧАСТЬ I. ЗАДАЧИ

## Объявление переменных

### Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- каждая переменная программы должна быть объявлена;
- объявления переменных обычно помещают в начале функции, сразу за заголовком. Следует обратить внимание, что хотя язык C++ допускает объявление переменных практически в любом месте функции, объявлять переменные лучше все-таки в начале функции, снабжая инструкцию объявления кратким комментарием о назначении переменной;
- инструкция объявления переменной выглядит так:

  ```
  Тип ИмяПеременной;
  ```

- инструкцию объявления переменной можно использовать для инициализации переменной. В этом случае объявление переменной записывают следующим образом:

  ```
  Тип ИмяПеременной = НачальноеЗначение;
  ```

- в имени переменной можно использовать буквы латинского алфавита и цифры (первым символом должна быть буква);
- компилятор C++ различает прописные и строчные буквы, поэтому, например, имена `Summa` и `summa` обозначают разные переменные;
- основными числовыми типами языка C++ являются: `int` (целый) и `float` (дробный).
- после инструкции объявления переменной рекомендуется указывать назначение переменной.

### Задачи

**1.** Объявите переменные, необходимые для вычисления площади прямоугольника.

**2.** Объявите переменные, необходимые для пересчета веса из фунтов в килограммы.

**3.** Определите исходные данные и объявите переменные, необходимые для вычисления дохода по вкладу.

**4.** Объявите переменные, необходимые для вычисления площади круга.

**5.** Объявите переменные, необходимые для вычисления площади кольца.

**6.** Объявите переменные, необходимые для вычисления объема и площади поверхности цилиндра.

**7.** Объявите переменные, необходимые для вычисления стоимости покупки, состоящей из нескольких тетрадей, карандашей и линейки.

**8.** Объявите переменные, необходимые для вычисления стоимости покупки, состоящей из нескольких тетрадей и такого же количества обложек.

## Инструкция присваивания

### Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- инструкция присваивания предназначена для изменения значений переменных, в том числе и для вычислений "по формуле";
- в отличие большинства языков программирования, в C++ инструкция присваивания, выполняющая некоторое действие, может быть записана несколькими способами, например, вместо `x=x+dx` можно записать `x+=dx`, а вместо `i=i+1` воспользоваться оператором инкремента и записать `i++`;
- значение выражения в левой части инструкции присваивания зависит от типа операндов и операции, выполняемой над операндами. Целочисленное сложение и вычитание выполня-

ется без учета перепонения. Например, если пременная `n`, объявленная как `int`, имеет значение `32767`, то в результате выполнения инструкции `n=n+1`, значение переменной `n` будет равно `-32768`;

- результатом выполнении операции деления над целыми операндами является целое, которое получается отбрасыванием дробной части результата деления.

## Задачи

**9.** Запишите инструкцию, которая присваивает переменной `x` значение -1,5.

**10.** Запишите инструкцию, которая присваивает переменной `summa` нулевое значение.

**11.** Запишите инструкцию, которая увеличивает на единицу значение переменной `n`.

**12.** Запишите инструкцию, которая уменьшает на два значение переменной `counter`.

**13.** Запишите инструкцию вычисления среднего арифметического переменных `x1` и `x2`.

**14.** Запишите в виде инструкции присваивания формулу вычисления значения функции $y= -2,7x^3+0,23x^2-1,4$.

**15.** Запишите инструкцию, которая увеличивает значение переменной `x` на величину, находящуюся в переменной `dx`.

**16.** Запишите в виде инструкции присваивания формулу пересчета веса из фунтов в килограммы (один фунт это 405,9 грамма).

**17.** Запишите в виде инструкции присваивания формулу пересчета расстояния из километров в версты (одна верста — это 1066,8 м).

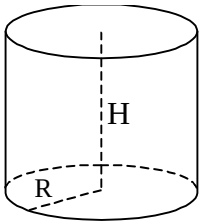
**18.** Запишите в виде инструкции присваивания формулу вычисления площади прямоугольника.

**19.** Запишите в виде инструкции присваивания формулу вычисления площади треугольника: $s = \frac{1}{2} \cdot a \cdot h$, где $a$ — длина основания, $h$ — высота треугольника.

**20.** Запишите в виде инструкции присваивания формулу вычисления площади трапеции: $s = \frac{a+b}{2} \cdot h$, где $a$ и $b$ — длины оснований, $h$ — высота трапеции.

**21.** Запишите в виде инструкции присваивания формулу вычисления площади круга: $s = \pi \cdot r^2$.

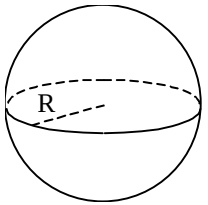
**22.** Запишите в виде инструкции присваивания формулы вычисления площади поверхности и объема цилиндра.




$$s = 2 \cdot \pi \cdot r(h + r)$$
$$v = \pi \cdot r^2 \cdot h$$

**23.** Запишите в виде инструкции присваивания формулу вычисления объема параллелепипеда.

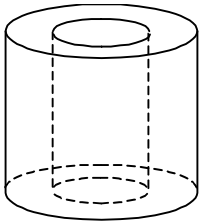
**24.** Объявите необходимые переменные и запишите в виде инструкции присваивания формулы вычисления объема и площади поверхности шара.




$$v = \frac{3}{4} \cdot \pi \cdot r^3$$
$$s = 4 \cdot \pi \cdot r^2$$

**25.** Запишите в виде инструкции присваивания формулу вычисления объема цилиндра.

**26.** Запишите в виде инструкции присваивания формулу вычисления объема полого цилиндра.



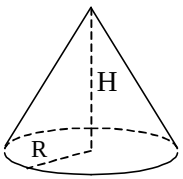
$$v = \pi \cdot h \cdot (r_1^2 - r_2^2)$$

$r_1$ - радиус цилиндра

$r_2$ — радиус отверстия

$h$ - высота цилиндра

**27.** Запишите в виде инструкции присваивания формулу вычисления объема конуса.




$$s = \frac{1}{3} \cdot \pi \cdot r^2 \cdot h$$

**28.** Запишите в виде инструкции присваивания формулу вычисления объема цилиндра.

**29.** Запишите в виде инструкции присваивания формулу вычисления тока, по известным значениям напряжения и сопротивления электрической цепи.

**30.** Запишите в виде инструкции присваивания формулу вычисления сопротивления электрической цепи по известным значениям напряжения и силы тока.

**31.** Запишите в виде инструкции присваивания формулу вычисления сопротивления электрической цепи, состоящей из трех последовательно соединенных резисторов.

**32.** Запишите в виде инструкции присваивания формулу вычисления сопротивления электрической цепи, состоящей из двух параллельно соединенных резисторов: $r = \frac{r_1 \cdot r_2}{r_1 + r_2}$.

**33.** Запишите в виде инструкции присваивания формулу пересчета сопротивления электрической цепи из омов в килоомы.

**34.** Объявите необходимые переменные и запишите в виде инструкции присваивания формулу вычисления стоимости покупки, состоящей из нескольких тетрадей, обложек к ним и карандашей.

**35.** Объявите необходимые переменные и запишите в виде инструкции присваивания формулу вычисления стоимости покупки, состоящей из помидоров, огурцов и нескольких пучков укропа.

# Вывод

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- функция `printf` обеспечивает вывод на экран монитора сообщений и значений переменных;
- первым параметром функции `printf` является строка вывода, определяющая выводимый текст и формат выода значений переменных, имена которых указаны в качестве остальных параметров функции;
- формат вывода значений переменных задается при помощи спецификатора преобразования — последовательности символов, начинающейся с символа `%`;
- при выводе числовых значений наиболее часто используются следующие спецификаторы: `%i` — для вывода целых со знаком `%u` — для вывода беззнаковых целых `%f` — для вывола дробных, в виде числа с плавющей точкой `%`*`n`*`.`*`m`*`f` — для вывода дробных в формате с фиксированной точкой, где *`n`* — количество цифр целой части, *`m`* — дробной;
- некоторые символы могут быть помещены в строку вывода только как последовательность других, обычных символов: `\n` — новая строка, `\t` — табуляция, `\"` — двойная кавычка, `\\` — символ \;
- наряду с функцией `printf`, для вывода на экран сообщений можно использовать функцию `puts`, которая после вывода текста автоматически переводит курсор в начало следующей строки;

- чтобы сразу после окончания работы программы окно, в котором программа работала, не было автоматически перекрыто другим окном, например окном редактора текста среды разработки или панелями Norton Commander, в конец программы нужно вставить следующие две инструкции:

```
printf("Для завершения нажмите клавишу <Enter>");
getch();
```

## Задачи

**36.** Написать программу, которая выводит на экран Вашу имя и фамилию.

**37.** Написать программу, которая выводит на экран путь к файлу stdio.h.

**38.** Написать программу, которая выводит на экран четверостишие:

```
Унылая пора! Очей очарованье!
Приятна мне твоя прощальная краса —
Люблю я пышное природы увяданье,
В багрец и золото одетые леса.
                А. С. Пушкин
```

**39.** Написать инструкцию вывода значений переменных `a, b` и `c` (типа `float`) с пятью цифрами целой части и тремя — дробной, в виде:

```
a = значение  b=значение  c=значение
```

**40.** Написать инструкцию вывода значений переменных `h` и `l` (типа `float`), которые содержат значения высоты и длины прямоугольника. Перед значением переменной должен быть пояснительный текст (`высота=`, `ширина=`), а после — единица измерения (`см`).

**41.** Записать инструкцию, которая выводит в одной строке значения переменных `a`, `b` и `c` целого типа (`int`).

**42.** Написать инструкцию вывода значений целых переменных `a`, `b` и `c`. Значение каждой переменной должно быть выведено в отдельной строке.

**43.** Написать инструкции вывода значений дробных переменных `x1` и `x2`. На экране перед значением переменной должен быть выведен поясняющий текст, представляющий собой имя переменной, за которым следует знак "равно".

## Факультатив

- Чтобы иметь возможность выводить на экран текст разным цветом, надо использовать функции `cprintf` и `cputs`. Следует обратить внимание на то, что переход к новой строке в функциях `cprintf` и `cputs` задается последовательностью `\n\r`.
- Цвет символов, выводимых функциями `cprintf` и `cputs`, устанавливает функция `textcolor(`*`Цвет`*`)`.
- Цвет фона устанавливает функция `textbackground(`*`Цвет`*`)`.
- Цвет можно задать при помощи целой или именованной константы.
- Чтобы использовать функции `clrscr`, `textcolor` и `textbackground`, в текст программы нужно включить директиву `#include <conio.h>`

## Задачи

**44.** Написать программу, которая выводит на синем фоне серыми буквами четверостишие:

```
Буря мглою небо кроет,
Вихри снежные крутя.
То как зверь она завоет,
То заплачет, как дитя.
                А.С.Пушкин
```

**45.** Написать программу, которая выводит на экран фразу: *"Каждый охотник желает знать, где сидят фазаны"*, позволяющую запомнить порядок следования цветов радуги (первая буква слова кодирует цвет: каждый — красный, охотник — оранжевый, желает — желтый, знать — зеленый, где — голубой, сидят — синий, фазаны — фиолетовый). Каждое слово фразы должно быть выведено наиболее подходящим цветом.

# Ввод

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

**1.** Для ввода исходных данных с клавиатуры предназначена функция `scanf`.

**2.** Первым параметром функции `scanf` является управляющая строка, остальные параметры — адреса переменных, значения которых должны быть введены.

**3.** Управляющая строка представляет собой заключенный в двойные кавычки список спецификаторов:

- `%i` — для ввода целых чисел со знаком
- `%u` — для ввода целых беззнаковых целых `%f` — для ввола дробных чисел
- `%c` — для ввода символа
- `%s` — для ввода строки

**4.** Использование имени переменной, а не ее адреса в качестве параметра функции `scanf` является типичной ошибкой начинающих программистов. Кстати, компилятор эту ошибку не обнаруживает.

## Задачи

**46.** Написать инструкцию, обеспечивающую ввод с клавиатуры значения переменной `radius` типа `float`.

**47.** Написать инструкции, которые обеспечивают ввод значений дробных (тип `float`) переменных `u` и `r`. Предполагается, что пользователь после набора каждого числа будет нажимать клавишу <Enter>.

**48.** Написать инструкцию, которая обеспечивает ввод значений переменных `u` и `r`. Предполагается, что пользователь будет набирать числа в одной строке.

**49.** Объявите необходимые переменные и напишите фрагмент программы вычисления объема цилиндра, обеспечивающий ввод исходных данных.

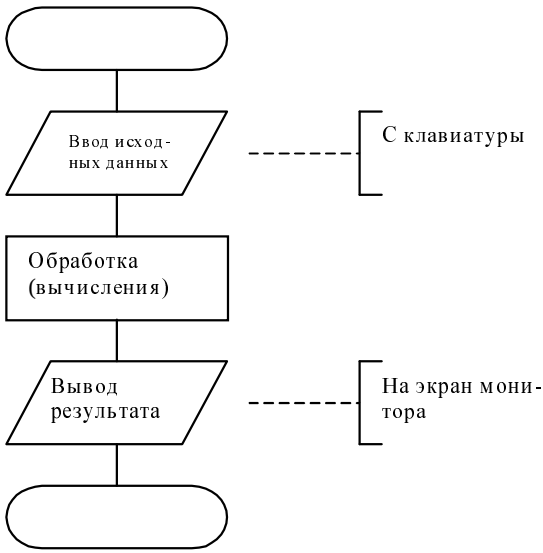
**50.** Объявите необходимые переменные и напишите инструкции ввода исходных данных для программы вычисления стоимости покупки, состоящей из нескольких тетрадей и карандашей. Предполагается, что пользователь будет вводить данные о каждой составляющей покупки в отдельной строке: сначала цену, затем количество.

# Программы с линейной структурой

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- ❒ программы с линейной структурой являются простейшими и используются, как правило, для реализации простых вычислений по формулам;
- ❒ в программах с линейной структурой инструкции выполняются последовательно, одна за другой;
- ❒ алгоритм программы с линейной структурой может быть представлен в виде схемы, показанной на рисунке.



## Задачи

**51.** Написать программу вычисления площади параллелограмма. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление площади прямоугольника
Введите исходные данные:
Длина (см) -> 9
Ширина (см) ->7.5

Площадь параллелограмма: 67.50 кв.см.
```

**52.** Написать программу вычисления объема параллелепипеда. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление объема параллелепипеда.
Введите исходные данные:
Длина (см) -> 9
Ширина (см) -> 7.5
Высота (см) -> 5

Объем: 337.50 куб.см.
```

**53.** Написать программу вычисления площади поверхности параллелепипеда. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление площади поверхности параллелепипеда.
Введите исходные данные:
Длина (см) -> 9
Ширина (см) -> 7.5
Высота (см) -> 5

Площадь поверхности: 90.00 кв.см.
```

**54.** Написать программу вычисления объема куба. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление объема куба.
Введите длину ребра (см) и нажмите клавишу <Enter>
-> 9.5

Объем куба: 857.38 куб.см.
```

**55.** Написать программу вычисления объема цилиндра. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление объема цилиндра
Введите исходные данные:
радиус основания (см) -> 5
высота цилиндра (см) -> 10
Объем цилиндра 1570.80 см.куб.
Для завершения нажмите <Enter>
```

**56.** Написать программу вычисления стоимости покупки, состоящей из нескольких тетрадей и карандашей. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление стоимости покупки.
Введите исходные данные:
Цена тетради (руб.) -> 2.75
Количество тетрадей ->5
Цена карандаша (руб.) ->0.85
Количество карандашей -> 2

Стоимость покупки: 15.45 руб.
```

**57.** Написать программу вычисления стоимости покупки, состоящей из нескольких тетрадей и такого же количества обложек к ним. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление стоимости покупки.
Введите исходные данные:
Цена тетради (руб.) -> 2.75
Цена обложки (руб.) ->0.5
Количество комплектов (шт.) -> 7

Стоимость покупки: 22.75 руб.
```

**58.** Написать программу вычисления стоимости некоторого количества (по весу) яблок. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление стоимости покупки.
Введите исходные данные:
Цена одного килограмма яблок (руб.) -> 8.5
Вес яблок (кг) -> 2.3

Стоимость покупки: 19.55 руб.
```

**59.** Написать программу вычисления площади треугольника, если известна длина основания и высоты. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление площади треугольника.
Введите исходные данные:
основание (см) -> 8.5
высота (см) -> 10

Площадь треугольника 42.50 кв.см.
```

**60.** Написать программу вычисления площади треугольника, если известны длины двух его сторон и величина угла между этими сторонами. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление площади треугольника.
Введите (через пробел) длины двух сторон (см.) треугольника
-> 25 17
Введите величину угла между сторонами треугольника
-> 30
Площадь треугольника: 106.25 кв.см.
```

**61.** Написать программу вычисления сопротивления электрической цепи, состоящей из двух параллельно соединенных сопротивлений. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление сопротивления электрической цепи
при параллельном соединении элементов.
Введите исходные данные:
Величина первого сопротивления (Ом) ->15
Величина второго сопротивления (Ом)->20

Сопротивление цепи: 8.57 Ом
```

**62.** Написать программу вычисления сопротивления электрической цепи, состоящей из двух последовательно соединенных сопротивлений. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление сопротивления электрической цепи.
Введите исходные данные:
Величина первого сопротивления (Ом) ->15
Величина второго сопротивления (Ом)->27.3

Сопротивление цепи (последовательное соединение): 42.30 Ом
```

**63.** Написать программу вычисления силы тока в электрической цепи. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление силы тока в электрической цепи.
Введите исходные данные:
Напряжение (вольт) -> 36
Сопротивление (Ом) -> 1500

Сила тока: 0.024 Ампер.
```

**64.** Написать программу вычисления расстояния между населенными пунктами, изображенными на карте. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление расстояния между населенными пунктами.
Введите исходные данные:
масштаб карты (количество километров в одном сантиметре) -> 120
расстояние между точками, изображающими населенные пункты
```

```
(см) -> 3.5

Расстояние между населенными пунктами 420 км.
```

**65.** аписать программу вычисления стоимости поездки на автомобиле на дачу (туда и обратно). Исходными данными являются: расстояние до дачи (км); количество бензина, которое потребляет автомобиль на 100 км пробега; цена одного литра бензина. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление стоимости поездки на дачу и обратно.
Расстояние до дачи (км) ->67
Расход бензина (литров на 100 км пробега) ->8.5
Цена литра бензина (руб.) ->6.5

Поездка на дачу и обратно обойдется в 74.04 руб.
```

**66.** Написать программу, вычисляющую скорость, с которой бегун пробежал дистанцию. Рекомендуемый вид экрана во время выполнения программыприведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

```
Вычисление скорости бега
Введите длину дистанции (метров) -> 1000
Введите время (минут.секунд)-> 3.25
Дистанция: 1000
Время: 3 мин  25 сек = 205 сек
Вы бежали со скоростью  17.56 км/час

Для завершения работы нажмите <Enter>
```

**67.** Написать программу вычисления объема цилиндра. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление объема цилиндра.
Введите исходные данные:
радиус основания (см) -> 5.5
высота цилиндра (см) -> 7

Объем цилиндра 665.23 см.куб.
```

**68.** Написать программу вычисления площади поверхности цилиндра. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление площади поверхности цилиндра.
Введите исходные данные:
радиус основания (см) -> 5.5
высота цилиндра (см) -> 7

Площадь поверхности цилиндра: 431.97 кв.см.
```

**69.** Написать программу вычисления объема параллелепипеда. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление объема параллелепипеда.
Введите в одной строке длину, ширину и высоту параллелепипеда
(в сантиметрах).
Числа разделяйте пробелами. После ввода последнего числа на-
жмите <Enter>.
-> 7.5 2.5 3

Объем параллелепипеда 56.25 см.куб.
```

**70.** Написать программу пересчета расстояния из верст в километры (1 верста —это 1066,8 метров). Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Пересчет расстояния из верст в километры
Введите расстояние в верстах -> 100
100 верст  — это 106.68 км
```

**71.** Написать программу пересчета веса из фунтов в килограммы (1 российский фунт — 405,9 гр). Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Пересчет веса из фунтов в килограммы
Введите вес в фунтах -> 5
5 фунтов — это 2.05 кг
```

**72.** Написать программу вычисления величины дохода по вкладу. Процентная ставка (% годовых) и время хранения (дней) за-

даются во время выполнения программыработы программы. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление дохода по вкладу.
Введите исходные данные:
Величина вклада (руб.) ->2500
Срок вклада (дней) -> 30
Процентная ставка (годовых) -> 20
-----------------------------------
Доход: 41.10 руб.
Сумма по окончании срока вклада: 2541.10 руб.
```

**73.** Написать программу пересчета величины временного интервала, заданного в минутах, в величину, выраженную в часах и минутах. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите временной интервал (в минутах)-> 150
150 минут — это 2 ч. 30 мин.
```

**74.** Написать программу, которая преобразует введенное с клавиатуры дробное число в денежный формат. Например, число 12,5 должно быть преобразовано к виду 12 руб. 50 коп.

```
Преобразование числа в денежный формат.
Введите дробное число ->23.6

23.6 руб. — это 23 руб. 60 коп.
```

**75.** Написать программу пересчета веса из фунтов в килограммы (1 фунт — 405,9 гр). Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Пересчет веса из фунтов в килограммы
Введите вес в фунтах и нажмите <Enter>.
->3.5
3.5 фунт(а/ов) — это 1 кг. 420 гр.
```

**76.** Напишите программу, которая вычисляет площадь треугольника, если известны координаты его углов. Ниже приведен рекомендуемый вид экрана во время выполнения про-

граммы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление площади треугольника
Введите координаты углов
(числа разделяйте пробелом):
x1,y1 ->-2 5
x2,y2 ->1 7
x3,y3 ->5 -3

Площадь треугольника: 23.56 кв.см.
```

# Выбор

## Инструкция *if*

### Общие замечания

Приступая к решению задач этого раздела следует вспомнить, что:

- инструкция **if** используется для выбора одного из двух направлений дальнейшего хода программы;
- выбор последовательности инструкций осуществляется в зависимости от значения *условия* — заключенного в скобки выражения, записанного после **if**;
- инструкция, записанная после **else** выполняются в том случае, если значение выражения *условие* равно нулю, во всех остальных случаях выполняется инструкция следующая за условием;
- если при выполнении или невыполнении условия надо выполнить несколько инструкций программы, то эти инструкции следует объединить в группу — заключить в фигурные скобки;
- при помощи вложенных одна в другую нескольких инструкций **if** можно реализовать множественный выбор.

### Задачи

**77.** Написать программу, которая вычисляет частное двух чисел. Программа должна проверять правильность введенных пользователем данных и, если они неверные (делитель равен нулю), вы-

давать сообщение об ошибке. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление частного.
Введите в одной строке делимое и делитель,
затем нажмите <Enter>.
-> 12 0
Вы ошиблись. Делитель не должен быть равен нулю.
```

**78.** Написать программу вычисления площади кольца. Программа должна проверять правильность исходных данных. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление площади кольца.
Введите исходные данные:
радиус кольца (см) -> 3.5
радиус отверстия (см) -> 7

Ошибка! Радиус отверстия не может быть больше радиуса кольца.
```

**79.** Написать программу, которая переводит время из минут и секунд в секунды. Программа должна проверять правильность введенных пользователем данных и в случае, если данные неверные, выводить соответствующее сообщение. Рекомендуемый вид экрана во время выполнения программыприведен ниже. Ошибочные данные, введенные пользователем, выделены полужирным шрифтом.

```
Введите время (минут.секунд) -> 2.90
Ошибка! Количество секунд не может быть больше 60
nДля завершения нажмите <Enter>
```

**80.** Написать программу, которая проверяет, является ли год високосным. Ниже приведен рекомендуемый вид экрана во время выполнения программыработы программы. Данные, введенные пользователем, выделены полужирным шрифтом.

```
Введите год, например 2000, и нажмите <Enter>
->2001
2000 год - не високосный
Для завершения нажмите <Enter>
```

**81.** Написать программу вычисления сопротивления электрической цепи, состоящей из двух сопротивлений. Сопротивления могут быть соединены последовательно или параллельно. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление сопротивления электрической цепи.
Введите исходные данные:
Величина первого сопротивления (Ом) ->15
Величина второго сопротивления (Ом)->27.3
Тип соединения (1-последовательное, 2- параллельное) ->2

Сопротивление цепи: 9.68 Ом
```

**82.** Написать программу решения квадратного уравнения. Программа должна проверять правильность исходных данных и в случае, если коэффициент при второй степени неизвестного равен нулю, выводить соответствующее сообщение. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
* Решение квадратного уравнения *
Введите в одной строке значения коэффициентов и нажмите
<Enter>
->12 27 -10
Корни уравнения:
x1= -25.551
x2= -28.449
```

**83.** Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 10% предоставляется, если сумма покупки больше 1000 руб. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление стоимости покупки с учетом скидки
Введите сумму покупки и нажмите <Enter>
->1200
Вам предоставляется скидка 10%
Сумма покупки с учетом скидки: 1080.00 руб.
```

**84.** Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 3% предоставляется, если сумма покупки больше 500 руб, в 5% — если сумма больше 1000 руб. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление стоимости покупки с учетом скидки
Введите сумму покупки и нажмите <Enter>
->640
Вам предоставляется скидка 3%
Сумма с учетом скидки: 620.80 руб.
```

**85.** Написать программу проверки знания даты основания Санкт-Петербурга. В случае неправильного ответа пользователя, программа должна выводить правильный ответ. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
В каком году был основан Санкт-Петербург?
Введите число и нажмите <Enter>
-> 1705
Вы ошиблись, Санкт-Петербург был основан в 1703 году.
```

**86.** Написать программу проверки знания даты начала второй мировой войны. В случае неправильного ответа пользователя, программа должна выводить правильный ответ. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
В каком году началась вторая мировая война?
Введите число и нажмите <Enter>
-> 1939
Правильно.
```

**87.** Напишите программу проверки знания истории архитектуры. Программа должна вывести вопрос и три варианта ответа. Пользователь должен выбрать правильный ответ и ввести его номер. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Архитектор Исаакиевского собора:
1. Доменико Трезини
```

```
2. Огюст Монферран
3. Карл Росси

Введите номер правильного ответа и нажмите <Enter>
-> 3
Вы ошиблись.
Архитектор Исаакиевского собора — Огюст Монферран.
```

**88.** Напишите программу проверки знания истории архитектуры. Программа должна вывести вопрос и три варианта ответа. Пользователь должен выбрать правильный ответ и ввести его номер. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Невский проспект получил свое название:
1. по имени реки, на берегах которой расположен Санкт-
Петербург
2. по имени близко расположенного монастыря Александро-
Невской лавры
3. в память о знаменитом полководце Александре Невском

Введите номер правильного ответа и нажмите <Enter>
-> 1
Вы ошиблись.
Правильный ответ: 2.
```

**89.** Написать программу, которая сравнивает два введенных с клавиатуры числа. Программа должна указать, какое число больше, или, если числа равны, вывести соответствующее сообщение. Ниже приведен рекомендуемый вид экрана во время выполнения программыработы программы.

```
Введите в одной строке два целых числа и нажмите <Enter>.
-> 34 67
34 меньше 67
```

**90.** Написать программу, которая выводит пример на умножение двух однозначных чисел, запрашивает ответ пользователя, проверяет его и выводит сообщение "Правильно!" или "Вы ошиблись" и правильный результат. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Сколько будет 6x7 ?
Введите ответ и нажмите <Enter>
-> 56
Вы ошиблись. 6x7=42
```

**91.** Написать программу, которая выводит пример на вычитание (в пределах 100), запрашивает ответ пользователя, проверяет его и выводит сообщение "Правильно!" или "Вы ошиблись" и правильный результат. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Сколько будет 83-17 ?
Введите ответ и нажмите <Enter>
->67
Вы ошиблись. 83-17=66
```

**92.** Написать программу, которая проверяет, является ли введенное пользователем целое число четным. Ниже приведен рекомендуемый вид экрана программы во время выполнения программыее работы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите целое число и нажмите <Enter>
-> 23
Число 23 — нечетное.
```

**93.** Написать программу, которая проверяет, делится ли на три введенное с клавиатуры целое число. Ниже приведен рекомендуемый вид экрана программы во время ее работы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите целое число и нажмите <Enter>
-> 451
Число 451 нацело на три не делится.
```

**94.** Написать программу вычисления стоимости разговора по телефону с учетом 20% скидки, предоставляемой по субботам и воскресеньям. Ниже приведен рекомендуемый вид экрана программы во время ее работы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление стоимости разговора по телефону.
Введите исходные данные:
Длительность разговора (целое количество минут) -> 3
```

```
День недели (1 — понедельник, ... 7 — воскресенье) -> 6
Предоставляется скидка 20%.
Стоимость разговора: 5.52 руб.
```

**95.** Написать программу, которая вычисляет оптимальный для пользователя, сравнивает его с реальным и выдает рекомендацию о необходимости поправиться или похудеть. Оптимальный вес вычисляется по формуле: Рост (см) — 100 . Рекомендуемый вид экрана во время выполнения программыприведен ниже (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите в одной строке через пробел
рост (см) и вес (кг) затем нажмите <Enter>
->170 68
Вам надо поправиться на 2.00 кг.
```

**96.** Напишите программу, которая запрашивает у пользователя номер месяца и затем выводит соответствующее название времени года. В случае, если пользователь введет недопустимое число, программа должна вывести сообщение "Ошибка ввода данных". Ниже приведен рекомендуемый вид экрана во время работы программы.

```
Введите номер месяца (число от 1 до 12)
-> 11
Зима
```

**97.** Написать программу, которая запрашивает у пользователя номер дня недели и выводит одно из сообщений: "Рабочий день", "Суббота" или "Воскресенье".

**98.** Написать программу, которая после введенного с клавиатуры числа (в диапазоне от 1 до 999), обозначающего денежную единицу, дописывает слово "рубль" в правильной форме. Например, 12 рублей, 21 рубль и так далее.

**99.** Написать программу, которая после введенного с клавиатуры числа (в диапазоне от 1 до 99), обозначающего денежную единицу, дописывает слово "копейка" в правильной форме. Например, 5 копеек, 41 копейка и так далее.

**100.** Написать программу, которая вычисляет дату следующего дня. Ниже приведен рекомендуемый вид экрана во время работы программы. (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите цифрами сегодняшнюю дату (число месяц год) -> 31 12
2000
Последний день месяца!
С наступающим новым годом!
Завтра 1.1.2001
```

# Инструкция *switch*

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- инструкция **switch** предназначена для выбора одного из нескольких возможных направлений дальнейшего хода программы;
- выбор последовательности инструкций осуществляется в зависимости от равенства значения переменной-селектора константе, указанной после слова **case**;
- если значение переменной-селектора не равно ни одной из констант, записанных после **case**, то выполняются инструкции, расположенные после слова **default**;
- в качестве переменной-селектора можно использовать переменную целого (**int**) или символьного (**char**) типа.

## Задачи

**101.** Напишите программу, которая запрашивает у пользователя номер дня недели, затем выводит название дня недели или сообщение об ошибке, если введены не верные данные.

**102.** Написать программу, которая вычисляет стоимость междугороднего телефонного разговора (цена одной минуты определяется расстоянием до города, в котором находится абонент). Исходными данными для программы являются код города и длительность разговора. Ниже приведены коды некоторых городов и рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

| Город | Код | Цена минуты (руб.) |
|---|---|---|
| Владивосток | 423 | 2,2 |
| Москва | 095 | 1,0 |
| Мурманск | 815 | 1,2 |
| Самара | 846 | 1,4 |

```
Вычисление стоимости разговора по телефону.
Введите исходные данные:
Код города -> 423
Длительность (целое количество минут)-> 3
Город: Владивосток
Цена минуты: 2.20 руб.
Стоимость разговора: 6.60 руб.
```

**103.** Напишите программу, которая по дате определяет день недели, на который эта дата приходится. Для вычисления дня недели воспользуйтесь формулой:

$$\left(d+\left[\frac{1}{5}(13m-1)\right]+Y+\left[\frac{Y}{4}\right]+\left[\frac{c}{4}\right]-2c+777\right) \bmod 7$$

Здесь d — число месяца, m — номер месяца, если начинать счет с марта, как это делали в Древнем Риме ( март — 1, апрель -2, ..., февраль — 12), Y — номер года в столетии, c — количество столетий. Квадратные скобки означают, что надо взять целую часть от значения, находящегося в скобках. Вычисленное по формуле значение определяет день недели: 1 — понедельник, 2 — вторник, ..., 6 — суббота, 0 — воскресенье.

# Циклы

## *for*

### Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- инструкция **for** используется для организации циклов с фиксированным, известным во время разработки программы, числом повторений;

- количество повторений цикла определяется начальным значением переменной-счетчика и условием завершения цикла;
- переменная-счетчик должна быть целого (**int**) типа и может быть объявлена непосредственно в инструкции цикла.

## Задачи

**104.** Написать программу, которая выводит на экран Ваше имя и фамилию 10 раз.

**105.** Написать программу, которая выводит таблицу квадратов первых десяти целых положительных чисел. Ниже приведен рекомендуемый вид экрана во время работы программы.

```
Таблица квадратов
--------------
Число Квадрат
--------------
  1       1
  2       4
  3       9
  4      16
  5      25
  6      36
  7      49
  8      64
  9      81
 10     100
--------------
```

**106.** Написать программу, которая выводит таблицу квадратов первых пяти целых положительных нечетных чисел. Ниже приведен рекомендуемый вид экрана во время работы программы.

```
Таблица квадратов нечетных чисел.
--------------
Число Квадрат
--------------
  1       1
  3       9
  5      25
  7      49
  9      81
--------------
```

**107.** Написать программу, которая вычисляет сумму первых n целых положительных целых чисел. Количество суммируемых чисел должно водиться во время работы программы. Ниже приведен рекомендуемый вид экрана (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление суммы положительных чисел.
Введите количество суммируемых чисел -> 20
Сумма первых 20 положительных чисел равна 210
```

**108.** Написать программу, которая вычисляет сумму первых n целых положительных четных целых чисел. Количество суммируемых чисел должно вводиться во время работы программы. Ниже приведен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление суммы четных положительных чисел.
Введите количество суммируемых чисел и нажмите <Enter>
-> 12
Сумма первых 12 положительных четных чисел равна 156
```

**109.** Написать программу, которая вычисляет сумму первых n членов ряда: 1,3,5,7 ... Количество суммируемых членов ряда задается во время работы программы. Ниже приведен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление частичной суммы ряда: 1,3,5,7, ...
Введите количество суммируемых членов ряда ->15
Сумма первых 15 членов ряда равна 330
```

**110.** Написать программу, которая вычисляет сумму первых n членов ряда: $1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\ldots$ Количество суммируемых членов ряда задается во время работы программы. Ниже приведен рекомендуемый вид экрана (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление частичной суммы ряда: 1+1/2+1/3+ ...
Введите ко-во суммируемых членов ряда ->15
Сумма первых 15 членов ряда равна 3.3182
```

**111.** Написать программу, которая выводит таблицу степеней двойки, от нулевой до десятой. Ниже приведен рекомендуемый вид экрана во время работы программы.

```
Таблица степеней двойки

 0      1
 1      2
 2      4
 3      8
 4     16
 5     32
 6     64
 7    128
 8    256
 9    512
10   1024
```

**112.** Написать программу, которая вычисляет факториал введенного с клавиатуры числа. (Факториалом числа n называется произведение целых чисел от 1 до n. Например, факториал 1 равен 1, 8 — 40320).

```
Вычисление факториала.
Введите число, факториал которого надо вычислить
-> 7
Факториал 7 равен 5040
```

**113.** Написать программу, которая выводит таблицу значений функции $y=-2,4x^2+5x-3$ в диапазоне от -2 до 2, с шагом 0,5. Ниже приведен рекомендуемый вид экрана во время работы программы.

```
---------------
   x  |   y
---------------
  -2  | -22.60
-1.5  | -15.90
  -1  | -10.40
-0.5  |  -6.10
   0  |  -3.00
 0.5  |  -1.10
   1  |  -0.40
 1.5  |  -0.90
   2  |  -2.60
----------------
```

**114.** Написать программу, которая вводит с клавиатуры 5 дробных чисел и вычисляет их среднее арифметическое. Рекомендуемый вид экрана во время выполнения программыприведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

```
Вычисление среднего арифметического последовательности
дробных чисел. После ввода каждого числа нажимайте <Enter>
-> 5.4
-> 7.8
-> 3.0
-> 1.5
-> 2.3
Среднее арифметическое введенной последовательности: 4.00
Для завершения нажмите <Enter>
```

**115.** Написать программу, которая вычисляет среднее арифметическое вводимой с клавиатуры последовательности дробных чисел. Количество чисел должно задаваться во время работы программы. Рекомендуемый вид экрана приведен ниже.

```
Вычисление среднего арифметического последовательности дроб-
ных чисел.
Введите количество чисел последовательности -> 5
Вводите последовательность. После ввода каждого числа нажи-
майте <Enter>
-> 5.4
-> 7.8
-> 3.0
-> 1.5
-> 2.3
Среднее арифметическое введенной последовательности: 4.00
Для завершения нажмите <Enter>
```

**116.** Написать программу, которая вводит с клавиатуры последовательность из пяти дробных чисел и после ввода каждого числа выводит среднее арифметическое введенной части последовательности. Рекомендуемый вид экрана во время выполнения программыприведен ниже.

```
Обработка последовательности дробных чисел
После ввода каждого числа нажимайте <Enter>
->12.3
```

```
Введено чисел: 1 Сумма: 12.30 Сред.арифметическое: 12.30
->15
Введено чисел: 2 Сумма: 27.30 Сред.арифметическое: 13.65
->10
Введено чисел: 3 Сумма: 37.30 Сред.арифметическое: 12.43
->5.6
Введено чисел: 4 Сумма: 42.90 Сред.арифметическое: 10.73
->11.5
Введено чисел: 5 Сумма: 54.40 Сред.арифметическое: 10.88
Для завершения нажмите <Enter>
```

**117.** Написать программу, которая вычисляет среднее арифметическое последовательности дробных чисел, вводимых с клавиатуры. После ввода последнего числа, программа должна вывести минимальное и максимальное число последовательности. Количество чисел последовательности должно задаваться во время работы программы. Рекомендуемый вид экрана приведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

```
Обработка последовательности дробных чисел.
Введите количество чисел последовательности -> 5
Вводите последовательность. После ввода каждого числа нажи-
майте <Enter>
-> 5.4
-> 7.8
-> 3.0
-> 1.5
-> 2.3

Количество чисел: 5
Среднее арифметическое: 4.00
Минимальное число:
Максимальное число:
Для завершения нажмите <Enter>
```

**118.** Написать программу, которая генерирует последовательность из 10 случайных чисел в диапазоне от 1 до 10, выводит эти числа на экран и вычисляет их среднее арифметическое. Рекомендуемый вид экрана во время выполнения программы приведен ниже.

```
*** Случайные числа ***
 1  3  4  2  7  4  9  6  2  1   сред.арифм. 3.9
```

**119.** Написать программу, которая генерирует три последовательности из десяти случайных чисел в диапазоне от 1 до 10, выводит каждую последовательность на экран и вычисляет среднее арифметическое каждой последовательности. Рекомендуемый вид экрана во время выполнения программыприведен ниже.

```
*** Случайные числа ***
  6 10  4  2  5  8  1  7  7  3  сред.арифм. 5.30
 10  3  6  1 10  1  3  8  7  6  сред.арифм. 5.50
  5  2  2  5  4  2  2  1  6 10  сред.арифм. 3.90
Для завершения работы нажмите <Enter>
```

**120.** Написать программу, которая выводит на экран таблицу стоимости, например, яблок в диапазоне от 100 гр до 1 кг с шагом 100 гр. Ниже приведен рекомендуемый вид экрана программы во время ее работы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите цену одного килограмма и нажмите <Enter>
(копейки от рублей отделяйте точкой)
-> 16.50

Вес       Стоимость
(гр.)     (руб.)
100       1.65
200       3.30
300       4.95
400       6.60
500       8.25
600       9.90
700       11.55
800       13.20
900       14.85
1000      16.50
```

**121.** Написать программу, которая выводит таблицу значений функции y=|x|. Диапазон изменения аргумента от -4 до 4, шаг приращения аргумента 0,5.

**122.** Написать программу, которая выводит таблицу значений функции y=|x-2|+|x+1|. Диапазон изменения аргумента от -4 до 4, шаг приращения аргумента 0,5.

**123.** Напишите программу, которая выводит на экран таблицу умножения, например, на 7. Рекомендуемый вид экрана во время выполнения программыприведен ниже.

```
7x2=14
7x3=21
7x4=28
7x5=35
7x6=42
7x7=49
7x8=56
7x9=63
```

**124.** Напишите программу, которая выводит на экран квадрат Пифагора — таблицу умножения. Рекомендуемый вид экрана во время выполнения программыприведен ниже.

```
   1  2  3  4  5  6  7  8  9 10
1  1  2  3  4  5  6  7  8  9 10
2  2  4  6  8 10 12 14 16 18 20
3  3  6  9 12 15 18 21 24 27 30
4  4  8 12 16 20 24 28 32 36 40
5  5 10 15 20 25 30 35 40 45 50
6  6 12 18 24 30 36 42 48 54 60
7  7 14 21 28 35 42 49 56 63 70
8  8 16 24 32 40 48 56 64 72 80
9  9 18 27 36 45 54 63 72 81 90
```

**125.** Напишите программу, которая вычисляет частичную сумму ряда: 1-1/3+1/5-1/7+1/9-...и сравнивает полученное значение с $\pi$ /4 (при суммировании достаточно большого количества членов этого ряда, величина частичной суммы приближается к $\pi/4$).

**126.** Напишите программу приближенного вычисления интеграла функции $f(x)=5x^2-x+2$ методом прямоугольников.

**127.** Напишите программу приближенного вычисления интеграла методом трапеций.

**128.** Напишите программу, которая выводит на экран изображение шахматной доски. Черные клетки отображать "звездочкой", белые — пробелом. Рекомендуемый вид экрана во время выполнения программыприведен ниже.

```
* * * *
 * * * *
* * * *
 * * * *
* * * *
 * * * *
* * * *
 * * * *
```

**129.** Написать программу, которая преобразует введенное пользователем десятичное число в двоичное. Рекомендуемый вид экрана во время выполнения программыприведен ниже.

```
Преобразование десятичного числа в двоичное
Введите целое число от 0 до 255 и нажмите <Enter>
->49
Десятичному числу 49 соответствует двоичное 00110001
Для завершения нажмите <Enter>
```

## Факультатив

**130.** Написать программу проверки знания таблицы умножения. Программа должна вывести 10 примеров и выставить оценку: за 10 правильных ответов — "отлично", за 9 и 8 -"хорошо", за 7 и 6 — "удовлетворительно", за 6 и менее — "плохо". Ниже приведен рекомендуемый вид экрана во время работы программы. Ответы пользователя выделены полужирным шрифтом.

```
*** Проверка знания таблицы умножения ***
После примера введите ответ и нажмите <Enter>.
5x3=15
7x7=49
1x4=4
4x3=12
9x4=36
8x8=64
7x8=52
Вы ошиблись! 7x8=56
4x7=28
3x5=15
2x5=10
Правильных ответов: 9
Оценка: Хорошо.
```

**131.** Написать программу проверки умения складывать и вычитать числа в пределах 100. Программа должна вывести 10 примеров, причем в каждом примере уменьшаемое должно быть больше или равно вычитаемому, т. е. не допускается предлагать испытуемому примеры с отрицательным результатом. Оценка выставляется по следующему правилу: за 10 правильных ответов — "отлично", за 9 и 8 — "хорошо", за 7 и 6 — "удовлетворительно", за 6 и менее — "плохо". Ниже приведен рекомендуемый вид экрана во время работы программы. Ответы пользователя выделены полужирным шрифтом.

```
Проверка умения складывать вычитать числа.
После примера введите ответ и нажмите <Enter>
75-4=71
35-9=29
Вы ошиблись! 35-9=26
14-1=13
6-5=1
37-19=28
Вы ошиблись! 37-19=18
53-14=39
94-87=7
90-16=74
4-2=2
89-41=48
Правильных ответов: 8
Оценка: Хорошо
```

**132.** Написать программу, которая выводит на экран работающие "электронные часы", которые работают в течение, например, трех минут или до тех пор, пока пользователь не нажмет любую клавишу.

# *do ... while*

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- число повторений инструкций цикла **`do while`** определяется ходом выполнения программы;

- инструкции цикла **do while** выполняются до тех пор, пока значение выражения, записанного после слова **while**, не станет равным нулю;
- после слова **while** надо записывать условие выполнения инструкций цикла;
- для завершения цикла **do while** в теле цикла обязательно должны быть инструкции, выполнение которых влияет на условие завершения цикла;
- цикл **do while** — это цикл с постусловием, т. е. инструкции тела цикла будут выполнены хотя бы один раз;
- цикл **do while**, как правило, используется для организации приближенных вычислений, в задачах поиска и обработки данных, вводимых с клавиатуры или из файла.

## Задачи

**133.** Написать программу, вычисляющую сумму и среднее арифметическое последовательности положительных чисел, которые вводятся с клавиатуры. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Вычисление среднего арифметического последовательности поло-
жительных чисел.
Вводите после стрелки числа. Для завершения ввода введите
ноль.
->45
->23
->15
->0
Введено чисел: 3
Сумма чисел: 83
Среднее арифметическое: 27.67
```

**134.** Написать программу, которая определяет максимальное число из введенной с клавиатуры последовательности положительных чисел (длина последовательности неограниченна). Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Определение максимального числа последовательности положи-
тельных чисел.
Вводите после стрелки числа. Для завершения ввода введите
ноль.
->56
->75
->43
->0
Максимальное число: 75
```

**135.** Написать программу, которая определяет минимальное число во введенной с клавиатуры последовательности положительных чисел (длина последовательности неограниченна). Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Определение минимального числа в последовательности положи-
тельных чисел.
Вводите после стрелки числа. Для завершения ввода введите
ноль.
->12->75
->10
->9
->23
->0
Минимальное число: 9
```

**136.** Напишите программу, которая проверяет, является ли введенное пользователем целое число простым. Рекомендуемый вид экрана во время выполнения программыприведен ниже. Данные введенные пользователем выделены полужирным шрифтом.

```
Введите целое число и нажмите <Enter>
-> 45
45 — не простое число.
```

**137.** Написать программу приближенного вычисления интеграла методом трапеций. После каждого цикла вычислений программа должна выводить вычисленное значение, количество и величину интервалов.

**138.** Написать программу, которая "задумывает" число в диапазоне от 1 до 10 и предлагает пользователю угадать число за 5 по-

пыток. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Игра "Угадай число".
Компьютер "задумал" число от 1 до 10.
Угадайте его за 5 попыток.
Введите число и нажмите <Enter>
-> 5
Нет.
-> 3
Вы выиграли! Поздравляю!
```

### Факультатив

**139.** Написать программу-таймер, которая по истечении заданного промежутка времени, величина которого вводится с клавиатуры, выдает звуковой сигнал.

## *while*

### Общие замечания

Приступая к решению задач этого раздела следует вспомнить, что:

Приступая к решению задач этого раздела следует вспомнить, что:

- число повторений инструкций цикла **do while** определяется ходом выполнения программы;
- инструкции цикла **while** выполняются до тех пор, пока значение выражения, записанного после слова **while**, не станет равным нулю;
- после слова **while** надо записывать условие выполнения инструкций цикла;
- для завершения цикла **while** в теле цикла обязательно должны быть инструкции, выполнение которых влияет на условие завершения цикла;
- цикл **while** — это цикл с предусловием, т. е. возможна ситуация, при которой инструкции тела цикла ни разу не будут выполнены;

- цикл **while**, как правило, используется для организации приближенных вычислений, в задачах поиска и обработки данных, вводимых с клавиатуры или из файла.

## Задачи

**140.** Напишите программу, которая выводит на экран таблицу значения функции $y=2x^2-5x-8$ в диапазоне от -4 до 4. Шаг изменения аргумента 0.5.

**141.** Напишите программу, которая вычисляет число "Пи" с заданной пользователем точностью. Для вычисления значения числа "Пи" воспользуйтесь тем, что значение частичной суммы ряда 1-1/3+1/5-1/7+1/9-..., при суммировании достаточно большого количества членов, приближается к значению $\pi/4$. Рекомендуемый вид экрана во время выполнения программыприведен ниже (данные, введенные пользователем, выделены полужирным шрифтом).

```
Задайте точность вычисления ПИ -> 0.001
Значение числа ПИ с точностью 0.001000 равно 3.143589
Просуммировано 502 членов ряда.
```

**142.** Написать программу, которая вычисляет наибольший общий делитель двух целых чисел. Рекомендуемый вид экрана во время выполнения программыприведен ниже (данные, введенные пользователем, выделены полужирным шрифтом).

# Массивы

## Общие замечания

Приступая к решению задач этого раздела следует вспомнить, что:

- массив — это структура данных, представляющая собой набор, совокупность, элементов одного типа;
- в инструкции объявления массива указывается количество элементов массива;
- элементы массива нумеруются с нуля;
- доступ к элементу массива осуществляется путем указания индекса (номера) элемента. В качестве индекса можно ис-

пользовать выражение целого типа — константу или переменную. Индекс может меняться от 0 до n-1, где n — количество элементов массива;

- доступ к элементам массива можно осуществить при помощи указателя;
- в инструкции объявления массива удобно использовать именованную константу, объявленную в директиве `#define`;
- для ввода, вывода и обработки массивов удобно использовать инструкции циклов (**`for`**, **`while`**);
- типичной ошибкой при использовании массивов является обращение к несуществующему элементу, т. е. выход индекса за допустимое значение.

## Задачи

**143.** Написать программу, которая вводит с клавиатуры одномерный массив из 5 целых чисел, после чего выводит количество ненулевых элементов. Перед вводом каждого элемента должна выводиться подсказка с номером элемента.

Ввод массива целых чисел.

```
После ввода каждого числа нажмите <Enter>
a[1] -> 12
a[2] -> 0
a[3] -> 3
a[4] -> -1
a[5] -> 0
В массиве 3 ненулевых элемента.
```

**144.** Написать программу, которая выводит минимальный элемент введенного с клавиатуры массива целых чисел. Ниже приведен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Поиск минимального элемента массива.
Введите в одной строке элементы массива (5 целых чисел)
и нажмите <Enter>
-> 23 0 45 -5 12
Минимальный элемент массива: -5
```

**145.** Написать программу, которая выводит минимальный элемент введенного с клавиатуры массива целых чисел. Для доступа к элементам массива используйте указатель.

**146.** Написать программу, которая вычисляет среднее арифметическое ненулевых элементов введенного с клавиатуры массива целых чисел. Ниже приведен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите элементы массива (10 целых чисел) в одной строке и
нажмите <Enter>.
-> 23 0 45 -5 12 0 -2 30 0 64
Сумма элементов массива: 184
Количество ненулевых элементов: 7
Среднее арифметическое ненулевых элементов: 23.86
```

**147.** Написать программу, которая вычисляет среднее арифметическое элементов массива без учета минимального и максимального элементов массива. Ниже приведен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

```
Среднее арифметическое без учетаmin и max значений.
Введите массив (10 целых чисел в одной строке)
->12 10 5 7 15 4 10 17 23 7
Минимальный элемент: 4
Максимальный элемент: 23
Среднее арифм. без учета min и max значений:
```

**148.** Написать программу, которая вычисляет среднюю (за неделю) температуру воздуха. Исходные данные должны вводиться во время работы программы. Рекомендуемый вид экрана приведен ниже (данные, введенные пользователем выделены полужирным шрифтом).

```
Введите температуру воздуха за неделю.
Понедельник -> 12
Вторник -> 10
Среда -> 16
Четверг -> 18
Пятница -> 17
Суббота -> 16
Воскресенье -> 14
Средняя температура за неделю: 14.71 град.
```

**149.** Написать программу, которая проверяет, находится ли введенное с клавиатуры число в массиве. Массив должен вводиться во время работы программы.

**150.** Написать программу, которая проверят, представляют ли элементы введенного с клавиатуры массива возрастающую последовательность.

**151.** Написать программу, которая вычисляет, сколько раз введенное с клавиатуры число встречается в массиве.

**152.** Написать программу, которая проверяет, есть ли во введенном с клавиатуры массиве элементы с одинаковым значением.

**153.** Написать программу, которая методом прямого выбора сортирует по убыванию введенный с клавиатуры одномерный массив.

**154.** Написать программу, которая методом обмена ("пузырька") сортирует по убыванию введенный с клавиатуры одномерный массив.

**155.** Написать программу, которая объединяет два упорядоченных по возрастанию массива в один, также упорядоченный массив. Рекомендуемый вид экрана во время работы программы приведен ниже, данные, введенные пользователем, выделены полужирным шрифтом.

```
Объединение двух упорядоченных по возрастанию массивов.
Введите в одной строке элементы первого массива,
( 5 целых чисел) -> 1 3 5 7 9
Введите в одной строке элементы второго массива,
( 5 целых чисел) -> 2 4 6 8 10
Массив — результат
1 2 3 4 5 6 7 8 9 10
Для завершения работы нажмите <Enter>.
```

**156.** Написать программу, которая, используя метод бинарного поиска, выполняет поиск в упорядоченном по возрастанию массиве.

**157.** Написать программу, которая определяет количество учеников в классе, чей рост превышает средний. Рекомендуемый вид экрана во время работы программы приведен ниже. Введенные пользователем данные выделены полужирным шрифтом.

```
*** Анализ роста учеников ***
Введите рост (см) и нажмите <Enter>.
Для завершения введите 0 и нажмите <Enter>
```

```
->175
->170
->180
->168
->170
->0
Средний рост: 172.6 см
У 2 человек рост превышает средний.
```

**158.** Написать программу, которая вводит по строкам с клавиатуры двумерный массив и вычисляет сумму его элементов по столбцам.

**159.** Написать программу, которая вводит по строкам с клавиатуры двумерный массив и вычисляет сумму его элементов по строкам.

**160.** Написать программу, которая обрабатывает результаты экзамена. Для каждой оценки программа дожна вычислить процент от общего количества оценок. Рекомендуемый вид экрана во время работы программы приведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

```
Обработка результатов экзамена
Введите исходные данные:
пятерок ->12
четверок ->10
троек ->7
двоек ->1
Результаты экзамена
----------------------------
пятерок   12      %
четверок  10      %
троек      7      %
двоек      1      %
----------------------------
Для завершения программы нажмите <Enter>
```

**161.** Написать программу, которая вводит по строкам с клавиатуры двумерный массив и вычисляет среднее арифметическое его элементов.

**162.** Написать программу, которая вычисляет определитель квадратной матрицы второго порядка. Рекомендуемый вид экрана во время работы программы приведен ниже (введенные пользователем данные выделены полужирным шрифтом).

```
Введите матрицу второго порядка.
После ввода элементов строки нажимайте <Enter>
-> 5 -7
-> 1 3
Определитель матрицы
  5.00 -7.00
  1.00  3.00
Равен  22.00
```

**163.** Написать программу, которая определяет номер строки квадратной матрицы сумма элементов которой максимальна.

**164.** Написать программу, которая проверяет, является ли введенная с клавиатуры квадратная матрица "магическим квадратом".

Магическим квадратом называется матрица, у которой сумма чисел в каждом горизонтальном ряду, в каждом вертикальном и по каждой из диагоналей одна и та же (см. приведенный ниже рисунок).

| | | |
|---|---|---|
| 2 | 9 | 4 |
| 7 | 5 | 3 |
| 6 | 1 | 8 |

| | | | |
|---|---|---|---|
| 13 | 8 | 12 | 1 |
| 2 | 11 | 7 | 14 |
| 3 | 10 | 6 | 15 |
| 16 | 5 | 9 | 4 |

# Символы и строки

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- каждому символу соответствует число — код символа;
- в C/C++ строка — это массив символов;
- последним символом строки обязательно должен быть нуль-символ, код которого равен 0, и который в тексте программы изображается так `'\0'`;
- сообщения или подсказки, используемые в программе, удобно представить как массив указателей на строки и инициализи-

ровать массив, задать сообщения, в инструкции объявления массива:

```
char *mes[] ={"Сообщение 1","Сообщение 2", ... , "Сообщение"};
```

- если вводимая во время работы программы строка содержит пробелы, то функция `scanf` вводит только часть строки, до первого пробела, а функция `gers` — всю строку, в том числе и соответствующий клавише <Enter> символ `'\n'`.

## Задачи

**165.** Написать программу, которая запрашивает имя пользователя и здоровается с ним. Рекомендуемый вид экрана во время выполнения программыприведен ниже (данные, введенные пользователем, выделены полужирным шрифтом).

```
Как Вас зовут?
Введите свое имя и фамилию, затем нажмите <Enter>
-> Вася Иванов
Здравствуйте, Вася Иванов!
```

**166.** Написать программу, которая запрашивает у пользователи имя и отчество, затем здоровается с ним. Для ввода используйте функцию `getch()`.

**167.** Напишите программу, которая вычисляет длину введенной с клавиатуры строки.

**168.** Напишите программу, которая выводит на экран сообщение в "телеграфном" стиле: буквы сообщения должны появляться по одной, с некоторой задержкой.

**169.** Напишите программу, которая выводит код введенного пользователем символа. Программа должна завершать работу в результате ввода, например, точки. Рекомендуемый вид экрана во время выполнения программыприведен ниже (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите символ и нажмите <Enter>.
Для завершения введите точку.
->1
Символ: 1 Код: 49
->2
```

```
Символ: 2 Код: 50
->ы
Символ: ы Код:235
->.
```

**170.** Написать программу, которая выводит на экран первую часть таблицы кодировки символов (символы с кодами от 0 до 127). Таблица должна состоять из восьми колонок и шестнадцати строк. В первой колонке должны быть символы с кодом от 0 до 15, во второй — от 16 до 31 и так далее.

**171.** Написать программу, которая в веденной с клавиатуры строке преобразует строчные буквы русского алфавита в прописные (учтите, что стандартная функция `upcase` с символами русского алфавита не работает). Рекомендуемый вид экрана во время выполнения программыприведен ниже (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите строку текста и нажмите <Enter>
-> изучив основы C++, можно начать программировать под
Windows
Строка, преобразованная к верхнему регистру:
ИЗУЧИВ ОСНОВЫ C++, МОЖНО НАЧАТЬ ПРОГРАММИРОВАТЬ ПОД WINDOWS
```

**172.** Написать программу, которая удаляет из введенной с клавиатуры строки начальные пробелы.

**173.** Написать программу, которая проверяет, является ли введенная с клавиатуры строка целым числом. Рекомендуемый вид экрана во время выполнения программыприведен ниже (данные, введенные пользователем, выделены полужирным шрифтом).

```
Введите число и нажмите <Enter>
->23.5
Введенная строка не является целым числом.
```

**174.** Написать программу, которая проверяет, является ли введенная с клавиатуры строка двоичным числом.

**175.** Написать программу, которая проверяет, является введенная с клавиатуры строка шестнадцатеричным числом.

**176.** Написать программу, которая проверяет, является ли введенная с клавиатуры строка дробным числом.

**177.** Написать программу, которая преобразует введенное с клавиатуры восьмиразрядное двоичное число в десятичное. Реко-

мендуемый вид экрана во время выполнения программыприведен ниже (введенные пользователем данные выделены полужирным шрифтом).

```
Введите восьмиразрядное двоичное число
и нажмите <Enter>
->11101010
Двоичному числу 11101010 соответствует десятичное 234
Для завершения нажмите <Enter>
```

**178.** Написать программу, которая преобразует введенное с клавиатуры двухразрядное шестнадцатеричное число в десятичное.

**179.** Написать программу, которая преобразует введенное пользователем десятичное число в число в указанной системе счисления (от 2 до 10). Рекомендуемый вид экрана во время выполнения программыприведен ниже.

```
Введите целое число -> 67
Введите основание системы счисления -> 2
Десятичному числу 67 соответствует число 100011 по основанию 2
```

**180.** Написать программу, которая преобразует введенное пользователем десятичное число в шестнадцатеричное.

**181.** Написать программу, которая вычисляет значение выражения $N_0O_1N_1O_2..O_kN_k$, где $N_i$ — целое одноразрядное число, $O_i$ — один из двух знаков простейших арифметических действий: сложения ( + ) или вычитания. Ниже приведен рекомендуемый вид экрана во время работы программы, данные, введенные пользователем, выделены полужирным шрифтом.

```
Введите арифметическое выражение,
например, 4+5-3-5+2 и нажмите <Enter>
->9-5+4+2-6
Значение введенного выражения: 4
Для завершения программы нажмите <Enter>
```

## Факультатив

**182.** Написать программу, которая подводит итоги олимпийских игр. Программа должна получить от пользователя число медалей разного достоинства, завоеванное каждой командой-участницей, вычислить общее количество медалей и соответствующее ему число очков. и после этого упорядочить список в соответствии с

набранным количеством очков. Количество очков вычисляется по следующему правилу: за золотую медаль команда получает семь очков, за серебряную — шесть, за бронзовую — пять очков.

Рекомендуемый вид экрана во время работы программы приведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

```
Итоги олимпийских игр
Введите в одной строке количество золотых,
серебряных и бронзовых медалей.
Австрия -> 3 5 9
Германия -> 12 9 8
Канада -> 6 5 4
Китай -> 0 6 2
Корея -> 3 1 2
Норвегия -> 10 10 5
Россия -> 9 6 3
США -> 6 3 4
Финляндия -> 2 4 6
Япония -> 5 1 4

Итоги летней олимпиады в Сиднее, 2000 г.
     Страна   Золото Серебро Бронза Всего Очков
1        США
2     Россия
3      Китай
4   Германия
5     Канада
6    Франция
7  Финляндия
8     Япония
9      Китай
10     Корея
```

**183.** Написать программу, реализующую игру угадай число. Правила игры следующие. Играют двое. Один задумывает число, второй — угадывает. На каждом шаге угадывающий делает предположение, а задумавший число — говорит, сколько цифр числа угаданы и сколько из угаданных цифр занимают правильные позиции в числе. Например, если задумано число 725, и выдвинуто предположение, что задумано число 523, то угаданы две цифры (5 и 2) и одна из них (2) занимает верную позицию.

Ниже приведен рекомендуемый вид экрана во время работы программы. Данные, введенные пользователем выделены полужирным шрифтом.

```
Компьютер задумал трехзначное число. Вы должны его отгадать.
После очередного числа, вам будет сообщено, сколько цифр
угадано и сколько из них находятся на своих местах.
После ввода числа нажимайте <Enter>.
Для завершения игры нажмите <Esc>.
Ваш вариант -> 123 Угадано: 0. На своих местах: 0
Ваш вариант -> 456 Угадано: 1. На своих местах: 0
Ваш вариант -> 654 Угадано: 2. На своих местах: 2
Ваш вариант -> 657 Угадано: 2. На своих местах: 2
Ваш вариант -> 658 Угадано: 3. На своих местах: 3
*** ВЫ УГАДАЛИ ! ***
Нажмите <Enter> для завершения.
```

**184.** Напишите программу-телеграф, которая принимает от пользователя сообщение и выводит его на экран в виде последовательности точек и тире. Вывод точек и тире можно сопроводить звуковым сигналом, соответствующей длительности. Азбука Морзе для букв русского алфавита приведена ниже.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| А | .- | Б | -... | В | .--- | Г | --. |
| Д | -.. | Е | | Ж | ...- | З | --.. |
| И | .. | Й | .--- | К | -.- | Л | .-.. |
| М | -- | Н | -. | О | --- | П | .--. |
| Р | .-. | С | ... | Т | - | У | ..- |
| Ф | ..-. | Х | .... | Ц | -.-. | Ч | ---. |
| Ш | ---- | Щ | --.- | Ъ | -..- | Ы | -.-- |
| Ь | -..- | Э | ..-. | Ю | ..-- | Я | .-.- |

# Функции

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- для передачи данных в функцию надо использовать только параметры. Глобальные переменные, т. е. переменные, объявленные вне функции, использовать не рекомендуется;

- тип каждого фактического параметра (константы или переменной) в инструкции вызова функции должен совпадать с типом соответствующего формального параметра, указанного в объявлении функции;
- если параметр функции используется для возврата результата, то в объявлении функции этот параметр должен быть ссылкой, а в инструкции вызова функции в качестве фактического параметра должен быть указан адрес переменной.

## Задачи

**185.** Написать функцию, которая вычисляет объем цилиндра. Параметрами функции должны быть радиус и высота цилиндра.

**186.** Написать функцию, которая возвращает максимальное из двух целых чисел, полученных в качестве аргумента.

**187.** Написать функцию, которая сравнивает два целых числа и возвращает результат сравнения в виде одного из знаков: `>`, `<` или `=`.

**188.** Написать функцию, которая вычисляет сопротивление цепи, состоящей из двух резисторов. Параметрами функции являются величины сопротивлений и тип соединения (последовательное или параллельное). Функция должна проверять корректность параметров: если не верно указан тип соединения, то функция должна возвращать -1.

**189.** Написать функцию, которая вычисляет значение $a^b$. Числа a и b могут быть любыми дробными положительными числами.

**190.** Написать функцию `Procent`, которая возвращает процент от полученного в качестве аргумента числа.

**191.** Написать функцию "`Факториал`" и программу, использующую эту функцию для ывода таблицы факториалов.

**192.** Написать функцию `Dohod`, которая вычисляет доход по вкладу. Исходными данными для функции являются: величина вклада, процентная ставка (годовых) и срок вклада (количество дней).

**193.** Написать функцию `glasn`, которая возвращает `1`, если символ, полученный функцией в качестве аргумента, является гласной буквой русского алфавита, и ноль в противном случае.

**194.** Написать функцию `sogl`, которая возвращает `1`, если символ, полученный функцией в качестве аргумента, является согласной буквой русского алфавита, и 0 в противном случае

**195.** Написать функцию, которая возвращает преобразованную к верхнему регистру строку, полученную в качестве аргумента.

**196.** Написать функцию, обеспечивающую решение квадратного уравнения. Параметрами функции должны быть коэффициенты и корни уравнения. Значение функции должно передавать в использующую функцию программу информацию о наличии у уравнения корней: 2 — два разных корня, 1 — корни одинаковые, 0 — уравнение не имеет решения. Кроме того, функция должна проверять корректность исходных данных. Если исходные данные неверные, то функция должна возвращать -1.

**197.** Написать функцию, которая выводит на экран строку, состоящую из звездочек. Длина строки (количество звездочек) является параметром функции.

**198.** Написать функцию, которая выводит строку, состоящую из одинаковых символов. Длина строки и символ являются параметрами процедуры.

**199.** Написать функцию, которая вычисляет объем и площадь поверхности параллелепипеда.

**200.** Написать функцию `frame`, которая выводит на экран рамку. В качестве параметров функции должны передаваться координаты левого верхнего угла и размер рамки.

## Факультатив

**201.** Написать функцию, обеспечивающую ввод с клавиатуры целого положительного числа. При нажатии клавиши соответствующий символ должен появляться на экране только в том случае, если этот символ является цифрой. Функция должна позволять редактировать введенное число при помощи клавиши <Backspace>. При нажатии клавиши <Enter> функция должна завершать работу и возвращать введенное число.

**202.** Написать функцию, обеспечивающую ввод с клавиатуры дробного числа. При нажатии клавиши соответствующий символ должен появляться на экране только в том случае, если этот

символ является допустимым в данной позиции. Например, функция не должна допускать ввод более чем одной точки и знака минус не в первой позиции. Функция должна позволять редактировать введенное число при помощи клавиши <Backspace>. При нажатии клавиши <Enter> функция должна завершать работу и возвращать введенное число.

**203.** Написать программу, реализующую игру "21". Действия по выдаче очередной карты игроку и компьютеру реализуйте в виде функции.

# Графика

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- в графическом режиме экран представляет собой совокупность точек, каждая из которых может быть окрашена в один из 16 цветов;
- координаты точек возрастают слева направо и сверху вниз. Левая верхняя точка имеет координаты (0.0), правая нижняя — (639,479);
- для того, чтобы программа могла выводить на экран графические примитивы (линии, окружности, прямоугольники), необходимо инициализировать графический режим.

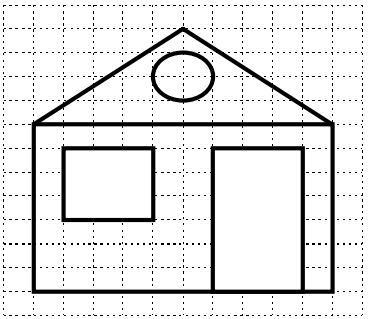
Шаблон графической программы выглядит следующим образом:

```
// шаблон графической программы
#include <graphics.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();
```

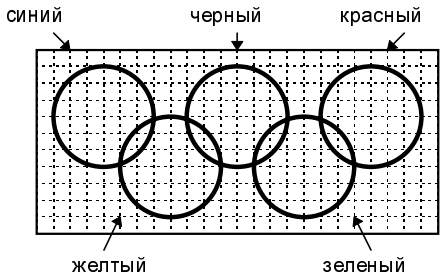
```
    if (errorcode != grOk)  // ошибка инициализации
                            // графического режима
    {
       printf("Ошибка: %d\n", errorcode);
       puts("Для завершения программы нажмите <Enter>");
       getch();
       return;
    }

    // далее инструкции программы


    closegraph(); // завершение графического режима
}
```

## Задачи

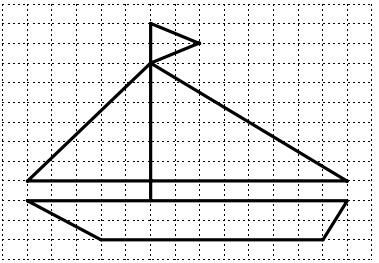
**204.** Написать программу, которая выверчивает на экране домик.



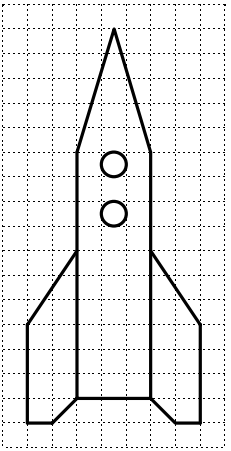
**205.** Написать программу, которая выводит экран флаг Олимпийских игр. Изображение флага приведено ниже (одной клетке соответствует пять пикселов).



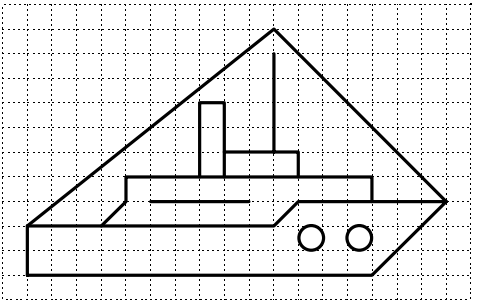
**206.** Написать программу, которая вычерчивает на экране кораблик.



**207.** Написать программу, которая вычерчивает на экране ракету.



**208.** Написать программу, которая с использованием метода базовой точки выводит на экран изображение кораблика.

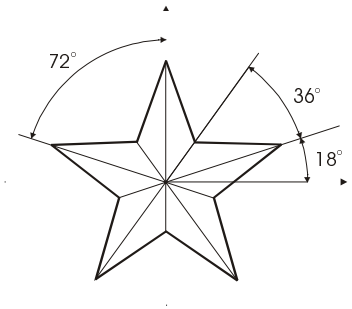


**209.** Написать программу, которая вычерчивает на экране узор из 100 окружностей случайного диаметра и цвета.

**210.** Написать программу, которая вычерчивает на экране узор из 50 прямоугольников случайного размера и цвета.

**211.** Написать программу, которая вычерчивает на экране узор — ломаную линию из 100 звеньев, со случайными координатами, случайного цвета.

**212.** Написать программу, которая выводит на экран контур пятиконечной звезды.




**213.** Написать программу, которая выводит на экран пятиконечную звезду красного цвета с белой окантовкой.

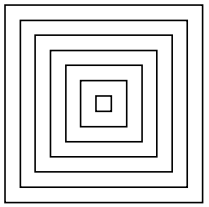
**214.** Написать программу, которая вычерчивает на экране шестиугольник.

**215.** Написать программу, которая рисует на экране Государственный флаг России.

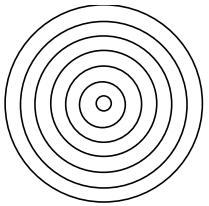
**216.** Написать программу, которая рисует на экране веселую рожицу желтого цвета.

**217.** Написать программу, которая рисует на экране грустную рожицу.

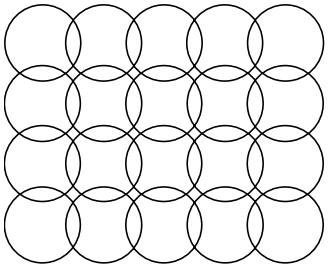
**218.** Написать программу, которая выводит на экран изображенный ниже узор.



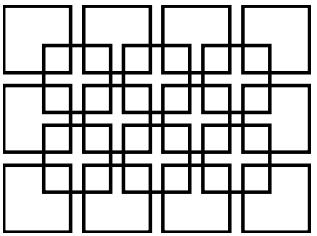
**219.** Написать программу, которая выводит на экран изображенный ниже узор. Окружности должны быть разного цвета: от синего до белого (смотри таблицу кодировки цветов, справочник, функция `setcolor`)..



**220.** Написать программу, которая выводит изображенный ниже узор.

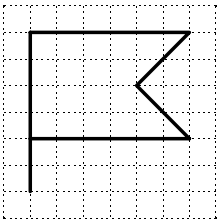


**221.** Написать программу, которая выводит изображенный ниже узор.

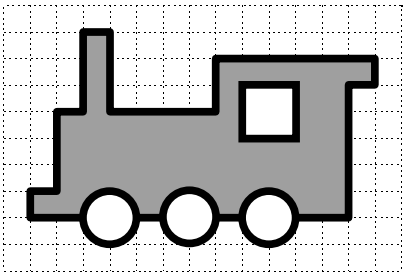


**222.** Написать программу, которая выводит на экран изображение шахматной доски.

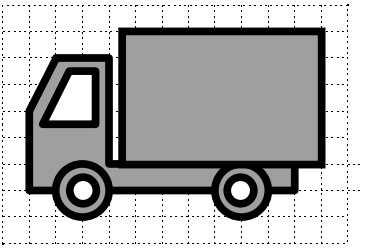
**223.** Написать программу, которая рисует на экране флажок красного цвета.



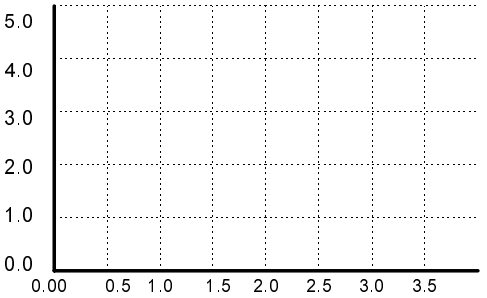
**224.** Написать программу, которая рисует на экране паровоз. Используйте метод базовой точки.



**225.** Написать программу, которая рисует на экране автомобиль. Инструкции, обеспечивающие вычерчивание колеса автомобиля, оформите как функцию.



**226.** Написать программу, которая выводит на экран оцифрованную координатную сетку.




**227.** Написать программу, которая выводит на экран точечный график функции $y=0,5x^2+4x-3$. Диапазон изменения аргумента: от -15 до 5; шаг аргумента: 0,1. График вывести на фоне координатных осей, точка пересечения которых должна находиться в центре экрана.

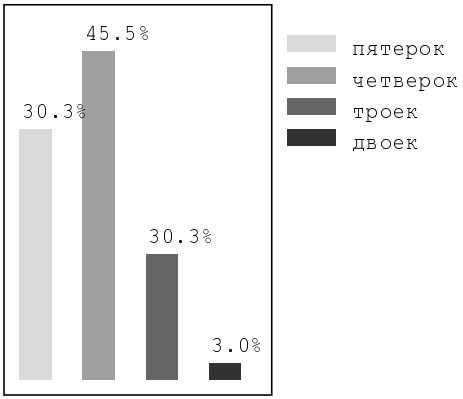
**228.** Написать программу, которая рисует движущуюся по экрану окружность.

**229.** Написать функцию, которая рисует на экране кораблик. В качестве параметров функция должна получать координаты базовой точки и цвет, которым следует рисовать. Используя эту фунцию, напишите программу, которая выводит на экран движущийся кораблик.

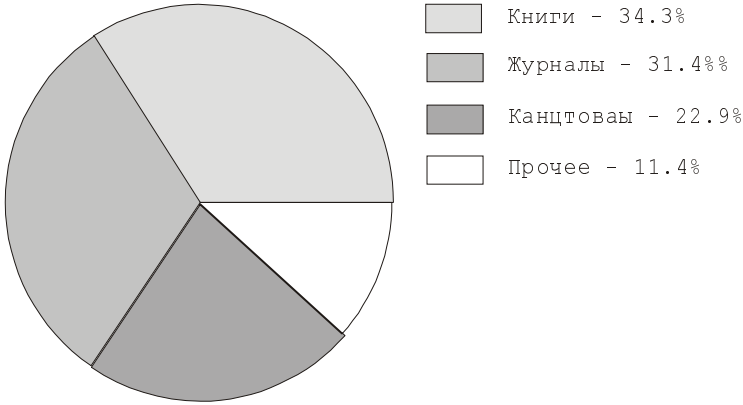
**230.** Написать программу, которая выводит на экран гистограмму успеваемости в классе, например по итогам контрольной работы. Исходные данные следует ввести в алфавитно-цифровом режиме работы. Рекомендуемый вид экрана во время работы программы приведен ниже. Ниже показан вид экрана во время ввода исходных данных, а ниже — вид диаграммы.

```
Обработка результатов контрольной работы
Введите исходные данные:
пятерок -> 10
четверок -> 15
троек -> 7
двоек -> 1
```

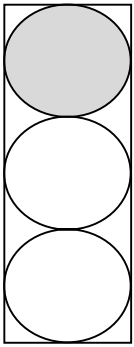
```
Результаты контрольной работы
```

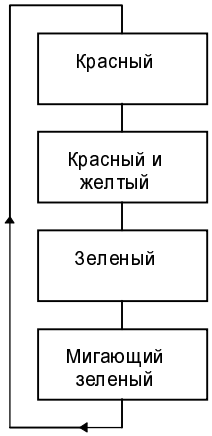



**231.** Написать программу, которая выводит круговую диаграмму, отражающую товарооборот (в процентах) книжного магазина. Исходные данные (объем продаж в рублях по категориям: книги, журналы, открытки и канцтовары) вводятся во время работы программы. Пример диаграммы приведен ниже.



**232.** Написать программу, которая выводит на экран изображение работающего светофора. Рекомендуемый вид светофора и алгоритм его работы приведены ниже.






### Факультатив

**233.** Написать программу, которая выводит на экран изображение идущих часов, у которых есть секундная и минутная стрелки.

**234.** Написать программу, которая выводит на экран график функции $y=2\ \mathrm{Sin}(x)\ e^{x/5}$.

# Файлы

## Общие замечания

Приступая к решению задач этого раздела, следует вспомнить, что:

- в программе, которая выполняет операции чтения из файла или запись в файл, должна быть объявлена переменная-указатель на тип `FILE`;
- для того, чтобы файл был доступен, его надо открыть, указав, для выполнения какого действия открывается файл: чтения, записи или обновления данных, а также тип файла (двоичный или текстовый);

- при работе с файлами возможны ошибки. Поэтому, рекомендуется при помощи функции `ferror` проверять результат выполнения потенциально опасных, с точки зрения возникновения ошибок, операций с файлами (`fopen`);
- чтение данных из текстового файла можно выполнить при помощи функции `fscanf`, запись — `fprintf`;
- по завершении работы с файлом файл нужно обязательно закрыть (функция `fclose`).

## Задачи

**235.** Напишите программу, которая на сменном диске компьютера (диск A:) создает файл numbers.txt и записывает в него 5 введенных пользователем целых чисел. Просмотрите при помощи редактора текста, например, встроенного в Norton Commander, созданный файл. Убедитесь, что каждое число находится в отдельной строке.

**236.** Напишите программу, которая дописывает в файл A:\numbers.txt пять введенных пользователем целых чисел. Убедитесь, при помощи редактора текста, что в файле находятся 10 чисел.

**237.** Напишите программу, которая выводит на экран содержимое файла A:\numbers.txt.

**238.** Напишите программу, которая вычисляет среднее арифметическое чисел, находящихся в файле A:\numbers.txt.

**239.** Напишите программу, которая позволяет просматривать текстовые файлы (выводит на экран содержимое файла), например, файлы исходных программ C++. Имя просматриваемого файла должно передаваться программе в качесве параметра, в командной строке, во время ее запуска.

**240.** Напишите программу, которая дописывает в находящийся на диске A: файл phone.txt имя, фамилию и номер телефона, например, вашего товарища. Если файла на диске нет, то программа должна создать его. В файле каждый элемент данных (имя, фамилия, телефон) должен находится в отдельной строке. Рекомендуемый вид экрана во время работы программы приведен ниже.

```
Добавление в телефонный справочник
Фамилия -> Сидоров
Имя -> Вася
Телефон -> 234-84-37
Информация добавлена.
Для завершения работы нажмите <Enter>
```

**241.** Напишите программу, которая позволяет за один сеанс работы добавить информацию о нескольких людях в файл A:\phone.txt. Рекомендуемый вид экрана во время работы программы приведен ниже.

```
Добавление в телефонный справочник
Для завершения вместо ввода фамилии нажмите <Enter>
Фамилия -> Сидоров
Имя -> Вася
Телефон -> 234-84-37
Информация добавлена.
Фамилия -> Орлов
Имя -> Андрей
Телефон -> 552-18-40
Информация добавлена.
Фамилия ->
Ввод завершен
Для завершения работы нажмите <Enter>
```

**242.** Напишите программу, которая позволяет найти в телефонном справочнике (A:\phone.txt) найти нужные сведения. Программа должна запрашивать фамилию человека и выводить его телефон. Если в справочнике есть люди с одинаковыми фамилиями, то программа должна вывести список всех этих людей. Рекомендуемый вид экрана во время работы программы приведен ниже.

```
Поиск в телефонном справочнике.
Введите фамилию и нажмите <Enter>. Для завершения работы с
программой сразу после приглашения нажмите <Enter>
->Петров
В справочнике данных о Петров нет.
->Иванов
Иванов Вася 578-12-45
Иванов Сергей 244-34-02
->
```

**243.** Напишите программу, которая объединяет возможности программ "Добавление в телефонный справочник" и "Поиск в телефонном справочнике". При запуске программы на экран должно выводиться меню, вид которого приведен ниже.

```
*** Телефонный справочник ***
1. Добавление
2. Поиск
3. Завершение работы

Введите номер пункта меню и нажмите <Enter>
->
```

## Факультатив

**244.** Напишите универсальную программу тестирования. Тест, последовательность вопросов и варианты ответов, должны находиться в текстовом файле. Имя файла теста программа должна получать из командной строки запуска программы. Количество вопросов теста не ограничено. Вместе с тем предлагается ввести следующее ограничение: текст вопроса и альтернативных ответов не должен занимать более одной строки экрана.

Программа должна выставлять оценку по следующему правилу: ОТЛИЧНО — за правильные ответы на все вопросы, ХОРОШО — если испытуемый правильно ответил не менее чем на 80% вопросов, УДОВЛЕТВОРИТЕЛЬНО — если правильных ответов более 60%, и ПЛОХО — если правильных ответов меньше 60%.

Ниже приведена рекомендуемая структура файла вопросов теста ($N_i$ — количество альтернативных ответов к i-ому вопросу, $K_i$, — номер правильного ответа), пример файла теста и вид экрана во время работы программы (номера ответов, введенные пользователем, выделены полужирным шрифтом).

```
Вопрос1
N1 M1
Ответ
...
Ответ
Вопрос2
N2 M2
```

```
Ответ
...
Ответ
Вопрос_k
N_k M_k
Ответ
...
Ответ

Архитектор Исаакиевского собора
3 2
Доменико Трезини
Огюст Монферран
Карл Росси
Архитектор Зимнего дворца
2 2
Франческо Бартоломео
Огюст Монферран
Невский проспект получил свое название
3 2
по имени реки, на которой стоит Санкт-Петербург.
по имени близко расположенного монастыря, Александро-Невской
лавры.
в память о знаменитом полководце — Александре Невском.
Сейчас Вам будет предложен тест.
К каждому вопросу дается несколько вариантов ответа.
Вы должны ввести номер правильного ответа и нажать клавишу
<Enter>

Архитектор Исаакиевского собора:
1. Доменико Трезини
2. Огюст Монферран
3. Карл Росси
->2

Архитектор Зимнего дворца:
1. Франческо Бартоломео
2. Карл Росси
->2

Невский проспект получил свое название:
```

```
1. по имени реки, на которой стоит Санкт-Петербург.
2. по имени близко расположенного монастыря, Александро-
Невской лавры.
3.в память о знаменитом полководце — Александре Невском.
->2

Ваша оценка ОТЛИЧНО!
Для завершения работы программы нажмите <Enter>
```

**245.** Напишите программу, которая выводит на экран список файлов и названия программ C++, находящихся в указанном при запуске программы каталоге. Предполагается, что первая строка программы является коментарием, содержащим информацию о назначении программы.

**246.** Напишите программу, которая по желанию пользователя выводит таблицу пересчета из дюймов в сантиметры на экран, принтер или в файл. Ниже приведен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом.

```
*** Таблица пересчета из дюймов в миллиметры ***
Результат выводить:
1 — на экран;
2 — на принтер;
3 — в файл.
Введите число от 1 до 3 и нажмите <Enter> '
Ваш выбор -> 2
```

| Дюймы | Миллиметры |
|---|---|
| 0.5 | 12.7 |
| 1.0 | 25.4 |
| 1.5 | 38.1 |
| 2.0 | 50.8 |
| 2.5 | 63.5 |
| 3.0 | 76.2 |
| 3.5 | 88.9 |
| 4.0 | 101.6 |
| 4.5 | 114.3 |
| 5.0 | 127.0 |

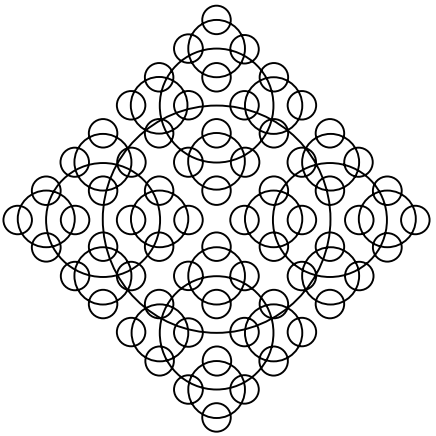
# Рекурсия

Приступая к решению задач этого раздела следует вспомнить:

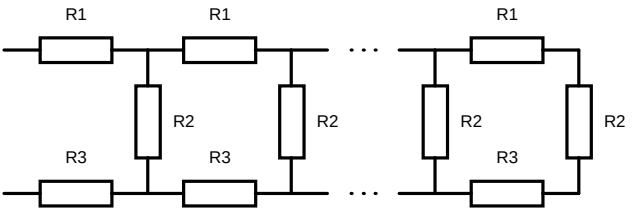
- рекурсивной называется такая функция, которая может вызывать сама себя;
- для завершения процесса рекурсии, в алгоритме рекурсивной функции обязательно должна быть веточка, обеспечивающая непосредственное завершение функции (процедуры).

**247.** Написать рекурсивную функцию вычисления факториала и программу, проверяющую ее работоспособность.

**248.** Написать программу, которая выводит на экран приведенный ниже узор.



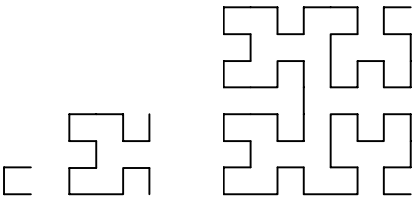
**249.** Написать программу, которая вычисляет сопротивление электрической цепи, схема которой приведена на рисунке. Величины сопротивлений и порядок цепи (количество сопротивлений R2) должны вводиться во время работы программы.



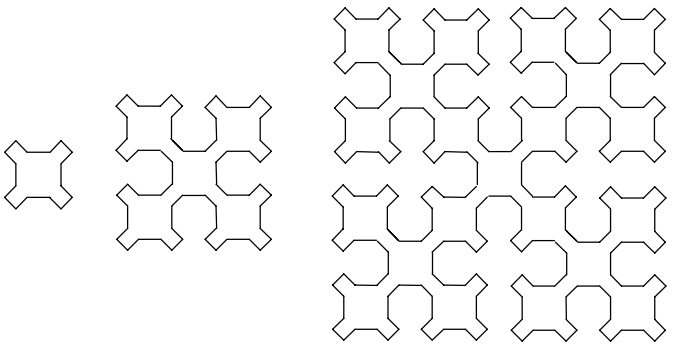
**250.** Напишите программу, которая вычерчивает на экране приведенную выше схему электрической цепи. Порядок цепи должен вводиться во время работы программы.

## Факультатив

**251.** Написать программу, которая вычерчивает на экране кривую Гильберта, показанную на рисунке. Обратите внимание, что кривая второго порядка получается путем соединения четырех кривых первого порядка, две из которых повернуты на 90 градусов: одна по, другая — против часовой стрелки. Аналогичным образом получается кривая третьего порядка, но при этом в качестве "кирпичиков" используются кривые второго порядка. Порядок вычерчиваемой кривой должен вводиться во время работы программы.



**252.** Написать программу, которая вычерчивает на экране кривую Серпинского. Порядок кривой должен вводиться во время работы программы. Вид кривых Серпинского первого, второго и третьего порядка приведен ниже.



# ЧАСТЬ II. РЕШЕНИЯ

## Задача 1

```
float a, b; // ширина и длина прямоугольника
float s;    // площадь прямоугольника
```

## Задача 2

```
float funt; // вес в фунтах
float kg;   // вес в килограммах
```

## Задача 3

```
float  summa;   // сумма вклада
int    srok;    // срок вклада (дней)
int    stavka;  // процентная ставка (годовых)
float  dohod;   // величина дохода
```

## Задача 5

```
float r1, r2; // внешний радиус и радиус отверстия
float s;      // площадь кольца
```

## Задача 7

```
float  CenaTetr;   // цена тетради
int    KolTetr;    // количество тетрадей
float  CenaKar;    // цена карандаша
int    KolKar;     // количество карандашей
float  CenaLin;    // цена линейки
float  Summa;      // стоимость покупки
```

## Задача 11

```
n++;
```

## Задача 12

```
counter -= 2;
```

## Задача 14

```
y:=-2.7*x*x*x + 0.23*x*x - 1.4;
```

## Задача 15

```
x += dx;
```

## Задача 16

```
kg = funt*0.4059;
```

## Задача 21

```
// константа M_PI, равная числу "ПИ", объявлена в файле
// math.h
s = M_PI * r * r;
```

## Задача 22

```
// константа M_PI, равная числу "ПИ", объявлена в файле
// math.h
s = 2*M_PI*r*(h+r);
v = M_PI *r*r*h;
```

## Задача 24

```
float r;     // радиус шара
float v, s;  // площадь поверхности и объем шара

v = (3*M_PI*r*r*r)/4; // константа M_PI объявлена в
s = 4*M_PI*r*r;       // файле math.h
```

## Задача 34

```
float ctetr, cobl, ckar; // цена тетради, обложки и карандаша
int   ntetr, nkar;       // кол-во тетрадей и карандашей
float summ;              // сумма покупки
```

```
// предполагается, что к каждой тетради
// покупается обложка
summ = ntetr*(ctetr+cobl) + nkar*ckar;
```

## Задача 37

```
#include <stdio.h>
#include <conio.h>
void main()
{
    printf("Файл stdio.h находится в каталоге");
    printf("c:\\borlandc\\include\n");
    printf("Для завершения нажмите <Enter>");
    getch(); // ждет нажатия клавиши
}
```

## Задача 38

```
// Выводит текст стихотворения
#include <stdio.h>
#include <conio.h>
void main()
{
   printf("Унылая пора! Очей очарованье!\n");
   printf("Приятна мне твоя прощальная краса -\n");
   printf("Люблю я пышное природы увяданье,\n");
   printf("В багрец и золото одетые леса.\n\n");
   printf("                  А.С.Пушкин\n");
   printf("\n\nДля завершения нажмите <Enter>");
   getch(); // чтобы стихотворение не исчезло с экрана
}
```

## Задача 39

```
printf("a=%5.3f    b=%5.3f    c=%5.3f", a, b, c);
```

## Задача 40

```
printf("высота = %3.2f см\nширина = %3.2f см\n", h, l);
```

## Задача 41

```
printf("a=%i  b=%i  c=%i", a, b, c);
```

## Задача 42

```
printf("a=%i\nb=%i\nc=%i\n", a, b, c);
```

## Задача 44

```
// Выводит текст стихотворения
#include <conio.h>
void main()
{
   textbackground(BLUE); // цвет фона
   textcolor(LIGHTGRAY); // цвет символов
   clrscr();             // очистить экран
   cprintf("Буря мглою небо кроет\n\r");
   cprintf("Вихри снежные крутя.\n\r");
   cprintf("То как зверь она завоет,\n\r");
   cprintf("То заплачет, как дитя.\n\n\r");
   cprintf("            А.С.Пушкин\n\n\r");
   cprintf("\n\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 45

```
// Выводит разноцветный текст
#include <conio.h>
void main()
{
   clrscr();
   textcolor(RED);
   cprintf("Каждый \n\r");
   textcolor(LIGHTRED); // оранжевый заменим алым
   cprintf("охотник \n\r");
   textcolor(YELLOW);
   cprintf("желает \n\r");
   textcolor(GREEN);
   cprintf("знать \n\r");
   textcolor(LIGHTBLUE);
```

```
   cprintf("где \n\r");
   textcolor(BLUE);
   cprintf("сидят \n\r");
   textcolor(MAGENTA);
   cprintf("фазаны!\n\r");
   textcolor(LIGHTGRAY);
   cprintf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 47

```
scanf("%f", &u);
scanf("%f", &r);
```

## Задача 48

```
scanf("%f %f", &u, &r);
```

## Задача 49

```
// Объявление переменных
float r, h; // радиус и высота цилиндра
float v;    // объем цилиндра

// фрагмент программы
printf("Введите исходные данные:\n");
printf("Радиус цилиндра ->");
scanf("%f", &r);
printf("Высота цилиндра ->");
scanf("%f", &h);
```

## Задача 50

```
float ctetr, ckar; // цена тетради и карандаша
int ntetr, nkar;   // количество тетрадей и карандашей
printf("Введите цену и количество (в одной строке)\n");
printf("Тетради ->");
scanf("%f %i", &ctetr, &ntetr)
printf("Карандаши ->");
scanf("%f %i", &ckar, &nkar)
```

## Задача 51

```
// Вычисление площади прямоугольника
#include <stdio.h>
#include <conio.h>
void main()
{
  float l,w; // длина и ширина прямоугольника
  float s;   // площадь прямоугольника

  printf("\nВычисление площади прямоугольника\n");
  printf("Введите исходные данные:\n");
  printf("Длина (см.)  -> ");
  scanf("%f", &l);
  printf("Ширина (см.) -> ");
  scanf("%f", &w);
  s = l * w;
  printf("Площадь параллелограмма: %10.2f кв.см.\n", s);

  printf("\n\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 53

```
// Вычисление площади поверхности параллелепипеда
#include <stdio.h>
#include <conio.h>
void main()
{
  float l,w,h; // длина, ширина и высота параллелепипеда
  float s;     // площадь поверхности параллелепипеда

  printf("\nВычисление площади поверхности");
  printf("параллелепипеда\n");
  printf("Введите исходные данные:\n");
  printf("Длина (см)  -> ");
  scanf("%f", &l);
  printf("Ширина (см) -> ");
  scanf("%f", &w);
  printf("Высота (см) -> ");
```

```
  scanf("%f", &w);
  s = (l*w + l*h + w*h)*2;
  printf("Площадь поверхности: %6.2f кв.см\n",s);
  printf("\n\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 55

```
// Вычисление объема цилиндра
#include <stdio.h>
#include <conio.h>
void main()
{
   float r,h,v; // радиус основания, высота и объем цилиндра

   printf("Вычисление объема цилиндра\n");
   printf("Введите исходные данные:\n");
   printf("Радиус основания (см) -> ");
   scanf("%f", &r);
   printf("Высота цилиндра (см) -> ");
   scanf("%f", &h);
   v = 2*3.1415926*r*r*h;
   printf("\nОбъем цилиндра %6.2f куб.см\n", v);

   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 56

```
// Вычисление стоимости покупки
#include <stdio.h>
#include <conio.h>
void main()
{
   float kar,tetr; // цена карандаша и тетради
   int nk,nt;      // количество тетрадей и карандашей
   float summ;     // стоимость покупки }

   printf("\nВычисление стоимости покупки\n");
```

```
   printf("Введите исходные данные:\n");
   printf("Цена тетради (руб.) -> ");
   scanf("%f", &tetr);
   printf("Количество тетрадей -> ");
   scanf("%i", &nt);
   printf("Цена карандаша (руб.) -> ");
   scanf("%f", &kar);
   printf("Количество карандашей -> ");
   scanf("%i", &nk);
   summ=tetr*nt + kar*nk;
   printf("\nСтоимость покупки: %6.2f руб.\n", summ);

   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 60

```
// Вычисление площади треугольника по двум
// сторонам и величине угла между ними
#include <stdio.h>
#include <conio.h>
#include "math.h"   // sin и константа M_PI - число "ПИ"
void main()
{
    float a,b; // длины сторон
    float u;   // величина угла, выраженная в градусах
    float s;   // площадь треугольника

    printf("\nВычисление площади треугольника\n");
    printf("Введите в одной строке длины сторон ");
    printf("(см) -> ");
    scanf("%f%f", &a, &b);
    printf("Введите величину угла между сторонами ");
    printf("(град.) -> ");
    scanf("%f", &u);
    /* s=a*h/2, где a - основание, h - высота.
       h - может быть вычислена по формуле h=b*sin(u).
       Аргумент функции sin должен быть выражен в радианах.
       1 рад. = 180/pi, где pi - число "ПИ").
    */
```

```
    s = a*b*sin(u*M_PI/180)/2;
    printf("Площадь треугольника: %6.2f кв.см",s);

    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 61

```
// Вычисление сопротивления электрической цепи,
//  состоящей из двух параллельно соединенных элементов.
#include <stdio.h>
#include <conio.h>
void main()
{
  float r1,r2; // сопротивление элементов цепи
  float r;     // суммарное сопротивление цепи

  printf("\nВычисление сопротивления электрической цепи\n");
  printf("при параллельном соединении элементов\n");
  printf("Введите исходные данные:\n");
  printf("Величина первого сопротивления (Ом) -> ");
  scanf("%f",&r1);
  printf("Величина второго сопротивления (Ом) -> ");
  scanf("%f",&r2);
  r=r1*r2/(r1+r2);

  printf("Сопротивление цепи: %6.2f Ом",r);

  printf("\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 65

```
// Вычисление стоимости поездки на дачу и обратно
#include <stdio.h>
#include <conio.h>
void main()
{
    float rast;  // расстояние до дачи
```

```
    float potr;  // потребление бензина на 100 км. пути
    float cena;  // цена одного литра бензина
    float summ;  // стоимость поездки на дачу и обратно

    printf("\nСтоимость поездки на дачу и обратно\n");
    printf("Расстояние до дачи (км) -> ");
    scanf("%f",&rast);
    printf("Расход бензина (литров на 100 км.) -> ");
    scanf("%f",&potr);
    printf("Цена литра бензина (руб.) -> ");
    scanf("%f",&cena);
    summ = 2 * potr/100 * rast * cena;
    printf("Поездка на дачу и обратно обойдется");
    printf("в %6.2f руб.",summ);

    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 66

```
// Скорость бега
#include <stdio.h>
#include <conio.h>
void main()
{
   float s;   // дистанция
   float t;   // время
   float v;   // скорость
   int min;   // минут
   int sek;   // секунд
   float  ts; // время в секундах

   printf("Вычисление скорости бега\n");
   printf("Введите длину дистанции (метров) -> ");
   scanf("%f", &s);
   printf("Введите время (минут. секунд)-> ");
   scanf("%f", &t);

   min = t;
   sek = (t - min) * 100;
   ts = min * 60 + sek;
```

```
    v = (s /1000) / (ts / 3600);

    printf("Дистанция: %4.0f м\n", s);
    printf("Время: %i мин %i сек = %4.0f сек\n", min, \
    sek, ts);
    printf("Вы бежали со скоростью %2.2f км/час\n", v);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 68

```
// Вычисление площади поверхности цилиндра
#include <stdio.h>
#include <conio.h>
#include "math.h"  // константа M_PI - число "ПИ"
void main()
{
    float r; // радиус основания цилиндра
    float h; // высота цилиндра
    float s; // площадь поверхности цилиндра

    printf("\nВычисление площади поверхности цилиндра\n");
    printf("Введите исходные данные:\n");
    printf("радиус основания цилиндра (см) ->");
    scanf("%f", &r);
    printf("высота цилиндра (см) ->");
    scanf("%f", &h);
    s = 2*M_PI*r*r +2*M_PI*r*h;
    printf("Площадь поверхности цилиндра %6.2f кв.см\n", s);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 70

```
// Пересчет расстояния из верст в километры
#include <stdio.h>
#include <conio.h>
```

```
void main()
{
   float v; // расстояние в верстах
   float k; // расстояние в километрах

   printf("\nПересчет расстояния из верст в километры\n");
   printf("Введите расстояние в верстах ->");
   scanf("%f", &v);
   k = v*1.0668;
   printf("%6.2f версты - это %6.2f км\n", v,k);
   printf("\nДля завершения нажмите <Enter>");

   getch();
}
```

## Задача 72

```
// Вычисление дохода по вкладу
#include <stdio.h>
#include <conio.h>
void main()
{
  float summ;    // сумма вклада
  int srok;      // срок вклада
  float stavka;  // процентная ставка
  float dohod;   // доход по вкладу

  printf("\nВычисление дохода по вкладу\n");
  printf("Введите исходные данные:\n");
  printf("Величина вклада (руб.) -> ");
  scanf("%f", &summ);
  printf("Срок вклада (дней) -> ");
  scanf("%i", &srok);
  printf("Процентная ставка (годовых) -> ");
  scanf("%f", &stavka);
  dohod = summ * stavka/365/100 * srok; // 365 - кол-во
                                        // дней в году
  summ = summ + dohod;

  printf("---------------------------------\n");
  printf("Доход: %9.2f руб.\n", dohod);
```

```
  printf("Сумма по окончании срока вклада: \
  %9.2f руб.\n",summ);

  printf("\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 73

```
// Преобразование величины, выраженной в минутах,
// в значение, выраженное в часах и минутах
#include <stdio.h>
#include <conio.h>
void main()
{
  int min; // интервал в минутах
  int h;   // количество часов
  int m;   // количество минут

  printf("Введите временной интервал (в минутах) -> ");
  scanf("%i",&min);
  h = (int)min /  60;
  m = min % 60;
  printf("%i мин. - это %i час.%i мин.\n", min, h, m);

  printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 74

```
// Преобразование числа в денежный формат
#include <stdio.h>
#include <conio.h>
void main()
{
    float f;     // дробное число
    int r;       // целая часть числа (рубли)
    int k;       // дробная часть числа (копейки)

    printf("\nПреобразование числа в денежный формат\n");
```

```
    printf("Введите дробное число -> ");
    scanf("%f",&f);

    r =  (int)f;
    k = f * 100 - r*100;
    printf("%6.2f руб. - это %i руб. %i коп.\n", f, r, k);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 77

```
// Вычисление частного
#include <stdio.h>
#include <conio.h>
void main()
{
    float a,b,c; // делимое, делитель и частное

    printf("\nВычисление частного\n");
    printf("Введите в одной строке делимое и делитель, ");
    printf("затем нажмите <Enter>");
    printf("-> ");
    scanf("%f%f", &a, &b);
    if (b != 0)
    {
       c = a / b;
       printf("частное от деления %5.2f на %5.2f ", a, b);
       printf("равно %5.2f", c);
    }
    else {
       printf("Ошибка! Делитель не должен быть равен");
       printf("нулю!\n");
    }
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 78

```
// Вычисление площади кольца
#include <stdio.h>
#include <conio.h>
void main()
{
     float r1,r2; // радиус кольца и отверстия
     float s;     // площадь кольца

     printf("\nВведите исходные данные:\n");
     printf("радиус кольца (см) -> ");
     scanf("%f",&r1);
     printf("радиус отверстия (см) -> ");
     scanf("%f",&r2);
     if (r1 > r2)
      {
        s = 2 * 3.14 * (r1 - r2);
        printf("\nПлощадь кольца %6.2f кв.см\n", s);
      }
     else
      {
        printf("\nОшибка! Радиус отверстия не может быть");
        printf("больше радиуса кольца.\n");
      }
     printf("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 79

```
// Перевод времени из минут и секунд в секунды
#include <stdio.h>
#include <conio.h>
void main()
{
   float t;     // время в минутах и секундах, например 1.25
   int ts;      // время в секундах
   int min;     // число минут
   int sek;     // число секунд
```

```
   printf("Введите время (минут.секунд) -> ");
   scanf("%f", &t);
   min = t;  // t типа float, поэтому кол-во секунд
             // "усекается"
   sek = (t - min) * 100;
   if (sek > 60)
    {
      printf("Ошибка!");
      printf("Количество секунд не может быть больше 60");
    }
   else
    {
      ts = min * 60 + sek;
      printf("%i мин %i сек = %i сек", min, sek, ts);
    }
   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 80

```
// Проверяет, является ли год високосным
#include <stdio.h>
#include <conio.h>
void main()
{
   int year;
   int r;    // остаток от деления year на 4

   printf("Введите год, например 2001, и нажмите <Enter>");
   printf("->");
   scanf("%i", &year);
   r = year % 4;
   if ( r )
    printf("%i год - невисокосный\n", year);
   else
    printf("%i год - високосный\n", year);

   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 81

```
// Вычисление сопротивления электрической цепи
#include <stdio.h>
#include <conio.h>
void main()
{
  float r1,r2; // величины сопротивлений цепи
  float r;     // суммарное сопротивление
  int t;       // тип соединения элементов:
               //    1 - последовательное;
               //    2 - параллельное
  printf("\nВычисление сопротивления электрической цепи\n");
  printf("Введите исходные данные:\n");
  printf("Величина первого сопротивления (Ом) ->");
  scanf("%f", &r1);
  printf("Величина второго сопротивления (Ом) ->");
  scanf("%f", &r2);
  printf("Тип соединения элементов ");
  printf("(1-последовательное, 2-параллельное) ->");
  scanf("%i", &t);
  if (t == 1)
       r = r1 + r2;
  else r = r1*r2 / (r1+r2);
  printf("Сопротивление цепи: %6.2f Ом\n", r);

  printf("\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 82

```
// Решение квадратного уравнения
#include <stdio.h>
#include <conio.h>
#include "math.h"
void main()
{
   float a,b,c;    // коэффициенты уравнения
   float x1,x2;    // корни уравнения
   float d;    // дискриминант
```

```
   printf("\n* Решение квадратного уравнения *\n");
   printf("Введите в одной строке значения коэффициентов");
   printf(" и нажмите <Enter>");
   printf("-> ");
   scanf("%f%f%f", &a, &b, &c);    // ввод коэффициентов
   d = b*b - 4*a*c;                // дискриминант
   if (d < 0)
      printf("Уравнение не имеет решения\n");
   else {
      x1 = (-b + sqrt(d))/(2*a);
      x2 = (-b - sqrt(d))/(2*a);
      printf("Корни уравнения: x1=%3.2f x2=%3.2f\n", x1, x2);
   }

   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 84

```
// Вычисление стоимости покупки с учетом скидки
#include <stdio.h>
#include <conio.h>
void main()
{
  float summ; // сумма покупки

  printf("\nВычисление стоимости покупки с учетом скидки\n");
  printf("Введите стоимость покупки и нажмите <Enter>");
  printf("-> ");
  scanf("%f", &summ);
  if (summ < 500)
      printf("Скидка не предоставляется.\n");
  else {
      printf("Вам предоставляется скидка ");
      if (summ > 1000) {
      printf("5%\n");
      summ = 0.97 * summ;
      }
      else {
          printf("3%\n");
```

```
          summ = 0.97 * summ;
      };
      printf("Сумма с учетом скидки: %3.2f руб.\n", summ);
  }

  printf("\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 85

```
// Проверка знания истории
#include <stdio.h>
#include <conio.h>
void main()
{
  int year;   // ответ испытуемого

  printf("\nВ каком году был основан Санкт-Петербург?\n");
  printf("Введите число и нажмите <Enter>");
  printf("-> ");
  scanf("%i", &year);
  if (year == 1703)
      printf("Правильно.");
  else {
      printf("Вы ошиблись, ");
      printf("Санкт-Петербург был основан в 1703 году.\n");
  }

  printf("\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 87

```
// Проверка знания истории архитектуры
#include <stdio.h>
#include <conio.h>
void main()
{
   int otv; // номер выбранного варианта ответа
```

```
   printf("Архитектор Исаакиевского собора:\n");
   printf("1. Доменико Трезини\n");
   printf("2. Огюст Монферран\n");
   printf("3. Карл Росси\n");

  printf("Введите номер ответа и нажмите <Enter>");
   printf("-> ");
   scanf("%i", &otv);
   if (otv == 2)
       printf("Правильно.");
   else {
       printf("Вы ошиблись.\n Архитектор Исаакиевского ");
       printf("собора Огюст Монферран.\n");
   }
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 89

```
// Сравнение двух целых чисел
#include <stdio.h>
#include <conio.h>
void main()
{
   int a,b; // сравниваемые числа

   printf("\nВведите в одной строке два целых ");
   printf("числа и нажмите <Enter>");
   printf("->");
   scanf("%i%i", &a, &b);
   if (a == b)
       printf("Числа равны");
   else if (a < b)
          printf("%i меньше %i\n", a, b);
        else  printf("%i больше %i\n", a, b);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 90

```
// Проверка умения умножать числа
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>  // для доступа к srand
#include <time.h>    // для доступа к time
void main()
{
  int m1, m2, p; // сомножители и произведение
  int otv;       // ответ испытуемого
  time_t t;      // текущее время - для инициализации
                 // генератора случайных чисел

  srand((unsigned) time(&t));; // инициализация генератора
                               // случ. чисел

  m1 = rand() % 9 +1;  // остаток от деления rand() на 9
                       // лежит в диапазоне от 0 до 8
  m2 = rand() % 9 +1;
  p = m1 * m2;
  printf("Сколько будет %ix%i ?\n", m1, m2);
  printf("Введите ответ и нажмите <Enter>");
  printf("-> ");
  scanf("%i", &otv);
  if (p == otv)
      printf("Правильно.");
  else
      printf("Вы ошиблись.\n%ix%i=%i", m1, m2, p);

  printf("\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 92

```
// Проверяет на четность введенное с клавиатуры число
#include <stdio.h>
#include <conio.h>
void main()
{
    int n; // введенное пользователем число
```

```
    printf("\nВведите целое число и нажмите <Enter>");
    printf("-> ");
    scanf("%i", &n);
    printf("Число %i ");
    if (n % 2 == 0)
       printf("четное.");
    else
       printf("нечетное.");

    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 94

```
// Вычисление стоимости телефонного разговора с учетом
// скидки, предоставляемой по субботам и воскресеньям
#include <stdio.h>
#include <conio.h>
void main()
{
     int time;     // длительность разговора
     int day;      // день недели
     float summa;  // стоимость разговора

     printf("\nВычисление стоимости разговора по");
     printf("телефону\n");
     printf("Введите исходные данные:\n");
     printf("Длительность разговора ");
     printf("(целое кол-во минут) ->");
     scanf("%i", &time);
     printf("День недели");
     printf(" (1-понедельник,..,7-воскресенье) ->");
     scanf("%i", &day);
     summa = 2.3 * time;       // цена минуты 2.3 руб.
     if (day == 6 || day == 7)
      {
        printf("Предоставляется скидка 20%\n");
        summa = summa * 0.8;
       };
     printf("Стоимость разговора: %3.2f руб.\n",summa);
```

```
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 95

```
// Контроль веса
#include <stdio.h>
#include <conio.h>
void main()
{
     float w;   // вес
     float h;   // рост
     float opt; // оптимальный вес
     float d;   // отклонение от оптимального веса

     printf("\Введите в одной строке, через пробел,\n");
     printf("рост (см) и вес (кг), затем нажмите <Enter>");
     printf("->");
     scanf("%f%f", &h, &w);
     opt = h - 100;
     if (w == opt)
       printf("Ваш вес оптимален!");
     else
      if (w < opt)
       {
         d = opt - w;
         printf("Вам надо поправиться на %2.2f кг.\n", d);
       }
      else
       {
         d = w - opt;
         printf("Вам надо похудеть на %2.2f кг.\n", d);
       }

     printf("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 96

```
// Определение времени года по номеру месяца
#include <stdio.h>
#include <conio.h>
void main()
{
   int month; // номер месяца

   puts("\nВведите номер месяца (число от 1 до 12)");
   printf("-> ");
   scanf("%i", &month);
   if (month < 1 && month > 12)
       printf("Число должно быть от 1 до 12");
   else if (month >= 3 && month <= 5)
            printf("Весна");
    else if (month >= 6 && month <= 8)
             printf("Лето");
         else if (month >= 9  && month <= 11)
                  printf("Осень");
              else printf("Зима");

    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 98

```
// Дописывает после числа слово "рубль" в правильной форме
#include <stdio.h>
#include <conio.h>
void main()
{
    int n; // число
    int r; // сначала остаток от деления n на 100 (последние
           // две цифры), затем - на 10 (последняя цифра)

    printf("\nВведите целое число, не больше 999 -> ");
    scanf("%i", &n);
    printf("%i ", n);
    // правильная форма слова определяется последней
```

```
    // цифрой, за исключением чисел от 11 до 14
    if (n > 100)
       r  =  n % 100;
    else r = n;

    // здесь r - последние две цифры
    if ( r >= 11 && r <= 14 )
       printf("рублей\n");
    else
    {
        r = r % 10;
        // здесь r - последняя цифра
        if ( r >= 2 && r <= 4 )
            printf("рубля\n");
        else if (r == 1)
                 printf("рубль\n");
             else printf("рублей\n");
    }
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 100

```
// Вычисление даты следующего дня
#include <stdio.h>
#include <conio.h>
void main()
{
    int day;
    int month;
    int year;
    int last;  // 1, если текущий день — последний день месяца
    int r;     // если год високосный, то остаток от
               // деления year на 4 равен нулю

    printf("Введите в одной строке (цифрами) ");
    printf("сегодняшнюю дату\n");
    printf("(число месяц год) -> ");
    scanf("%i%i%i", &day, &month, &year);
```

```
    last = 0;
    if (month == 2) {
         if ((year % 4) != 0 && day == 28) last = 1;
         if ((year % 4) == 0 && day == 29) last = 1;
    }
    else if ((month == 4 || month == 6 ||
              month == 9 || month == 11)
                  && (day == 31))
         last = 1;
      else if (day == 31)
            last = 1;

    if (last == 1) {
      printf("Последний день месяца!\n");
      day = 1;
      if (month == 12) {
         month = 1;
         year++;
         printf("С наступающим Новым годом!\n");
      }
      else month++;
     }
     else day++;
     printf("Завтра %i %i %i",day,month,year);

     printf("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 101

```
// Выводит название дня недели
#include <stdio.h>
#include <conio.h>
void main()
{
   int nd; // номер дня недели

   puts("\nВведите номер дня недели (1..7");
   printf("->");
```

```
   scanf("%i", &nd);
   switch (nd)
   {
        case 1:  puts("Понедельник"); break;
        case 2:  puts("Вторник");     break;
        case 3:  puts("Среда");       break;
        case 4:  puts("Четверг");     break;
        case 5:  puts("Пятница");     break;
        case 6:  puts("Суббота");     break;
        case 7:  puts("Воскресенье"); break;

        default: puts("Число должно быть в диапазоне 1..7");
   }
   getch();
}
```

## Задача 102

```
// Определение стоимости междугородного
// телефонного разговора
#include <stdio.h>
#include <conio.h>
void main()
{
    int kod;     // код города
    float cena;  // цена минуты
    int dlit;    // длительность разговора
    float summ;  // стоимость разговора

    printf("\nВычисление стоимости разговора по");
    printf("телефону.\n");
    printf("Введите исходные данные:\n");
    printf("Длительность разговора (целое кол-во минут) ->");
    scanf("%i", &dlit);
    puts("Код города");
    puts("Владивосток\t432");
    puts("Москва\t\t095");
    puts("Мурманск\t815");
    puts("Самара\t\t846");
    printf("->");
    scanf("%i", &kod);
```

```
    printf("Город: ");
    switch (kod)
    {
      case 432: puts("Владивосток");
                cena = 2.2;
                break;
      case 95:  puts("Москва");
                cena = 1;
                break;
      case 815: puts("Мурманск");
                cena = 1.2;
                break;
      case 846: puts("Самара");
                cena = 1.4;
                break;
      default:  printf("неверно введен код.");
                cena = 0;
    }
    if (cena != 0) {
       summ = cena * dlit;
       printf("Цена минуты: %i руб.\n", cena);
       printf("Стоимость разговора: %3.2f руб.\n", summ);
    }
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 103

```
// По дате определяет день недели
#include <stdio.h>
#include <conio.h>
void main()
{
    int day,month,year; // день, месяц, год

    int c,y;            // столетие и год в столетии
    int m;              // месяц по древнеримскому календарю
    int d;              // день недели
```

```
    puts("\nОпределение дня недели по дате");
    puts("Введите дату: день месяц год.");
    puts("Например, 5 12 2001");
    printf("->");
    scanf("%i %i %i", &day, &month, &year);

    if (month == 1 || month == 2)
         year--;     // январь и февраль относятся
                     // к предыдущему году

    m = month - 2;       // год начинается с марта
    if (m <= 0) m += 12; // для января и февраля
    // здесь m - номер месяца по римскому календарю
    c = year / 100;
    y = year - c*100;

    d = (day+(13*m-1)/5+y+y/4+c/4-2*c+777)%7;

    switch (d)
    {
      case 1: puts("Понедельник"); break;
      case 2: puts("Вторник");     break;
      case 3: puts("Среда");       break;
      case 4: puts("Четверг");     break;
      case 5: puts("Пятница");     break;
      case 6: puts("Суббота");     break;
      case 0: puts("Воскресенье");
    }
    printf("\nДля завершения нажмите <Enter>\n");

    getch();
}
```

## Задача 105

```
// Выводит таблицу квадратов нечетных чисел
#include <stdio.h>
#include <conio.h>
void main()
{
```

```
    int x = 1;  // число
    int y;      // квадрат числа
    int i;      // счетчик циклов

    printf("Таблица квадратов\n");
    printf("-----------------\n");
    printf("Число\tКвадрат\n");
    printf("-----------------\n");
    for (i = 1; i <= 10; i++)
    {
       y = x*x;
       printf("%3i\t%4i\n", x, y);
       x += 2;
    }
    printf("-----------------\n");

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 107

```
// Вычисляет сумму первых n целых положительных чисел
#include <stdio.h>
#include <conio.h>
void main()
{
    int n;     // кол-во суммируемых чисел
    int summ;  // сумма
    int i;     // счетчик циклов

    printf("Вычисление суммы положительных чисел\n");
    printf("Введите количество суммируемых чисел -> ");
    scanf("%i", &n);
    summ = 0;
    for (i = 1; i <= n; i++)
       summ = summ+i;
    printf("Сумма первых %i целых положительных чисел ",n);
    printf("равна %i", summ);
```

```
    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 109

```
// Вычисляет частичную сумму ряда: 1,3,6,9 ...
#include <stdio.h>
#include <conio.h>
void main()
{
    int e;        // член ряда
    int n;        // кол-во суммируемых членов
    int summ = 0; // частичная сумма ряда
    int i;        // счетчик циклов

    printf("Вычисление частичной суммы ряда: ");
    printf("1,3,6,9, ...\n");
    printf("Введите количество суммируемых членов -> ");
    scanf("%i", &n);
    e = 1;
    for (i = 1; i <= n; i++)
    {
        summ += e;
        e += 2;
    }
    printf("Сумма первых %i членов ряда равна %i", n, summ);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 110

```
// Вычисление суммы ряда 1+1/2+1/3+ ...
#include <stdio.h>
#include <conio.h>
void main()
{
     int n;     // кол-во суммируемых членов ряда
```

```
    float i;   // номер элемента ряда. Если объявить как
               // int, то при вычислении 1/i будет выполнено
               // усечение дробной части

    float elem;      // значение элемента ряда
    float summ = 0 ; // сумма элементов ряда

    printf("Вычисление частичной суммы ряда");
    printf("1+1/2+1/3+...\n");
    printf("Введите кол-во суммируемых членов ряда\n");
    printf("-> ");
    scanf("%i",&n);
    summ = 0;
    for (i = 1; i <= n; i++) {
       elem = 1 / i;
       summ += elem;
    }
    printf("Сумма первых %i", n);
    printf(" членов ряда равна %6.3f",summ);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 111

```
// Таблица степеней двойки
#include <stdio.h>
#include <conio.h>
void main()
{
    int n;  // показатель степени
    int x;  // значение 2 в степени n

    printf("\nТаблица степеней двойки\n");
    x = 1;
    for (n = 0; n <= 10; n++)
    {
       printf("%3i%5i\n", n, x);
       x *= 2;
    }
```

```
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 113

```
// Таблица функции
#include <stdio.h>
#include <conio.h>

#define  LB -2.0   // нижняя граница диапазона изменения
                   // аргумента
#define  HB  2.0   // верхняя граница диапазона изменения
                   // аргумента
#define  DX  0.5   // приращение аргумента

void main()
{
    float x,y;  // аргумент и значение функции
    int n;      // кол-во точек
    int i;      // счетчик циклов

    n = (HB - LB)/DX +1;
    x = LB;
    printf("------------------\n");
    printf("   x    |   y\n");
    printf("------------------\n");
    for (i = 1; i<=n; i++)
    {
       y = -2.4*x*x+5*x-3;
       printf("%6.2f  | %6.2f\n" ,x ,y);
       x += DX;
    }
    printf("------------------\n");

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 116

```
// Среднее арифметическое дробных чисел, вводимых
// с клавиатуры
#include <stdio.h>
#include <conio.h>

#define L 5 // количество чисел последовательности

void main()
{
    float a;     // число
    int n;       // кол-во введенных чисел
    float sum;   // сумма введенных чисел
    float sred;  // среднее арифметическое введенных чисел

    printf("\nОбработка последовательности дробных чисел\n");
    printf("После ввода каждого числа нажимайте <Enter>");
    sum = 0;
    for (n = 1; n <= L; n++)
    {
      printf("-> ");
      scanf("%f", &a);
      sum += a;
      printf("Введено чисел: %i ", n);
      printf("Сумма: %6.2f\n", sum);
    }

    sred = sum / L;
    printf("Сред. арифметическое: %6.2f\n", sred);
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 117

```
// Вычисляет среднее арифметическое и определяет
// минимальное и максимальное число последовательности
// дробных чисел, вводимых с клавиатуры
#include <stdio.h>
#include <conio.h>
```

```
void main()
{
    float a;     // очередное число
    int n;       // количество чисел
    float sum;   // сумма введенных чисел
    float sred;  // среднее арифметическое
    float min;   // минимальное число последовательности
    float max;   // максимальное число последовательности
    int i;       // счетчик циклов

    printf("Обработка последовательности дробных чисел.\n");
    printf("Введите количество чисел последовательности ->");
    scanf("%i", &n);
    printf("Введите последовательность.\n");
    printf("После ввода каждого числа нажимайте <Enter>");
    printf("->");
    scanf("%f",&a);  // вводим первое число
                     // последовательности
    // предположим, что:
    min = a; // пусть первое число является минимальным
    max = a; // пусть первое число является максимальным
    sum = a;
    // введем остальные числа
    for (i = 1; i < n; i++)
    {
       printf("->");
       scanf("%f", &a);
       sum += a;
       if (a < min)  min = a;
       if (a > max)  max = a;
     }
     sred = sum / n;
     printf("Количество чисел: %i\n", n);
     printf("Среднее арифметическое: %6.2f\n", sred);
     printf("Минимальное число: %6.2f\n", min);
     printf("Максимальное число: %6.2f\n", max);

     printf("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 119

```
// вычисление среднего арифметического случайных
// последовательностей
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>  // для доступа к srand и rand
#include <time.h>

#define L 10  // длина последовательности
#define N 3   // количество последовательностей

void main()
{
    int r;         // случайное число
    int sum;       // сумма чисел последовательности
    float sred;    // среднее арифметическое
    int i,j;       // счетчики циклов
    time_t t;      // текущее время - для инициализации
                   // генератора случайных чисел

    srand((unsigned) time(&t));; // инициализация генератора
                                 // случайных чисел
    for (i = 1; i <= N; i++)
    {
       // генерируем последовательность
       printf("\nСлучайные числа: ");
       sum = 0;  // не забыть обнулить !
       for (j = 1; j <= L; j++)
       {
          r = rand() % 10 +1 ;
          printf("%i ", r);
          sum += r;
       }
       sred = (float)sum / L;  // чтобы не было усечения
       printf("\nСред.арифм.: %3.2f\n", sred);
    }

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 121

```
// Таблица функции y=|x|
#include <stdio.h>
#include <conio.h>
#include "math.h"

#define  LB -4 // нижняя граница диапазона изменения
               // аргумента
#define  HB  4 // верхняя граница диапазона изменения
               // аргумента
#define  DX  0.5  // приращение аргумента

void main()
{
    float x,y;  // аргумент и значение функции
    int n;      // кол-во точек
    int i;      // счетчик циклов

    printf("\nТаблица значений функции y=|x| \n");
    n = (HB - LB)/DX + 1;
    x = LB;
    for (i = 1; i <= n; i++)
    {
       y = fabs(x);
       printf("%4.2f   %3.2f\n", x, y);
       x += DX;
    }

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 123

```
// Выводит таблицу умножения на 7
#include <stdio.h>
#include <conio.h>
void main()
{
    int m;  // число, для которого надо вывести
            // таблицу умножения (множимое)
```

```
    int n;  // множитель
    int p;  // произведение

    m = 7;
    printf("\nТаблица умножения на %i\n", m);
    for (n = 1; n<=9; n++)
    {
       p = m * n;
       printf("%ix%i=%i\n", m, n, p);
    }

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 124

```
// Квадрат Пифагора - таблица умножения
#include <stdio.h>
#include <conio.h>
void main()
{
    int i,j; // номер строки и столбца таблицы

    printf("    ");          // левая верхняя клетка таблицы
    for (j = 1; j <=10; j++) // первая строка - номера
                             // столбцов
    printf("%4i",j);
    printf("\n");

    for (i = 1; i <=10; i++)
    {
       printf("%4i",i);             // номер строки
       for (j = 1; j <= 10; j++)    // строка таблицы
           printf("%4i",i*j);
       printf("\n");
    }

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 125

```
// Вычисление числа "ПИ" с использованием
// свойства ряда 1 -1/3 + 1/5 - 1/7 + ...
#include <stdio.h>
#include <conio.h>
void main()
{

  float x;      // член ряда
  int n;        // количество суммируемых членов
  float summ;   // частичная сумма
  int i;        // счетчик циклов

  // при суммировании достаточно большого (~1000) количества
  // элементов ряда, значение суммы стремится к "ПИ"/4
  printf("Вычисление суммы ряда 1 -1/3 + 1/5 - 1/7 + ...\n");
  printf("Введите кол-во суммируемых членов ряда ->" );
  scanf("%i", &n);
  summ = 0;
  for (i = 1; i <= n; i++)
  {
      x = (float)1/(2*i - 1);

      if ((i % 2) == 0) x = -1 * x;
      summ += x;
  }
  printf("Сумма ряда: %2.6f\n", summ);
  printf("Вычисленное значение ");
  printf("числа ПИ = %2.6f\n", summ * 4);

  printf("\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 126

```
// Приближенное вычисление интеграла
// методом прямоугольников (цикл for)
#include <stdio.h>
#include <conio.h>
```

```
void main()
{
   float a,b;   // границы отрезка
   float dx;    // приращение аргумента
   float s;     // приближенное значение интеграла
   int n;       // количество интервалов
   float x;     // аргумент
   float y;     // значение функции в начале интервала
   int i;

   printf("\Приближенное вычисление интеграла\n");
   printf("Нижняя граница интервала -> ");
   scanf("%f", &a);
   printf("Верхняя граница интервала -> ");
   scanf("%f", &b);
   printf("Приращение аргумента -> ");
   scanf("%f", &dx);
   n = (b - a) / dx + 1;
   x = a;
   s = 0;
   for (i = 1; i<=n; i++)
   {
      y = x*x + 2;  // значение функции в начале интервала
      s += y*dx;
      x += dx;
   }
   printf("Значение интеграла: %6.3f", s);

   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 127

```
// Приближенное вычисление интеграла методом трапеций
#include <stdio.h>
#include <conio.h>
void main()
{
   float a,b;   // границы отрезка
   float dx;    // приращение аргумента
```

```
   float s;     // приближенное значение интеграла
   int n;       // количество интервалов
   float x;     // аргумент
   float y1,y2; // значение функции в начале и в конце
   int i;

   printf("\nПриближенное вычисление интеграла\n");
   printf("методом трапеций\n");
   printf("Нижняя граница отрезка -> ");
   scanf("%f", &a);
   printf("Верхняя граница отрезка -> ");
   scanf("%f", &b);
   printf("Приращение аргумента -> ");
   scanf("%f", &dx);
   n = (b - a) / dx;
   x = a;
   s = 0;
   for (i = 1; i <=n; i++)
   {
      y1 = x*x + 2;   // значение ф-и в начале интервала
      x += dx;
      y2 = x*x + 2;   // значение ф-и в конце интервала
      s += (y1 + y2)*dx/2;
   }
   printf("Значение интеграла: %6.3f", s);

   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 129

```
// Преобразование десятичного числа в двоичное
#include <stdio.h>
#include <conio.h>
void main()
{
   int dec;     // десятичное число
   int v;       // вес формируемого разряда
   int i;       // номер формируемого разряда
```

```
   printf("\nПреобразование десятичного числа в двоичное\n");
   printf("Введите целое число от 0 до 255");
   printf ("и нажмите <Enter>");
   printf("-> ");
   scanf("%i", &dec);
   printf("Десятичному числу %i соответствует двоичное ",
   dec);
   v = 128;  // вес старшего (восьмого) разряда
   for (i = 1; i <= 8; i++)
   {
      if (dec >= v)
      {
        printf("1");
        dec -= v;
      }
      else printf("0");
      v = v / 2; // вес следующего разряда в два раза меньше
   }
   printf("\n\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 130

```
// Программа проверяет знание таблицы умножения
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>  // для доступа к srand и rand
#include <time.h>

void main()
{
   int numb1,numb2; // сомножители
   int res;         // произведение
   int otv;         // ответ испытуемого
   int kol = 0;     // количество правильных ответов
   int i;           // счетчик циклов
   time_t t;        // текущее время - для инициализации
                    // генератора случайных чисел
```

```
   printf("\*** Проверка знания таблицы умножения ***\n");
   printf(" После примера введите ответ и нажмите <Enter>");

   srand((unsigned) time(&t));; // инициализация генератора
                                // случайных чисел

   for (i = 1; i <= 10; i++)  // 10 примеров
   {
      numb1 = rand()%7 + 2 ;  // число от 2 до 9
      numb2 = rand()%7 + 2 ;
      res = numb1 * numb2;
      printf("%ix%i=", numb1, numb2);
      scanf("%i",&otv);
      if (otv == res)
         kol++;
      else printf("Вы ошиблись! %ix%i=%i\nПродолжим...\n",
                  numb1, numb2, res);
   }
   printf("\nПравильных ответов: %i\n", kol);
   printf("Ваша оценка: ");
   switch (kol)
   {
       case 10: puts("5"); break;
       case 9:  puts("4"); break;
       case 8:  puts("4"); break;
       case 7:  puts("3"); break;
       default: puts("2"); break;
   }
   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 131

```
// Проверка умения складывать и вычитать числа
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>  // для доступа к srand и rand
#include <time.h>
```

```
#define LEVEL 97+2   // действия над числами от 2 до 99

void main()
{
   int numb1,numb2; // числа
   int op;          // действие над числами:
                    // 0 - сложение, 1 - вычитание
   char zop;        // знак операции - "плюс" или "минус"
   int res;         // результат
   int otv;         // ответ испытуемого
   int kol = 0;     // количество правильных ответов
   int buf;         // буфер для обмена numb1  и numb2,
                    // в случае, если numb1<numb2
   int i;           // счетчик циклов
   time_t t;        // текущее время - для инициализации
                    // генератора случайных чисел

   printf("\nПроверка умения складывать и вычитать числа\n");
   printf("После примера введите ответ и нажмите <Enter>");
   kol = 0;
   srand((unsigned) time(&t));; // инициализация генератора
                                // случайных чисел

   for (i = 1; i <= 10; i++)
   {
      // сгенерируем пример
      numb1 = rand() % LEVEL;  // число от 2 до 99
      numb2 = rand() % LEVEL;
      op = rand()%2;           // действие над числами
      if (op == 0)
      {
         res = numb1 + numb2;
         zop = '+';
      }
      else
      { // Вычитание
         zop = '-';
         if (numb1 < numb2)
         {
           // обменяем numb1 и numb2
```

```
            buf = numb2;
            numb2 = numb1;
            numb1 = buf;
         }
         res = numb1 - numb2;
      }
      printf("%i%c%i=", numb1, zop, numb2); // вывести пример
      scanf("%i", &otv);  // получить ответ испытуемого
      if (otv == res)
         kol++;
      else printf("Вы ошиблись. %i%c%i=%i\n",
                     numb1, zop, numb2, res);
   }
   printf("Правильных ответов: %i\n", kol);
   printf("Ваша оценка:\n");
   switch (kol)
   {
       case 10: puts("5"); break;
       case 9:  puts("4"); break;
       case 8:  puts("4"); break;
       case 7:  puts("3"); break;
       default: puts("2"); break;
   }
   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 132

```
// Электронные часы
#include <stdio.h>
#include <conio.h>
#include "dos.h"    // для доступа к delay
void main()
{
    int min,sec;    // минуты, секунды
    clrscr();       // очистить экран
    _setcursortype(_NOCURSOR);  // убрать курсор
    printf("Чтобы остановить таймер, нажмите любую клавишу");
    for (min = 0; min <= 2; min++)
   {
```

```
      for (sec = 0; sec <= 59; sec++)
      {
        delay(1000);     // задержка 1000 ms
        gotoxy(1,3);     // курсор в 1-ую колонку 1-ой строки
        printf("%i:%2i", min, sec);
        if (kbhit()) break;
      }
      if (kbhit()) break;
    }
    _setcursortype(_NORMALCURSOR);
    getch(); // клавиша, остановившая часы
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 133

```
// Вычисление среднего арифметического
// последовательности положительных чисел
#include <stdio.h>
#include <conio.h>
void main()
{
    int a;    // число, введенное с клавиатуры
    int n;    // количество чисел
    int s;    // сумма чисел
    float m;  // среднее арифметическое

    s = 0;
    n = 0;
    printf("\Вычисление среднего арифметического");
    printf("последовательности положительных чисел.\n");
    printf("Вводите числа. Для завершения введите ноль.\n");
    do {
       printf("-> ");
       scanf("%i", &a);
       if (a > 0)
       {
          s += a;
          n++;
       }
```

```
    } while (a > 0);
    printf("Введено чисел: %i\n", n);
    printf("Сумма чисел: %i\n", s);
    m  =  (float) s / n;
    printf("Среднее арифметическое: %3.2f", m);

    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 134

```
// Определение максимального числа
// в последовательности положительных чисел
#include <stdio.h>
#include <conio.h>
void main()
{
    int a; // очередное число
    int m; // максимальное число

    puts("\nОпределение максимального числа");
    puts("последовательности положительных чисел.");
    puts("Вводите числа. Для завершения введите ноль.");
    m = 0;
    do {
       printf("-> ");
       scanf("%i", &a);
       if  (a > m)  m  =  a;
    } while (a > 0);
    printf("Максимальное число: %i", m);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 135

```
// Определение минимального числа
// последовательности положительных чисел
#include <stdio.h>
#include <conio.h>
```

```
void main()
{
    int a;    // очередное число
    int min;  // минимальное число

    printf("\nОпределение минимального числа\n");
    printf("в последовательности положительных чисел.\n");
    printf("Вводите числа. Для завершения введите ноль.\n");
    printf("-> ");
    scanf("%i", &a);
    min = a;     // пусть первое число минимальное
    while ( a > 0)
    {
       if (a < min) min = a;
       printf("-> ");
       scanf("%i", &a);
    }
    printf("Минимальное число последовательности: ");
    printf("%i\n", min);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 136

```
// Проверяет, является ли число простым
#include <stdio.h>
#include <conio.h>
void main()
{
     int n;  // число
     int d;  // делитель
     int r;  // остаток от деления n на d

     printf("Введите целое число-> ");
     scanf("%i", &n);
     d = 2;        // сначала будем делить на два
     do {
        r = n % d;
```

```
        if (r != 0) d++;
     }
     while ( r != 0 ); // пока n не разделится на d
     if (d == n)
       printf("%i - простое число" ,n);
     else  printf("%i - не простое число" ,n);

    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 138

```
// Игра "Угадай число"
#include <conio.h>
#include <stdlib.h>  // для доступа к srand
#include <time.h>
void main()
{
   int comp;   // задуманное число
   int igrok;  // вариант игрока
   int n;      // количество попыток
   time_t t;   // текущее время - для инициализации
               // генератора случайных чисел


   srand((unsigned) time(&t));
   comp = rand() % 10 +1 ;     // число от 1 до 10

   clrscr();
   cprintf("\n\rКомпьютер \"задумал\" число от 1 до\
   10.\n\r");
   cprintf("Вы должны его угадать за три попытки.");
   n = 0;
   do {
    cprintf("\n\r->");
    cscanf("%i",&igrok);
    n++;
   }  while ((igrok != comp)&&(n < 3));

   if (igrok == comp)
   {
```

```
      textcolor(RED+BLINK);
      cprintf("\n\rВЫ ВЫИГРАЛИ!");
   }
   else
   {
      textcolor(GREEN);
      cprintf("\n\rВы проиграли.);
      cprintf("Компьютер задумал число %d",comp);
   }
   textcolor(LIGHTGRAY);
   cprintf("\n\rДля завершения нажмите любую клавишу...");
   getch();getch();
}
```

## Задача 140

```
// Выводит таблицу функции
#include <stdio.h>
#include <conio.h>
void main()
{
    float x,dx;      // аргумент и его приращение
    float x1,x2;     // диапазон изменения аргумента
    float y;         // значение функции

    x1 = -4;
    x2 = 4;
    dx = 0.5;
    x = x1;
    printf("--------------\n");
    printf("   x  |   y\n");
    printf("--------------\n");
    while (x < x2) {
       y = x*x + 2;
       printf("%3.2f  | %3.2f\n", x, y);
       x += dx;
    }
    printf("--------------\n");

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 141

```
// Вычисление числа "Пи"
#include <stdio.h>
#include <conio.h>
void main()
{
     float p;   // вычисляемое значение ПИ
     float t;   // точность вычисления
     int n;     // номер члена ряда
     float el;  // значение члена ряда

     p = 0;
     n = 1;
     el = 1; // начальное значение
     printf("\nЗадайте точность вычисления ПИ -> ");
     scanf("%f", &t);
     printf("Вычисление ПИ с точностью %f\n",t);
     while (el >= t )
     {
      el = (float) 1 / (2*n -1);
      if ((n % 2) == 0)
        p -= el;
      else p += el;
      n++;
     }
     p = p*4;
     printf("\nЗначение ПИ с точностью %f равно %f\n", t, p);
     printf("Просуммировано %i членов ряда.\n", n);

     printf("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 142

```
// Вычисление наибольшего общего делителя
// двух целых чисел (алгоритм Евклида)
#include <stdio.h>
#include <conio.h>
void main()
```

```
{
     int n1,n2;   // числа, НОД которых надо вычислить
     int nod;     // наибольший общий делитель
     int r;       // остаток от деления n1 на n2

     printf("\nВычисление наибольшего общего делителя ");
     printf("для двух целых чисел.\n");
     printf("Введите в одной строке два числа ");
     printf("и нажмите <Enter>");
     printf("-> ");
     scanf("%i%i", &n1, &n2);
     printf("НОД чисел %i и %i - это ", n1, n2);
     while (n1 % n2)
     {
       r = n1 % n2; // остаток от деления
       n1 = n2;
       n2 = r;
     }
     nod = n2;
     printf("%i\n", nod);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 143

```
// Подсчет ненулевых элементов массива
// (доступ к элементам по номеру)
#include <stdio.h>
#include <conio.h>

#define SIZE 5 // размер массива
void main()
{
    int a[SIZE];  //массив
    int n = 0;    // кол-во ненулевых эл-тов
    int i;        // индекс

    printf("\nВведите массив целых чисел.\n");
    printf("После ввода каждого числа ");
    printf("нажимайте <Enter>\n");
```

```
    for (i = 0; i < SIZE; i++)
    {
        printf("a[%i] ->",i+1);
        scanf("%i", &a[i]);
        if (a[i] != 0)  n++;
    }
    printf("В массиве %i ненулевых элемента.\n", n);
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 144

```
// Поиск минимального элемента массива
#include <stdio.h>
#include <conio.h>
#define HB  5    // размер массива
void main()
{
    int a[HB];  // массив
    int min;    // номер минимального элемента
    int i;      // индекс массива

    printf("\nПоиск минимального элемента массива\n");
    printf("Введите в одной строке элементы массива,\n");
    printf("%i целых чисел, и нажмите <Enter>\n", HB);
    printf("-> ");
    for (i = 0; i < HB; i++)
        scanf("%i",&a[i]);

    min = 0; // предположим, что первый эл-т минимальный
    // сравним оставшиеся эл-ты массива с минимальным
    for (i = 1; i < HB; i++)
        if (a[i] < a[min])  min  =  i;

    printf("Минимальный элемент массива: ");
    printf("a[%i]=%i ", min+1, a[min]);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 145

```
// Поиск минимального элемента массива
// (доступ к элементам при помощи указателя)
#include <stdio.h>
#include <conio.h>
#define HB  5    // размер массива
void main()
{
    int a[HB];   // массив
    int *min;    // номер минимального элемента
    int *p;      // указатель на элемент массива
    int i;

    printf("\nПоиск минимального элемента массива\n");
    printf("Введите в одной строке элементы массива,\n");
    printf("%i целых чисел, и нажмите <Enter>\n", HB);
    printf("-> ");
    p = a;
    for (i = 1; i <= HB; i++)
        scanf("%i", p++);

    min = a;    // пусть первый элемент минимальный
    p = a + 1;
    // теперь p содержит адрес второго элемента
    // сравним оставшиеся эл-ты массива с минимальным
    for (i = 2; i <= HB; i++)
    {
        if (*p < *min) min = p;
        p++;  // к следующему элементу
    }
    printf("Минимальный элемент массива: %i\n", *min);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 148

```
// Вычисление средней (за неделю) температуры воздуха
#include <stdio.h>
#include <conio.h>
```

```
void main()
{
    // названия дней недели - массив строковых констант
    char *day[] = {"Понедельник","Вторник","Среда",
           "Четверг","Пятница","Суббота","Воскресенье"};
    float t[7];    // температура
    float sum;     // сумма температур за неделю
    float sred;    // средняя температура за неделю
    int i;

    printf("\nВведите температуру воздуха:\n");
    for (i = 0; i <= 6; i++)
    {
       printf("%s->", day[i]);
       scanf("%f", &t[i]);
       sum += t[i];
    }
    sred = sum / 7;
    printf("\nСредняя температура за неделю: %2.1f", sred);

    printf("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 149

```
// Поиск в массиве методом перебора элементов
#include <stdio.h>
#include <conio.h>
#define HB 5
void main()
{
    int m[HB];  // массив целых
    int obr;    // образец для поиска
    int found;  // признак совпадения с образцом
    int i;

    printf("\nПоиск в массиве методом перебора\n");
    printf("Введите в одной строке %i целых\n", HB);
    printf("чисел и нажмите <Enter>\n");
    printf("->");
```

```
    for (i = 0; i < HB; i++)
        scanf("%i", &m[i]);
    printf("Введите образец для поиска (целое число)->");
    scanf("%i", &obr);

    // поиск простым перебором
    found = 0;
    i = 0;    // проверяем с первого элемента массива
    do {
        if (m[i] == obr )
            found = 1;   // совпадение с образцом
        else i++;        // переход к следующему элементу
    } while (!found && i < HB);
    if ( found )
         printf("Совпадение с элементом номер %i", i+1);
    else
         printf("Совпадений с образцом нет");

    printf("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 150

```
// Проверяет, отсортирован ли массив по возрастанию
#include <stdio.h>
#include <conio.h>
#define HB 5
void main()
{
    int a[HB];   // массив
    int k;       // индекс
    int ok;      // 1 - последовательность неубывающая

    printf("Проверка, упорядочен ли массив\n");
    printf("по возрастанию\n");
    printf("Введите массив (%i целых чисел ", HB);
    printf("в одной строке) и нажмите <Enter>\n");
    for (k = 0; k < HB; k++)
        scanf("%i", &a[k]);
```

```
    k = 0;
    ok = 1;
    do  {
        if (a[k] > a[k+1])
            ok = 0;
        k++;
    } while ( k < НВ-1 && ok);

    printf("Элементы массива ");
    if ( !ok )
        printf("не упорядочены ");
        printf("по возрастанию\n");

    printf("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 151

```
// Проверяет, сколько раз число встречается в массиве
#include <stdio.h>
#include <conio.h>
#define НВ 5 // размер массива
void main()
{
    int a[НВ]; // массив
    int obr;   // искомое число (образец)
    int n;     // кол-во элементов массива,
               // значение которых равно образцу
    int i;     // индекс

    printf("Введите массив (%i ", НВ);
    printf("целых чисел в одной строке)\n");
    printf("->");
    for (i = 0; i < НВ; i++)
        scanf("%i",&a[i]);
    printf("Введите образец для сравнения ->");
    scanf("%i", &obr);
    n = 0;
    for (i = 0; i < НВ; i++)
        if (a[i] == obr)  n++;
```

```
    if ( n )
       printf("Число %i встречается в массиве %i раз",
              obr, n);
    else printf("Ни один элемент массива не равен образцу");

    printf("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 153

```
// Сортировка массива методом прямого выбора
#include <stdio.h>
#include <conio.h>
#define SZ 5 // размер массива
void main()
{
    int a[SZ];  // массив of integer;
    int i;      // номер элемента, от которого ведется поиск
                // минимального эл-та
    int min;    // номер минимального элемента в части
                // массива от i до верхней границы массива
    int j;      // номер эл-та, сравниваемого с минимальным
    int buf;    // используется при обмене эл-тов массива
    int k;      // индекс для ввода и вывода

    printf("\nСортировка массива\n");
    printf("Введите массив (в одной строке %i", SZ);
    printf("целых чисел) и нажмите <Enter>\n");
    printf("->");
    for (k = 0; k < SZ; k++)
        scanf("%i", &a[k]);

    printf("Сортировка...\n");
    for (i = 0; i < SZ-1; i++)
    {
        // поиск минимального эл-та
        // в части массива от a[i] до последнего эл-та
        min = i;
        for (j = i+1; j < SZ; j++)
            if (a[j] < a[min])  min = j;
        // поменяем местами a[min] и a[i]
```

```
        buf = a[i];
        a[i] = a[min];
        a[min] = buf;
        // цикл сортировки закончен
        // отладочная печать
        // выведем промежуточное состояние массива
        for (k = 0; k < SZ; k++)
            printf("%i ", a[k]);
        printf("\n");
    }

    // выведем отсортированный массив
    printf("Массив отсортирован\n");
    for (k = 0; k < SZ; k++)
            printf("%i ", a[k]);
    printf("\n");

    printf("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 154

```
// Сортировка массива методом "пузырька"
#include <stdio.h>
#include <conio.h>
#define SZ 5
void main()
{
    int a[SZ];
    int i;     // счетчик циклов
    int k;     // текущий индекс элемента массива
    int buf;

    printf("\nСортировка массива методом \"пузырька\"\n");
    printf("Введите массив (в одной строке %i ", SZ);
    printf("целых чисел) и нажмите <Enter>\n");
    for (k = 0; k < SZ; k++)
        scanf("%i", &a[k]);
    printf("Сортировка...\n");
```

```
    for (i = 0; i < SZ-1; i++)
    {
        for (k = 0; k < SZ-1; k++)
        {
            if (a[k] > a[k+1])
            {
                // обменяем k-й и (k+1)-й элементы
                buf = a[k];
                a[k] = a[k+1];
                a[k+1] = buf;
             }
        }
        // отладочная печать - состояние
        // массива после очередного цикла сортировки
        for (k = 0; k < SZ; k++)
            printf("%i ",a[k]);
        printf("\n");
    }

    printf("Массив отсортирован\n");
    for (k = 0; k < SZ; k++)
            printf("%i ",a[k]);

    printf("\n\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 155

```
// Объединение двух упорядоченных массивов в один
#include <stdio.h>
#include <conio.h>
#define SZ 5  // размер исходных массивов
void main()
{
    int a[SZ], b[SZ];  // исходные массивы
    int c[SZ*2];       // массив - результат
    int k,i,m;         // индексы массивов a, b и c

    printf("Объединение двух упорядоченных ");
    printf("по возрастанию массивов\n");
    printf("Введите первый массив ");
```

```
    printf("(%i целых чисел) -> ", SZ);
    for (k = 0; k < SZ; k++)
        scanf("%i", &a[k]);

    printf("Введите второй массив ");
    printf("(%i целых чисел) -> ", SZ);
    for (i = 0; i < SZ; i++)
        scanf("%i", &b[i]);

    k = i = m = 0;
    do {
        if (a[k] < b[i] )
            c[m++] = a[k++];
        else
            if (a[k] > b[i])
                c[m++] = b[i++];
            else {
                c[m++] = a[k++];
                c[m++] = b[i++];
            }
    } while ( k < SZ && i < SZ); // один из двух исходных
    // массивов полностью не переписан в массив C

    while (k < SZ) // есть эл-ты A, не переписанные в C
        c[m++] = a[k++];

    while (i < SZ) // есть эл-ты B, не переписанные в C
        c[m++] = b[i++];

    printf("Массив - результат: \n");
    for (i = 0; i < 2 * SZ; i++)
        printf("%i ", c[i]);

    printf("Для завершения работы нажмите <Enter>\n");
    getch();
}
```

## Задача 156

```
// Бинарный поиск в упорядоченном массиве
#include <stdio.h>
#include <conio.h>
```

```
#define SZ 10     // размер массива
void main()
{
    int a[SZ]; // массив целых
    int obr;   // образец для поиска
    int ok;    // 1 - массив упорядочен

    int verh,niz;  // границы части массива, в которой
                   // выполняется поиск
    int sred;      // индекс среднего элемента в области
                   // поиска
    int found;     // 1 - поиск успешен
    int n;         // счетчик сравнений с образцом
    int i;

    // ввод массива
    printf("*** Бинарный поиск ");
    printf("в упорядоченном массиве ***\n");
    printf("Введите массив (в одной строке %i ", SZ);
    printf("целых чисел) и нажмите <Enter>\n");
    printf("-> ");
    for (i = 0; i < SZ; i++)
        scanf("%i", &a[i]);

    // проверим, упорядочен ли массив по возрастанию
    ok = 1; // пусть массив упорядочен
    i = 0;
    do
        if (a[i] <= a[i+1])
            i++;
        else ok = 0;
    while (ok && i < SZ -1);

    if ( !ok) {
        puts("Введенный массив не является");
        puts("упорядоченным по возрастанию\n");
        goto bye;
    }
```

```
    printf("Введите образец для поиска (целое число) -> ");
    scanf("%i", &obr);

    // бинарный поиск
    verh = 0;
    niz = SZ - 1;
    found = 0;
    n = 0;
    do {
        sred = (niz-verh) / 2 + verh; // делим массив пополам
        n++;
        if (a[sred] == obr)
           found = 1;
    else
        // в какой части, в верхней или в нижней,
        // может находиться искомый элемент?
        if ( obr < a[sred])
            niz = sred-1;      // в верхней
        else verh = sred+1;    // в нижней
    } while (verh <= niz && !found);
    if (found) {
        printf("Совпадение с элементом номер %i ", sred);
        printf("Выполнено %i сравнений" , n);
    }
    else
        printf("Образец в массиве не найден\n");
bye:
    printf("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 157

```
// Анализ роста учеников
#include <stdio.h>
#include <conio.h>
#define  SZ 30   //максимальное кол-во учеников
void main()
{
    int r;        // рост ученика
```

```
    int rost[SZ]; // рост всех учеников
    int n = 0;    // кол-во учеников, о которых
                  // введены сведения
    float sred;   // средний рост
    int m = 0;    // кол-во учеников, у которых
                  // рост больше среднего
    int sum = 0;  // суммарный рост
    int i = 0;

    printf("*** Анализ роста учеников ***\n");
    printf("Вводите рост (см) учеников\n");
    printf("Для завершения введите 0 и нажмите <Enter>\n");

    do {
        printf("-> ");
        scanf("%i", &r);
        if ( r )
        {
           rost[i++] = r;
           sum += r;
           n++;
         }
    } while (r && i < SZ);

    if ( n )
    {
        sred = (float) sum / n;
        m = 0;
        // сравним рост каждого со средним
        for (i = 0; i < n; i++)
            if (rost[i] > sred) m++;

        printf("Средний рост: %3.2f см\n", sred);
        printf("У %i учеников рост превышает средний\n", m);
    }

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 158

```
// Вычисление суммы элементов массива (по столбцам)
#include <stdio.h>
#include <conio.h>
#define  ROW 3    // кол-во строк
#define  COL 5    // кол-во столбцов

void main()
{
    int a[ROW][COL]; // массив
    int s[COL];      // сумма элементов
    int i,j;

    printf("\nВведите массив\n");
    printf("После ввода элементов каждой строки,");
    printf("\n%i целых чисел, нажимайте <Enter>\n", COL);
    for (i = 0; i < ROW; i++) // ROW строк
    {
        printf("->");
        for (j = 0; j < COL; j++)
            scanf("%i", &a[i][j]);
     }

    printf("\nВведенный массив\n");
    for (i = 0; i < ROW; i++)
    {
        for (j = 0; j < COL; j++)
            printf("%i " , a[i][j]);
        printf("\n");
    }

    // "очистим" массив s
    for (i = 0; i < COL; i++)
        s[i] = 0;

    // обработка
    for (j = 0; j < COL; j++)      // для каждого столбца
        for (i = 0; i < ROW; i++) // суммируем эл-ты
            s[j] += a[i][j];
```

```
    printf("-----------------------------\n");
    for (i = 0; i < COL; i++)
        printf("%i ", s[i]);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 160

```
// Обработка результатов экзамена
#include <stdio.h>
#include <conio.h>
void main()
{
    int n[6];   // количество двоек, ..., пятерок
    int s = 0;  // всего оценок
    float p[6]; // процент каждой оценки

    char *mes[6] ={"\0", "\0", "двоек\0","троек\0",
                   "четверок\0","пятерок\0"};
    int i;

    puts("Обработка результатов экзамена");
    puts("Введите исходные данные:");
    for (i = 5; i >= 2; i--)
    {
        printf("%s ->", mes[i]);
        scanf("%i", &n[i]);
        s += n[i];
    }
    // вычислим процент каждой оценки
    for (i = 2; i < 6; i++)
        p[i] = (float)n[i]/s*100;

    puts("Результаты экзамена");
    puts("---------------------------");
    for (i = 5; i >= 2; i--)
        printf("%8s  %2i    %2.0f%\n",mes[i], n[i], p[i]);
    puts("---------------------------");
```

```
    puts("Для завершения программы нажмите <Enter>");
    getch();
}
```

## Задача 162

```
// Определитель матрицы второго порядка
#include <stdio.h>
#include <conio.h>
void main()
{
    float a[2][2];   // матрица
    float det;       // определитель (детерминант)
    int i,j;         // индексы массива

    printf("\nВведите матрицу второго порядка.\n");
    printf("После ввода элементов строки нажмите <Enter>\n");
    for (i = 0; i < 2; i++)
    {
        printf("->");
        scanf("%f%f", &a[i][0], &a[i][1]);
     }
    det = a[0][0] * a[1][1] - a[0][1] * a[1][0];

    printf("Определитель матрицы\n");
    for (i = 0; i < 2; i++)
        printf("%f %f\n", a[i][0], a[i][1]);

    printf("равен %f", det);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 163

```
// Строка с максимальной суммой элементов
#include <stdio.h>
#include <conio.h>

#define N 3  // размер квадратной матрицы
```

```
void main()
{
    int m[N][N+1]; // последний столбец используем
                   // для хранения суммы эл-тов строки
    int max;       // строка с максимальной суммой
                   // элементов
    int i,j;       // индексы

    puts("\nОпределение строки с максимальной");
    puts("суммой элементов");
    printf("Введите матрицу %ix%i\n", N, N);
    for (i = 0; i < N; i++)
    {
        printf("Элементы %i-й строки -> ", i+1);
        for (j = 0; j < N; j++)
            scanf("%i", &m[i][j]);
    }

    // для каждой строки вычислим сумму эл-тов
    for (i = 0; i < N; i++)
    {
        m[i][N] = 0;
        for(j = 0; j < N; j++)
           m[i][N] += m[i][j];
    }

    // найдем строку с максимальной суммой
    max = 0;
    for (i = 1; i < N; i++)
        if ( m[i][N] > m[max][N] )
           max = i;

    printf("\nВ %i-й строке сумма элементов", max+1);
    printf("максимальна и равна %i\n", m[max][N]);

    printf("\nДля завершения нажмите <Enter>\n");

    getch();
}
```

## Задача 164

```
// Проверяет, является ли матрица "магическим" квадратом
#include <stdio.h>
#include <conio.h>
#define  SZ 5      // максимальный размер матрицы
void main()
{
    int a[SZ][SZ];  // матрица
    int n;          // размер проверяемой матрицы
    int ok;         // матрица - "магический" квадрат
    int i,j;        // индексы массива
    int sum;        // сумма эл-тов главной диагонали
    int temp;       // сумма элементов текущей строки,
                    // столбца или второй диагонали матрицы

    printf("*** МАГИЧЕСКИЙ КВАДРАТ ***\n");
    printf("\nВведите размер матрицы (3..%i) -> ", SZ);
    scanf("%i", &n);
    printf("Введите строки матрицы\n");
    printf("После ввода строки, %i целых чисел, ", n);
    printf("нажимайте <Enter>\n");
    for (i = 0; i < n; i++)
    {
        printf("->");
        for (j = 0; j < n; j++)
            scanf("%i", &a[i][j]);
    }

    ok = 1;  // пусть матрица - "магический" квадрат
    // вычислим сумму элементов главной диагонали
    sum = 0;
    for (i = 0; i < n; i++)
        sum += a[i][i];

    // вычисляем суммы по строкам
    i = 0;
    do {
        temp = 0; // сумма эл-тов текущей строки
        for (j = 0; j < n; j++)
            temp += a[i][j];
```

```
        if (temp != sum)  ok = 0;
        i++;
    } while (ok && i < n);

    if ( ok )
    {
        // здесь сумма элементов каждой строки
        // равна сумме элементов главной диагонали

        // вычисляем суммы по столбцам
        j = 0;
        do {
            temp = 0; // сумма эл-тов текущего столбца
            for (i = 0; i < n; i++)
                temp += a[i][j];
            if (temp != sum)  ok = 0;
            j++;
        } while (ok && i < n);
    }

    if ( ok ) {
         // здесь сумма элементов каждой строки
         // равна сумме элементов каждого столбца и
         // сумме элементов главной диагонали.
         // Вычислим сумму элементов второй
         // главной диагонали
         temp = 0;
         i = n - 1;
         for (j = 0; j < n; j++)
            temp += a[i--][j];
         if (temp != sum)  ok = 0;
    }
    printf("Введенная матрица ");
    if ( !ok )
        printf("не ");
    printf("является \"магическим\"квадратом.\n");

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 165

```
// Приветствие
#include <stdio.h>
#include <conio.h>
void main()
{
    char name[15]; // имя
    char fam[20];  // фамилия

    printf("Как Вас зовут?\n");
    printf("Введите свое имя и фамилию,");
    printf("затем нажмите <Enter>");
    printf("-> ");
    scanf("%s", &name);
    scanf("%s", &fam);
    // функция scanf читает из буфера клавиатуры символы
    // до разделителя - пробела
    printf("Здравствуйте, %s %s!\n", name, fam);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 166

```
// Приветствие (посимвольный ввод строки)
#include <stdio.h>
#include <conio.h>
void main()
{
    char name[40]; // имя и отчество пользователя
    char ch;
    int i;

    printf("Как Вас зовут?\n");
    printf("(введите свое имя, отчество и нажмите <Enter>");
    printf("-> ");
    i = 0;
    while ((ch=getch()) != 13 && i < 40) // пока не нажата
                                         // клавиша <Enter>
```

```
    {  putch(ch);
       name[i++] = ch;
    }
    name[i] = '\0';
    printf("\nЗдравствуйте, %s!\n", name);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 167

```
// Вычисляет длину строки
#include <stdio.h>
#include <conio.h>
void main()
{
   char st[80]; // введенная строка
   int i = 0;   // длина строки

   puts("\nВведите строку и нажмите <Enter>");
   printf("->");
   gets(st);
   while( st[i++])
          ;

   printf("Длина введенной строки: %i\n", i);
   printf("Для завершения работы нажмите <Enter>");
   getch();
}
```

## Задача 168

```
// Посимвольный вывод сообщения
#include <stdio.h>
#include <conio.h>
#include "dos.h" // для доступа к функции delay
void main()
{
    char msg[] = "\n\rПриветствую великого программиста!\0";
    int i;       // номер символа
```

```
    i = 0;
    while(msg[i])
    {
       putch(msg[i++]);
       delay(150);
    }

    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 169

```
// Выводит код символа
#include <stdio.h>
#include <conio.h>
void main()
{
    unsigned char ch;
    // Если ch объявить как char, то буквам русского
    // алфавита будут соответствовать отрицательные числа

    printf("\nВводите символы.\n");
    printf("Для завершения введите точку.\n");
    do {
      ch = getch();
      printf("Символ: %c Код: %i\n", ch, ch);
    } while ( ch != '.');

    printf("\n\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 170

```
// ASCII-таблица кодировки символов
#include <stdio.h>
#include <conio.h>

#define  SM 128  //  0  - символы с кодами 0 - 127
                 // 128 - символы с кодами 128 - 256
```

```
void main()
{
    // Если ch объявить как char, то буквам русского
    // алфавита будут соответствовать отрицательные числа
    unsigned char ch;     // символ

    int i,j;

    printf("\nASCII-таблица кодировки символов\n");
    for (i = 0; i <= 16; i++) // шестнадцать строк
    {    ch = i + SM;
       for (j = 1; j <= 8; j++) // восемь колонок
       {
          if (( ch <7 || ch >= 14) && ch !=26)
               printf("%3c -%4i", ch, ch);
          else // символы CR, LF, TAB не отображаются
               printf("%3c -     ", ch, ch);
          ch += 16;
       }
       printf("\n");
    }

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 171

```
// Преобразование прописных букв в строчные
#include <stdio.h>
#include <conio.h>
void main()
{
    unsigned char st[80];  // строка текста
    int i;                 // номер обрабатываемого символа

    printf("\nВведите строку текста и нажмите <Enter>");
    printf("->");
    gets(st);
    i = 0;
```

```
    while ( st[i] )
    {
      if ((st[i] >= 'a' && st[i] <= 'z') ||
      (st[i] >= 'a' && st[i] <= 'п'))
        st[i] -= 32;
      else if (st[i] >= 'p' && st[i] <= 'я')
         st[i] -= 80;
      i++;
    }
    printf("\n%s\n", st);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 172

```
// Удаление начальных пробелов из строки
#include <stdio.h>
#include <conio.h>
#include "string.h"
void main()
{
     unsigned char sst[80];  // строка
     unsigned char dst[80];  // буфер
     int i,j;

     printf("Удаление начальных пробелов\n");
     printf("Введите строку:");

     i=0;
     while ((sst[i] = getch()) != 13)
         putch(sst[i++]);
     sst[i] = '\0';

     i = 0; j = 0;
     // найдем первый символ, отличный от пробела
     while( sst[i] && sst[i] == ' ')
         i++;

     // здесь i - номер первого символа, отличного от пробела,
     // скопируем sst в dst
```

```
    while (sst[i])
        dst[j++] = sst[i++];
    dst[j] = '\0';

    printf("\nСтрока без начальных пробелов:%s\n",dst);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 173

```
// Проверяет, является ли строка целым числом
#include <stdio.h>
#include <conio.h>
void main()
{
    char st[40];   // строка
    int i;         // номер проверяемого символа

    printf("Введите целое число и нажмите <Enter>");
    printf("->");
    scanf("%s",&st);
    i = 0;
    while (st[i] >= '0' && st[i] <= '9')
       i++;

    // здесь st[i] '\0', если введены только цифры
    printf("Введенная строка ");
    if (st[i])
      printf("не ");
    printf("является целым числом.\n");

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 175

```
// Проверяет, является ли введенная строка
//  шестнадцатеричным числом
```

```
#include <stdio.h>
#include <conio.h>
#include "string.h"
void main()
{
    char st[20];  // строка
    int i;        // номер проверяемого символа

    printf("\nВведите шестнадцатеричное число ->");
    scanf("%s", &st);

    strupr(st); // преобразуем к верхнему регистру

    i = 0;
    while ((st[i] >= '0' && st[i] <= '9') ||
       (st[i] >= 'A' && st[i] <= 'F'))
      i++;

    printf("Строка ");
    // если st[i] != '\0',
    // то i - номер первого ошибочного символа
    if ( st[i] )
      printf("не ");
    printf("является шестнадцатеричным числом.\n");

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 176

```
// Проверяет, является ли строка
//  дробным числом без знака

#include <stdio.h>
#include <conio.h>
void main()
{
   char st[20];   // строка
   int i;         // номер проверяемого символа
   int ok = 0;    // пусть строка — не дробное число
```

```
   printf("Введите дробное число и нажмите <Enter>");
   printf("->");
   scanf("%s", &st);

   i = 0;
   if  (st[i] >= '1' && st[i] <='9') // первый символ — цифра
   {
       // за цифрой могут быть еще цифры
       while ( st[i] >= '1' && st[i] <='9' )
       i++;
       // за цифрами должна быть точка
       if (st[i] == '.')
       {
         i++;
         // за точкой должна быть хотя бы одна цифра
         if  (st[i] >='1' && st[i] <='9')
         {
          // и еще цифры
          while ( st[i] >= '1' && st[i] <='9' )
             i++;
          ok = 1; // похоже строка - дробное число
          }
       }
    }
    printf("Строка %s ",st);
    if ( st[i] || !ok )
       printf("не ");
    printf("является дробным числом без знака.\n");

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 177

```
// Преобразует двоичное число в десятичное
#include <stdio.h>
#include <conio.h>
#include "string.h"
void main()
{
```

```
    char bin[16];    // изображение двоичного числа
    long int dec;    // десятичное число
    int i;           // номер разряда двоичного числа
    int v;           // вес i-го разряда двоичного числа

    printf("Введите восьмиразрядное двоичное число ");
    printf("и нажмите <Enter>");
    printf("->");
    scanf("%s", &bin);

    dec = 0;
    v = 1;    // вес младшего (0-го) разряда двоичного числа
    for ( i = strlen(bin) -1; i >= 0; i--)
    {
      if ( bin[i] == '1' )
     dec += v;
      v *= 2;     // вес следующего разряда
    }
    printf("Двоичному числу %s", bin);
    printf("соответствует десятичное %d", dec);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 178

```
// Преобразует шестнадцатеричное число в десятичное
// разобраться с переполнением!
#include <stdio.h>
#include <conio.h>
#include "string.h"
void main()
{
    char st[5];      // шестнадцатеричное число
    unsigned int dec;// десятичное число
    int v;           // вес разряда шестнадцатеричного числа
    int err = 0;     // err == 1 - в строке недопустимый символ
    int i;
```

```
    printf("Введите шестнадцатеричное ");
    printf("(не более 4-х знаков) число\n");
    printf("-> ");
    scanf("%s",&st);

    // преобразуем введенную строку к верхнему регистру
    strupr(st);

    dec = 0;
    v = 1;    // вес младшего разряда шестнадцатеричного
              // числа
    for ( i = strlen(st) -1; i >= 0; i--)
    {
      //printf("\n%d\n",v);
      if (st[i] >= '0' && st[i] <= '9')
     dec += v * (st[i]- 48); // (int)'0'=48, (int)'1'=49
                             // и т.д.
       else
     if (st[i] >= 'A' && st[i] <= 'F')
        // (int)'A'=65, (int)'B'=66 и т.д.
        // A обозначает 10, B - 11 и т.д.
        dec += v * (st[i]- 55);
     else // недопустимый символ
       { err = 1;
         break; }
      v *= 16;     // вес следующего разряда
   }
   if ( !err ) {
       printf("Шестнадцатеричному числу %s ", st);
       printf("соответствует десятичное %u\n", dec);
   }
   else {
       printf("Строка %s не является ", st);
       printf("шестнадцатеричным числом\n");
   }

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 179

```
// Преобразует десятичное число в другую
// систему счисления (от 2-х до 10-ти)
#include <stdio.h>
#include <conio.h>
void main()
{
    int osn,     // основание системы счисления
        n,       // исходное число
        cn,      // копия исходного числа
        r;       // остаток от деления числа
                 // на основание сист. счисл.
    char st[17]; // представление числа в заданной
                 // системе счисления
    int i;

    printf("\Введите целое число ->");
    scanf("%d", &n);
    printf("Введите основание системы счисления ->");
    scanf("%d", &osn);

    cn = n;
    //  делим исходное число на основание системы
    //  счисления до тех пор, пока остаток от деления
    //  больше основания системы счисления.
    //  Остаток от деления на каждом шаге - очередная цифра
    st[16] ='\0';
    i = 15;
    do {
      r = n  % osn;  // очередная цифра
      n = n / osn; // целая часть деления
      // printf("цифра:%d остаток:%d\n", r,n);
      st[i--] = r + 48; // преобразование цифры в символ
    } while ( n > 0);

    // "сдвинем" сформированную строку в начало
    i++;
    int j = 0;
    while(st[i])
      st[j++] = st[i++];
    st[j] = '\0';
```

```
    st[i--] = ' ';
    printf("Десятичному числу %d соответствует ", cn);
    printf("число %s по основанию %d\n", st, osn);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 180

```
// Преобразует десятичное число в шестнадцатеричное
#include <stdio.h>
#include <conio.h>
void main()
{
    int n;          // Исходное число
    int r;          // Остаток от деления числа на основание
                    // системы счисления
    char st[5];     // Представление числа в заданной
                    // системе счисления
    int i;
    printf("\nПреобразование десятичного числа );
    printf("в шестнадцатеричное\n");
    printf("Введите целое число ->");
    scanf("%d", &n);

    // делим исходное число на 16 до тех пор,
    // пока остаток от деления больше 16

    printf("\nДесятичному числу %d", n);
    printf(" соотвествует шестнадцатеричное ");
    st[5] = '\0';
    i = 4;
    do {
       r = n % 16;  // очередная цифра
       n = n / 16;  // целая часть рез-та деления
       if (r < 10)
           st[i--] = r + 48; // (int)'0'==48, (int)'1'==49 и т. д.
       else st[i--] = r + 55; // (int)'A'==65, (int)'B'==66
                              // и т. д.
    } while ( n > 0);
```

```
    // удалим начальные пробелы
    i++;
    int j = 0;
    while( st[i] )
       st[j++] = st[i++];
    st[j] = '\0';

    printf("%s\n", st);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 181

```
// Вычисление значения арифметического выражения
#include <stdio.h>
#include <conio.h>
#include "stdlib.h"
void main()
{
   char st[40];  // строка
   char buf[10]; // изображение очередного числа
   char op;      // оператор
   int rez;      // значение выражения
   int n;        // очередное число

   int i,j;

   printf("\nВведите арифметическое выражение,\n");
   printf("например, 45+5-3-125+2 и нажмите <Enter>");
   printf("(пробелы и другие знаки недопустимы)\n");
   printf("->");
   scanf("%s", &st);

   rez = 0; // значение выражения
   op = ' ';
   i = j = 0;
   while( st[i] )
   {
      // выделить число
```

```
      j = 0;
      while (st[i] >= '0' && st[i] <= '9')
         buf[j++] = st[i++];
      buf[j] = '\0';
      n = atoi(buf); // преобразовать строку в целое

      // выполнить действие
      switch ( op )
      {
        case '+': rez += n; break;
        case '-': rez -= n; break;
        case ' ': rez = n;  break; // первое число примера
      }

      // выделить знак операции
      op = st[i++];
   }
   printf("Значение введенного выражения: %d", rez);
   printf("\nДля завершения нажмите <Enter>");
   getch();
}
```

## Задача 182

```
// Итоги летней Олимпиады 2000 года
#include "stdio.h"
#include "conio.h"
#include "string.h"
#define NC 10      //количество стран-участниц
void main()
{
   char *strana[] = {"Австрия\0","Германия\0",
               "Канада\0","Китай\0","Корея\0",
               "Норвегия\0","Россия\0",
               "США\0","Франция\0","Япония\0"};

   // таблица результатов
   int result[NC+1][5];
   // NC+1-ая строка используется как буфер
   // при сортировке таблицы
```

```
int i,j;
int max;     // номер строки таблицы, в которой
             // количество очков максимально
char buf[9]; // используется при сортировке

printf("\n*** Сидней 2000 ***\n");
printf("Введите в одной строке количество золотых,\n");
printf("серебряных и бронзовых медалей\n");

// ввод исходных данных
for (i = 0; i < NC; i++)
{
   printf("%s ->", strana[i]);
   scanf("%i%i%i", &result[i][0],   // золотых
                   &result[i][1],   // серебряных
                   &result[i][2]);  // бронзовых
}

// вычислим общее количество медалей и очков
for (i = 0; i < NC; i++)
{
     result[i][3] =
            result[i][0]+result[i][1]+result[i][2];
     result[i][4] =
            result[i][0]*7+result[i][1]*6+result[i][2]*5;
}
// сортировка массива в соответствии с количеством очков
// методом простого выбора
for (i = 0; i < NC-1; i++)
{
    // в части таблицы, начиная со строки i,
    // найти j-ую строку, в которой элемент
    // result[j][5] максимальный

    max = i; //  пусть это строка с номером i
    for (j = i+1; j < NC; j++)
          if (result[j][4] > result[max][4])  max = j;

    // обменяем i-ую строку со строкой с номером max
    // в качестве буфера используем последнюю
    // строку таблицы.
```

```
      strcpy(buf,strana[i]);
      strcpy(strana[i],strana[max]);
      strcpy(strana[max],buf);
      for (j = 0; j < 5; j++)
           result[NC][j] = result[i][j];
      for (j = 0; j < 5; j++)
           result[i][j] = result[max][j];
      for (j = 0; j < 5; j++)
           result[max][j] = result[NC][j];

    }

    // здесь таблица упорядочена
    printf("Итоги летней Олимпиады в Сиднее, 2000 г.\n");
    printf("%12s%8s%8s%8s%8s%8s","Страна","Золото",
                  "Серебро","Бронза","Всего","Очков");
    for (i = 0; i < NC; i++)
    {
        printf("\n%12s", strana[i]);
        for (j = 0; j < 5; j++)
            printf("%8i", result[i][j]);
    }

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 183

```
// Игра "Угадай число"
#include "stdio.h"
#include "conio.h"
#include "stdlib.h"
#include "time.h"

#define  N 3   // уровень сложности - количество цифр в числе
#define DEBUG  // режим отладки

void main()
{
   char igrok[N];  // комбинация игрока
   char comp[N];   // комбинация компьютера
```

```
int a[N];  // a[i] == 1, если i-ая цифра
           // компьютера совпала с одной из цифр игрока

int ugad;      // угадано чисел
int mesto;     // из них на своих местах

int i,j;       // индексы
time_t t;

printf("\nКомпьютер задумал трехзначное число.\n");
printf("Вы должны его отгадать.\n");
printf("После ввода очередного числа вам будет ");
printf("сообщено, сколько цифр угадано, и сколько");
printf("из них находится на своих местах.\n");
printf("После ввода числа нажимайте <Enter>\n");

srand((unsigned)time(&t) );
for (i = 0; i < N; i++)  // компьютер "задумывает" число
   comp[i] = rand() % 10 + 48; // 48 - код символа '0'
#ifdef DEBUG
printf("Компьютер задумал: ");
for (i = 0; i < N; i++)
    printf("%c", comp[i]);
printf("\n");
#endif
do {
   printf("\nВаш вариант-> ");
   scanf("%s", &igrok);     // массив вводим как строку

   for (i = 0; i < N; i++) a[i] = 0;

   // проверим, сколько цифр угадано
   ugad = 0;
   for (i = 0; i < N; i++)    // каждую цифру игрока
      for (j = 0; j < N; j++) // сравним с цифрами
                              // компьютера
      {
         if ((igrok[i] == comp[j]) && !a[j])
         {
           ugad++;
```

```
                a[j] = 1; // запретим сравнивать
                          // эту цифру компьютера с оставшимися,
                          // еще не проверенными, цифрами игрока
                break;
              }
           }
        // проверим, сколько на своих местах
        mesto = 0;
        for (i = 0; i < N; i++)
            if (igrok[i] == comp[i])  mesto++;
                printf("Угадано: %i. На своих местах: %i", \
                     ugad, mesto);
    } while  ((ugad < N) || (mesto < N));
    printf("\n*** ВЫ УГАДАЛИ ЧИСЛО! ***\n");

    printf("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 184

```
// Телеграф - передача сообщений при помощи азбуки Морзе
#include "stdio.h"
#include "conio.h"
#include "string.h"  // strlen
#include "dos.h"     // delay

  // параметры передачи
#define  TONE 100  // частота сигнала (гц)
#define   L1   50  // длительность (мс) "точки"
#define   L2  100  // длительность (мс) "тире"
#define   L3   50  // пауза (мс) между точками и тире одной
                   // буквы
#define   L4  100  // пауза (мс) между буквами
#define   L5  150  // пауза (мс) между словами

void main()
{
    // кодировка букв русского алфавита
    char *morse[] = {
        ".-  ","-...",".---","--. ", //А,Б,В,Г
```

```
    "-.. ","."   ","...-","--..", // Д,Е,Ж,З
    ".."  ","."---","-.- ",".-..", // И,Й,К,Л
    "--  ","-.  ","--- ",".--.", // М,Н,О,П
    ".-. ","... ","-   ","..- ", // Р,С,Т,У
    "..-.","....","-.-.","---.", // Ф,Х,Ц,Ч
    "----","--.-","-..-","-.--", // Ш,Щ,Ъ,Ы
    "-..-","..-.","..--",".-.-"  // Ь,Э,Ю,Я
    };

unsigned char mes[80];  // сообщение
char sim[4];   // символ в кодировке Морзе -
               // последовательность точек и тире
char znak;     // "передаваемый" знак - тире или точка
int i,j;       // номер символа и знака

puts("\n*** Телеграф ***");
puts("Введите сообщение, которое надо передать");
puts("(используйте только заглавные русские буквы)");
printf("->");
gets(mes);
for (i = 0; i < strlen(mes); i++)
{
    if (mes[i] >= 'А' && mes[i] <='Я')
    {
       // определим код очередной буквы (ф-я Ord) сообщения
       // и получим из таблицы кодировки соответствующий
       // элемент массива - последовательность точек и тире
       strcpy(sim,morse[mes[i]-128]);
       j = 0;
       do
           if (sim[j] == '-' || sim[j] == '.')
           {
               putch(sim[j++]);
               sound(1000);
               if (sim[j] == '.')
                   delay(L1);
               else delay(L2);
               nosound;
               delay(L3);
           }
```

```
      while ( sim[j] != ' ' && j <4 );
      delay(L4); // пауза между буквами
     }
     else
       if (mes[i] == ' ')  // пробел между словами
       {
          printf("  "); // пробел между словами сообщения
          delay(L5);
       }
  }
  puts("\nСообщение передано!");
  puts("Для завершения работы нажмите <Enter>");
  getch();
}
```

## Задача 185

```
#include <stdio.h>
#include <conio.h>
#include <math.h> // для доступа к M_PI

// объем цилиндра
float vcil(float h, float r)
{
    return(M_PI*r*r*h);
}

void main()
{
    float r,h; // высота и радиус основания цилиндра
    float v;   // объем цилиндра
    puts("Вычисление объема цилиндра");
    printf("Введите высоту и радиус основания ->");
    scanf("%f%f", &h, &r);
    v = vcil(h, r);
    printf("Объем цилиндра %3.2f\n", v);
    printf("Для завершения нажмите <Enter>");
    getch();
}
```

## Задача 186

```
// Функция max возвращает максимальное из двух чисел
int max(int a, int b)
{
    if (a > b)
         return(a);
    else
         return(b);
}
```

## Задача 187

```
// Функция compare возвращает результат сравнения чисел
// в виде символа отношения
#include "stdio.h"
#include "conio.h"

char compare(int a, int b)
{
    char res;
    if (a > b) res = '>';
       else if (a < b)  res = '<';
              else res = '=';
    return(res);
}

void main()
{
    int x1,x2;   // сравниваемые числа
    char res;    // результат сравнения

    puts("Введите два целых числа и нажмите <Enter>");
    printf("->");
    scanf("%i%i", &x1, &x2);
    res = compare(x1,x2);  // вызов функции программиста
    printf("%i %c %i\n", x1, res, x2);

    puts("\nДля завершения работы программы \
    нажмите <Enter>");
    getch();
}
```

## Задача 188

```
// Вычисляет сопротивление электрической цепи
float sopr( float r1, float r2, int t)
{
    // r1,r2 - величины сопротивлений
    // t - тип соединения:
    //             1 - последовательное;
    //             2 - параллельное.
    // если тип соединения указан неверно,
    // то функция возвращает -1
    float r;
    if ( t==1)  r =  r1 + r2;
        else if (t== 2)  r = r1*r2/(r1+r2);
            else r = -1;
    return(r);
}
```

## Задача 191

```
// Функция "факториал"
#include "stdio.h"
#include "conio.h"

unsigned int factor(int x)
{
    unsigned int f = 1;
    for (int i = 2; i <= x; i++)
        f *= i;
    return(f);
}

void main()
{
    unsigned int f;
    puts("\nТаблица факториалов");
    for (int n = 1; n <= 8; n++)
    {
        f = factor(n);
        printf("%2i  %u\n", n, f);
    }
```

```
    puts("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 192

```
// Функция вычисляет доход по вкладу
float dohod(float sum,     // сумма вклада
            float stavka,  // процентная ставка (годовых)
            int srok)      // срок вклада (дней)
{
    return(sum*(stavka/100/365)*srok); // 365 кол-во дней в
                                       // году
}
```

## Задача 193

```
// Функция проверяет, является ли символ гласной буквой
int glasn(char ch)
{
    static char gl[] ="АаЕеИиОоУуЫыЭэЮюЯя\0";
    int i = 0;

    while (gl[i] && gl[i] != ch)
        i++;
    if ( gl[i] )
       return(1);
    else return(0);
}
```

## Задача 195

```
// Функция upcase
#include "stdio.h"
#include "conio.h"

// функция преобразования строчных букв в прописные
char* upcase(char *st)
{
    int i = 0;
    while ( st[i] )
```

```
    {
        if (st[i] >= 'a' && st[i] <= 'z' ||    // латинские
                st[i] >= 'а' && st[i] <= 'п')  // русские
            st[i] -= 32;
        else if (st[i] >= 'р' && st[i] <= 'я')
                st[i] -= 80;
        i++;
    }
    return st;
}

//  пример использования функции upcase
void main()
{
    char st[80];

    puts("Введите строку текста и нажмите <Enter>");
    printf("->");
    gets(st);
    puts(upcase(st));

    puts("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 196

```
// Функция решения квадратного уравнения
#include "stdio.h"
#include "conio.h"
#include "math.h"

int kvadur(float a, float b, float c,  // коэф-ты уравнения
           float *x1, float *x2)       // корни уравнения
    // значение функции - количество корней
    // или -1, если неверные исходные данные
{
    float d; // дискриминант
```

```
    if (a == 0) return(-1);

    d = b*b-4*a*c;
    if (d < 0)
    return(0);    // уравнение не имеет решения

    *x1 = (-b+sqrt(d))/(2*a);
    *x2 = (-b-sqrt(d))/(2*a);

    if (*x1 != *x2) return(2);
        else return(1);
}

// проверка работоспособности функции
void main()
{
    float a,b,c;  // коэффициенты уравнения
    float x1,x2;  // корни уравнения
    int n;        // кол-во корней

    puts("\nРешение квадратного уравнения");
    puts("Введите в одной строке коэффициенты и нажмите\
    <Enter>");
    printf("->");
    scanf("%f%f%f", &a, &b, &c);
    switch (kvadur(a,b,c,&x1,&x2))
    {
        case -1: puts("Ошибка исходных данных.");
                 break;
        case 0:  puts("Уравнение не имеет решения.");
                 break;
        case 1:  printf("Корни одинаковые: x=%3.2f", x1);
                 break;
        case 2:  printf("x1=%3.2f x2=%3.2f", x1, x2);
    }

    puts("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 197

```
// Функция starline выводит строку из звездочек
#include "stdio.h"
#include "conio.h"

// выводит строку из звездочек
void starline(int len)
{
    for (int i = 0; i < len; i++)
        putch('*');
}

void main()
{
    starline(10);
    puts("\nДля завершения работы нажмите <Enter>");
    getch();
}
```

## Задача 200

```
// Функция frame вычерчивает рамку
#include "stdio.h"
#include "conio.h"

// вычерчивает рамку
void frame(int l, int t, int w, int h)
{
    // l,t - координаты верхнего левого угла,
    // w, h - ширина и высота рамки

    int x,y; // координаты выводимого символа
    int i;
    // символы, из которых составляется рамка
    char c1 = 218, // левый верхний угол
         c2 = 196, // горизонтальная линия
         c3 = 191, // правый верхний угол
         c4 = 179, // вертикальная линия
         c5 = 192, // левый нижний угол
         c6 = 217; // правый нижний угол
```

```
    gotoxy(l,t);
    putch(c1);
    for (i = 0; i < w-2; i++) // символы верхней границы
                              // рамки
          putch(c2);
    putch(c3);
    y = t+1;
    x = l+w-1;
    for (i = 0; i < h-1; i++) //символы левой и правой границ
    {
        gotoxy(l,y);
        putch(c4);
        gotoxy(x,y);
        putch(c4);
        y++;
    }
    gotoxy(l,y);
    putch(c5);
    for (i = 0; i < w-2; i++) // символы нижней границы
        putch(c2);
    putch(c6);
}

void main()
{
    clrscr();
    frame(5,5,30,10);
    puts("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 201

```
// Функция getint
#include "stdio.h"
#include "conio.h"
#include "stdlib.h"

// Функция getint предназначена для ввода целого
// положительного числа, состоящего из одной или двух цифр.
// Во время ввода, для редактирования, может использоваться
// клавиша <Backspace>.
```

```
// При нажатии <Enter> функция возвращает
// введенное число.

#define K_BACK  8   // код клавиши <Backspace>
#define K_ENTER 13  // код клавиши <Enter>
#define HB      4   // допустимое количество цифр
int getint()
{
  char ch;        // текущий символ
  char buf[HB];   // введенные цифры
  int n = 0;      // кол-во введенных цифр

  buf[0] = '\0';
  while ((ch = getch()) != K_ENTER)
        if (ch >= '0' && ch <= '9'&& n < HB)
        {    putch(ch);
            buf[n++] = ch;
        }
        else if (ch == K_BACK && n)
             {
                printf("\b \b");
                n--;
             }
  if (n)
  {
    buf[n] = '\0';
    return(atoi(buf));
  }
  else return(-1);
}

void main() {
  int a; // введенное число

  puts("\nДемонстрация работы функции getint\n");

  puts("Функция getint предназначена для ввода");
  puts("целого положительного числа.");
  puts("Во время ввода, для редактирования, может");
  puts("использоваться клавиша <Backspace>");
```

```
  puts("При нажатии <Enter> функция возвращает");
  puts("введенное число или -1, если число не введено.");

  puts("Введите число и нажмите <Enter>");
  printf("->");
  if (a = getint())
        printf("\nВы ввели число %d", a);
  else puts("Число не введено.");

  puts("\nДля завершения нажмите <Enter>");
  getch();
}
```

## Задача 202

```
// Функции getfloat и pos
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// возвращает позицию символа в строке
int pos(char* st, char c)
{
    int i = 0;
    while ( st[i] != c && st[i] )
        i++;
    if ( st[i] )
        return(i+1);
    else
        return(0);
}

// вводит дробное число
float getfloat()
{
    #define N 10 // кол-во символов, включая точку и минус
    char ch;
    char buf[N+1];
    int i;
```

```
    for (i = 0; i < N+1; i++)
        buf[i++] ='\0';
    i = 0;
    do {
        ch=getch();
        if (ch >= '0' && ch <= '9' && i < 8) {
            putch(ch);
            buf[i++] = ch;
        }
        else
             switch (ch) {
                case '-' : if (!i)
                           {
                              putch(ch);
                              buf[i++] = ch;
                           }
                           break;
                case '.' : if ( !(pos(buf,.')))
                           {
                              putch(ch);
                              buf[i++] = ch;
                           }
                           break;
                case 8   : if (i)
                            {
                                printf("\b \b");
                                buf[--i] = '\0';
                            }
             }
   } while (ch != 13);
   return(atof(buf));
}

void main(void)
{
   float f;

   printf("Введите дробное число ->");
   f = getfloat();
   printf("\nВведено число %e\n", f);
   getch();
}
```

## Задача 203

```
// Игра "21"
#include "stdio.h"
#include "conio.h"
#include "stdlib.h" // функция rand
#include "time.h"   // функция time


int koloda[12];     // колода карт
int karta();        // функция "выдает" карту из колоды

void main()
{
    int igrok = 0;    // очки игрока
    int comp = 0;     // очки компьютера
    char otv;         // ответ игрока
    time_t t;

    // создадим колоду
    for (int i=2; i <=11; i++)
        koloda[i] = 4;
    koloda[5] = 0; // "пятерок" в колоде нет

    // инициализация генератора случайных чисел
    srand((unsigned)time(&t));

    printf("\nИгра в карты до хорошего не доведет!\n");

    do {
        // карта игроку
        igrok += karta();

        // карта компьютеру
        if (igrok < 21)
            comp += karta();

        if (igrok < 21 && comp < 21)
        {
            printf("У вас %d\n",igrok);
            printf("Еще карточку? (введите y или n) ");
            otv = getchar();
```

```
            // Игрок нажимает две клавиши: с буквой и <Enter>
            // предыдущий вызов getchar читает букву.
            // При этом в буфере клавиатуры остается код
            // клавиши <Enter>. Прочитаем его.
            int b;
            b = getchar();
        }

    } while (igrok <= 21 && comp <= 21 && otv != 'n') ;

    if (igrok == 21 || (igrok < 21 && igrok > comp)
                    || comp > 21)
        printf("Вы выиграли!\n");
    else
        printf("Вы проиграли!\n");

    printf ("У вас %d\n", igrok);
    printf ("У компьютера %d\n", comp);

    printf("Для завершения нажмите <Enter>");
    getch();
}

// выдает карту из колоды
int karta()
{
    int i;
    do
        i = rand() % 10 +2;
    while (koloda[i] == 0);
    koloda[i]--;
    return i;
}
```

## Задача 205

```
// Рисует олимпийский флаг
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
```

```
#define PATHTODRIVER "c:\\borlandc\\bgi\\"

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      exit(1);
   }

    // полотнище флага - сплошная заливка серым цветом
    setfillstyle(SOLID_FILL,LIGHTGRAY); //
    bar(80,80,200,135);

    // кольца
    setcolor(GREEN);    // зеленое
    circle(100,100,15);
    setcolor(BLACK);    // черное
    circle(140,100,15);
    setcolor(RED);      // красное
    circle(180,100,15);
    setcolor(YELLOW);   // желтое
    circle(120,115,15);
    setcolor(BLUE);     // синее
    circle(160,115,15);

    getch();
    closegraph(); // выход из графического режима
}
```

## Задача 208

```
// Выводит корабль (с использованием метода базовой точки)
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>

#define dx  10  // шаг сетки по X
#define dy  10  // шаг сетки по Y

void ship(int x, int y) // x, y - координаты базовой точки
{
    //  корпус
    moveto(x,y);
    lineto(x,y-2*dy);
    lineto(x+10*dx,y-2*dy);
    lineto(x+11*dx,y-3*dy);
    lineto(x+17*dx,y-3*dy);
    lineto(x+14*dx,y);
    lineto(x,y);
    // надстройка
    moveto(x+3*dx,y-2*dy);
    lineto(x+4*dx,y-3*dy);
    lineto(x+4*dx,y-4*dy);
    lineto(x+13*dx,y-4*dy);
    lineto(x+13*dx,y-3*dy);
    line(x+5*dx,y-3*dy,x+9*dx,y-3*dy);
    // капитанский мостик
    rectangle(x+8*dx,y-4*dy,x+11*dx,y-5*dy);
    // труба
    rectangle(x+7*dx,y-4*dy,x+8*dx,y-7*dy);
    // иллюминаторы
    circle(x+12*dx,y-2*dy,dx/2);
    circle(x+14*dx,y-2*dy,dx/2);
    // мачта
    line(x+10*dx,y-5*dy,x+10*dx,y-10*dy);
    // оснастка
    moveto(x+17*dx,y-3*dy);
    lineto(x+10*dx,y-10*dy);
    lineto(x,y-2*dy);
}
```

```
#define PATHTODRIVER "c:\\borlandc\\bgi\\"

void main(void)
{
    int gdriver = DETECT; // драйвер
    int gmode;            // режим
    int errorcode;        // код ошибки

    initgraph(&gdriver, &gmode, PATHTODRIVER);
    errorcode = graphresult();

    if (errorcode != grOk) // ошибка инициализации
                           // граф. режима
    {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      exit(1);
    }

    ship(50,150);
    getch();
    closegraph(); // выход из графического режима
}
```

## Задача 209

```
// Узор из окружностей
// случайного радиуса и цвета
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include "time.h"
#include "dos.h"

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// узор из окружностей
void cuzor(int n)
{
```

```
    #define DELAY  // задержка между выводом окружностей
    int x,y,r;     // координаты центра и радиус окружности
    time_t t;

    srand((unsigned)time(&t)); // инициализация ГСЧ
    for (int i = 0; i < n; i++)
    {
        x = rand() % 640;
        y = rand() % 480;
        r = rand() % 240;
        setcolor(rand() %16);
        circle(x,y,r);
        #ifdef DELAY
        delay(5);
        #endif
    }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      exit(1);
   }

   cuzor(200); // узор из окружностей

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 211

```
// Узор из линий
// случайного цвета
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include "time.h"

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// узор из линий
void luzor(int n)
{
    int x,y;   // координаты конца линии
    int c;     // цвет линии
    time_t t;

    srand((unsigned)time(&t)); // инициализация ГСЧ
    for (int i = 0; i < n; i++)
    {

        x = rand() % 640;
        y = rand() % 480;
        c = rand() % 16;
        setcolor(c);
        lineto(x,y);
     }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
```

```
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      exit(1);
   }

   luzor(200); // узор из окружностей

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 212

```
// Контур пятиконечной звезды
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// контур пятиконечной звезды
#include "math.h"
void starline(int x0, int y0, int r)
{
    // x0, y0 - координаты центра звезды
    // r - радиус звезды

    int x,y;// координаты конца луча
            // или впадины
    int a;  // угол между осью OX и прямой,
            // соединяющей центр звезды и
            // конец луча или точку впадины
    int r1; // радиус окружности расположения
            // точек впадин

#define RTOR 2.5 // отношение радиуса лучей
                 // к радиусу впадин
```

```
    a = 18;    // строим от правого гор. луча
    r1 = r/RTOR;
    x = x0+r*cos(a*2*M_PI/360);
    y = y0-r*sin(a*2*M_PI/360);
    moveto(x,y);
    for (int i = 0; i < 5; i++)
    {
        a = a+36;
        x = x0+r1*cos(a*2*M_PI/360);
        y = y0-r1*sin(a*2*M_PI/360);
        lineto(x,y); // от луча к впадине
        a = a+36;
        if (a > 360)  a = 18;
        x = x0+r*cos(a*2*M_PI/360);
        y = y0-r*sin(a*2*M_PI/360);
        lineto(x,y); // от впадины к лучу
    }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      exit(1);
   }

   starline(100, 100, 50);

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 213

```
// Пятиконечная звезда
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// пятиконечная звезда
#include "math.h"
void star(int x0, int y0, int r)
{
    // x0, y0 - координаты центра звезды
    // r - радиус звезды

    int poly[20]; // координаты концов лучей
                  // и впадин
    int a;  // угол между осью OX и прямой,
            // соединяющей центр звезды и
            // конец луча или точку впадины
    int r1; // радиус окружности расположения
            // точек впадин

#define RTOR 2.5 // отношение радиуса лучей
                 // к радиусу впадин

    int i;
    a = 18;    // строим от правого гор. луча
    r1 = r/RTOR;
    i=0;
    do {
        poly[i++] = x0+r*cos(a*2*M_PI/360);
        poly[i++] = y0-r*sin(a*2*M_PI/360);
        a = a+36;
        poly[i++] = x0+r1*cos(a*2*M_PI/360);
        poly[i++] = y0-r1*sin(a*2*M_PI/360);
        a = a+36;
        if (a > 360)  a = 18;
    } while(i < 20);
    setfillstyle(SOLID_FILL,RED);
```

```
    fillpoly(10,poly);
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
   }
   else {
      star(100, 100, 20);
      getch();
      closegraph(); // выход из графического режима
   }
}
```

## Задача 215

```
// Российский флаг
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"
void rusflag(int x, int y, int l, int h)
{
     // x, y - координаты левого верхнего угла
     // l, h - длина и высота флага
     int w = h / 3;

     // рисуем флаг
     setfillstyle(SOLID_FILL,WHITE);
```

```
     bar(x,y,x+l,y+w);
     setfillstyle(SOLID_FILL,BLUE);
     bar(x,y+w,x+l,y+2*w);
     setfillstyle(SOLID_FILL,RED);
     bar(x,y+2*w,x+l,y+3*w);
     outtextxy(x,y+h+5,"Россия\0");
}
void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

    rusflag(100,100,50,25);

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 216

```
// Веселая рожица
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// веселая рожица
```

```
void face(int x, int y)
{
  setfillstyle(SOLID_FILL,YELLOW);
  setcolor(YELLOW);          // чтобы на круге не было линии
  pieslice(x,y,0,360,20);
  setcolor(BLACK);
  arc(x,y+2,180,360,10);  // рот
                          // глаза
  circle(x-7,y-7,2);
  circle(x+7,y-7,2);
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   face(100,100);

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 219

```
// Узор из разноцветных концентрических окружностей
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
```

```
#define PATHTODRIVER "c:\\borlandc\\bgi\\"

void main(void)
{

   int x = 100,  // координаты центра окружности
       y = 100;
   int r = 5;    // радиус наименьшей окружности
   int dr = 5;   // приращение радиуса окружности
   int color;    // цвет окружности

   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
   {
      for (color = 1; color <= 15; color ++)
      {
          setcolor(color);
          circle(x,y,r);
          r += dr;
      }
      getch();
      closegraph(); // выход из графического режима
   }
   else
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения нажмите <Enter>");
      getch();
   }
}
```

## Задача 220

```
// Узор из окружностей
#include <graphics.h>
```

```
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"
// узор из окружностей
void uzor()
{
    int x = 100,    // координаты центра окружности
        y = 100;
    int r = 20;     // радиус окружности
    int d = 30;     // расстояние между центрами окружностей
    int i,j;        // счетчики циклов

    for (i = 0; i < 4; i++)
    {
        x = 100;
        for (j = 0; j < 5; j++)
        {
            circle(x, y, r);
            x += d;
        }
        y += d;
    }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
```

```
      return;
   }

   uzor();

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 221

```
// Узор из квадратов
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"
// узор из квадратов
void uzor()
{
    int x;
    int y = 100;
    int n;        // количество квадратов в ряду
    int d = 30;   // размер квадрата
    int l = 10;   // расстояние между квадратами
    for (int i = 0; i < 5; i++)
    {
        // для ряда определим координату X
        if (i % 2)
        {  // нечетный ряд
            n = 5;   // пять квадратов в ряду
            x = 100;
        }
        else { // четный ряд
            n = 4;
            x = 100 + d/2+l/2;
        }
        for (int j = 0; j < n; j++)
        {
            rectangle(x,y,x+d,y+d);
            x += d+l;
        }
```

```
        y += d/2+l/2;
    }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   uzor();

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 222

```
// Шахматная доска
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"
// шахматная доска
void doska()
{
    int x0 = 100,   // координаты левого верхнего угла доски
        y0 = 100;
```

```
    int x,y;         // координаты левого верхнего угла клетки
    int w = 25;      // размер клетки
    int i,j;         // номер строки и колонки

    x = x0;
    y = y0;
    for (i = 0; i < 8; i++)   // восемь строк
    {
        for ( j = 0; j < 8; j++) // восемь клеток в строке
       {
           // если сумма номера строки и номера
           // колонки, на пересечении которых находится
           // клетка, четная, то клетка - коричневая,
           //    иначе - желтая
           if ((i+j) % 2)
                setfillstyle(SOLID_FILL,BROWN);
               else setfillstyle(SOLID_FILL,YELLOW);
           bar(x,y,x+w,y+w);
           x += w;
       }
       x = x0;
       y += w;
    }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
```

```
      return;
   }

   doska();

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 223

```
// Флажок
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"
int flag()
{
   int point[12]; // координаты точек флажка

   // задать координаты контура флажка
   point[0] = 100; point[1] = 100;
   point[2] = 160; point[3] = 100;
   point[4] = 140; point[5] = 120;
   point[6] = 160; point[7] = 140;
   point[8] = 100; point[9] = 140;
   point[10]= 100; point[11] = 100;
   setfillstyle(SOLID_FILL, RED);
   fillpoly(6, point);
   line(100, 140, 100, 170);
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();
```

```
   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   flag();

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 224

```
// Вычерчивает паровоз
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// вычерчивает паровоз
void parovoz(int x0, int y0)
{
   #define dx 5 // шаг координатной сетки
   #define dy 5 // шаг координатной сетки

   int tr[30]; // координаты точек контура паровоза

   // корпус
   tr[0]  = x0+0*dx;   tr[1]  = y0+7*dy;
   tr[2]  = x0+0*dx;   tr[3]  = y0+6*dy;
   tr[4]  = x0+1*dx;   tr[5]  = y0+6*dy;
   tr[6]  = x0+1*dx;   tr[7]  = y0+3*dy;
   tr[8]  = x0+2*dx;   tr[9]  = y0+3*dy;
   tr[10] = x0+2*dx;   tr[11] = y0+0*dy;
   tr[12] = x0+3*dx;   tr[13] = y0+0*dy;
   tr[14] = x0+3*dx;   tr[15] = y0+3*dy;
```

```
   tr[16] = x0+7*dx;   tr[17] = y0+3*dy;
   tr[18] = x0+7*dx;   tr[19] = y0+1*dy;
   tr[20] = x0+13*dx;  tr[21] = y0+1*dy;
   tr[22] = x0+13*dx;  tr[23] = y0+2*dy;
   tr[24] = x0+12*dx;  tr[25] = y0+2*dy;
   tr[26] = x0+12*dx;  tr[27] = y0+7*dy;
   tr[28] = x0+0*dx;   tr[29] = y0+7*dy;
   drawpoly(15,tr);

   // окно
   rectangle(x0+8*dx,y0+2*dy,x0+10*dx,y0+4*dy);
   // колеса
   setfillstyle(SOLID_FILL, RED);
   setcolor(RED);
   pieslice(x0+3*dx,y0+7*dy,0,360,1*dx);
   pieslice(x0+6*dx,y0+7*dy,0,360,1*dx);
   pieslice(x0+9*dx,y0+7*dy,0,360,1*dx);
   // окантовка колес
   setcolor(WHITE);
   circle(x0+3*dx,y0+7*dy,1*dx);
   circle(x0+6*dx,y0+7*dy,1*dx);
   circle(x0+9*dx,y0+7*dy,1*dx);
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
   {
         parovoz(100,100);
         getch();
         closegraph(); // выход из графического режима
   }
   else {
      printf("Ошибка: %d\n", errorcode);
```

```
      puts("Для завершения нажмите <Enter>");
      getch();
   }
}
```

## Задача 226

```
// Оцифрованные координатные оси
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

void grid()
{
    int x0,y0; // координаты начала координатных осей
    int dx,dy; // шаг координатной сетки (в пикселах)
    int h,w;   // высота и ширина области вывода
               // координатной сетки
    int x,y;

    float lx,ly;    // метки линий сетки по X и Y
    float dlx,dly;  // шаг меток линий сетки по X и Y
    char st[8];     // изображение метки линии сетки

    x0 = 50; y0 = 400; // оси начинаются в точке (50,400)
    dx = 40; dy = 40;  // шаг координатной сетки 40 пикселов
    dlx = 0.5;         // шаг меток оси X
                       // метками будут: 0.5, 1.0, 1.5 ...
    dly = 1;           // шаг меток оси Y
                       // метками будут: 1, 2, 3 ...
    h = 300;
    w = 400;

    lx = 0;            // в начало координат ставятся метки 0
    ly = 0;

    line(x0,y0,x0,y0-h);  // ось X
    line(x0,y0,x0+w,y0);  // ось Y
    // засечки, сетка и оцифровка по оси X
```

```
    x = x0;
    do {
        // засечка
        setlinestyle(SOLID_LINE, 0, 1);
        line(x,y0-3,x,y0+3);
        // оцифровка
        sprintf(st,"%2.1f",lx);
        outtextxy(x-8,y0+5,st);
        lx += dlx;
        // линия сетки
        setlinestyle(DOTTED_LINE, 0, 1);
        line(x,y0-3,x,y0-h);
        x += dx;
    } while (x < x0+w);

    // засечки, сетка и оцифровка по оси Y
    y = y0;
    do {
        // засечка
        setlinestyle(SOLID_LINE, 0, 1);
        line(x0-3,y,x0+3,y);
        // оцифровка
        sprintf(st,"%2.1f",ly);
        outtextxy(x0-40,y,st);
        ly += dly;
        // линия сетки
        setlinestyle(DOTTED_LINE, 0, 1);
        line(x0+3,y,x0+w,y);
        setlinestyle(SOLID_LINE, 0, 1);
        y -= dy;
    } while (y > y0-h);
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();
```

```
   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   grid();

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 227

```
// График функции
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

void grafik()
{
    float x,dx;   // аргумент и его приращение
    float x1,x2;  // диапазон изменения аргумента
    float y;      // значение функции
    int mx,my;    // масштаб по X и Y - кол-во точек
                  // экрана, соответствующее единице
                  // по осям координат
    int x0,y0;    // начало осей координат
    int px,py;    // координаты точки графика на экране

    x0 = 320; y0 = 240;
    mx = 20; my = 20;
    // оси координат
    line(10,y0,630,y0);
```

```
    line(x0,10,x0,470);
    // график
    x1 = -15;
    x2 = 5;
    dx = 0.1;
    x = x1;
    while ( x < x2 )
    {
        y =  0.5*x*x + x*4 - 3;  // функция
        px = x0 + x*mx;
        py = y0 - y*my;
        putpixel(px,py,WHITE);
        x += dx;
    }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode != grOk)  // ошибка инициализации
                           // графического режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   grafik();

   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 228

```
// Движущаяся окружность
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>     // для доступа к delay

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// вычерчивает окружность заданного цвета
void okr(int x, int y, // координаты центра
         int r,        // радиус
         int color)    // цвет
{
    setcolor(color);
    circle(x,y,r);
}

void main(void)
{

   int x,y;   // координаты центра окружности
   int r = 5; // радиус наименьшей окружности

   #define dt 10   // задержка между перемещениями
                   // 0.01 сек
   #define dx 5    // шаг перемещения

   int maxx;       // X — координата крайней правой
                   // точки экрана

   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
   {
```

```
        x = 0;   // движение от левой границы экрана
        y = 100;
        maxx = getmaxx();
        while (x < maxx)
        {
           okr(x,y,r,RED);  // нарисовать окружность
           delay(dt);       // задержка
           okr(x,y,r,BLACK);// стереть окружность
           x += dx;
        }
       closegraph(); // выход из графического режима
   }
   else
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения нажмите <Enter>");
      getch();
   }
}
```

## Задача 229

```
// Плывущий корабль
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>

#define dx  5  // шаг сетки по X
#define dy  5  // шаг сетки по Y

void ship(int x, int y, int color ) // x, y — координаты
                                    // базовой точки
{
    setcolor(color);
    //  корпус
    moveto(x,y);
    lineto(x,y-2*dy);
    lineto(x+10*dx,y-2*dy);
    lineto(x+11*dx,y-3*dy);
    lineto(x+17*dx,y-3*dy);
```

```
    lineto(x+14*dx,y);
    lineto(x,y);
    // надстройка
    moveto(x+3*dx,y-2*dy);
    lineto(x+4*dx,y-3*dy);
    lineto(x+4*dx,y-4*dy);
    lineto(x+13*dx,y-4*dy);
    lineto(x+13*dx,y-3*dy);
    line(x+5*dx,y-3*dy,x+9*dx,y-3*dy);
    // капитанский мостик
    rectangle(x+8*dx,y-4*dy,x+11*dx,y-5*dy);
    // труба
    rectangle(x+7*dx,y-4*dy,x+8*dx,y-7*dy);
    // иллюминаторы
    circle(x+12*dx,y-2*dy,dx/2);
    circle(x+14*dx,y-2*dy,dx/2);
    // мачта
    line(x+10*dx,y-5*dy,x+10*dx,y-10*dy);
    // оснастка
    moveto(x+17*dx,y-3*dy);
    lineto(x+10*dx,y-10*dy);
    lineto(x,y-2*dy);
}

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

void main(void)
{
    int x,y;  // координаты корабля (базовой точки)
    int maxx; // коорд. крайней правой точки экрана


    int gdriver = DETECT; // драйвер
    int gmode;            // режим
    int errorcode;        // код ошибки

    initgraph(&gdriver, &gmode, PATHTODRIVER);
    errorcode = graphresult();

    if (errorcode != grOk)  // ошибка инициализации
                            // графического режима
```

```
    {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
    }

    maxx = getmaxx();
    x = -10 ; // корабль выплывает из-за правой
              // границы экрана
    y = 100;
    while ( x <  maxx)
    {
        ship(x,y, GREEN);  // нарисовать корабль
        delay(20);
        ship(x,y,BLACK);   // стереть корабль
        x += 5;
    }
    setcolor(GREEN);
    outtextxy(10,10,"Рейс завершен!");
    outtextxy(10,24,"Нажмите <Enter>");
    getch();
    closegraph(); // выход из графического режима
}
```

## Задача 230

```
// Столбиковая диаграмма
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

    // глобальные переменные
    char *mes[] ={"двоек\0","троек\0",
                  "четверок\0","пятерок\0"};

    int n[4];   // количество пятерок, четверок,
                // троек и двоек
```

```
    float p[4]; // процент каждой оценки
    int h[4];   // высота столбиков диаграмм

void obr() // ввод и обработка
{
    int s;      // всего оценок
    int m;      // номер максимального эл-та массива n
    int i;      // индекс массива

    puts("Обработка результатов контрольной работы");
    puts("Введите исходные данные:");
    for (i = 3; i >= 0; i--)
    {
        printf("%s ->", mes[i]);
        scanf("%i", &n[i]);
    }
    // обработка
    s = 0;

    // всего оценок
    for (i = 0; i < 4; i++)
        s += n[i];

    // процент каждой оценки
    for (i = 0; i < 4; i++)
        p[i] = (float)n[i]/s*100;

    // вычислим высоту каждого столбика диаграммы,
    // но сначала определим, каких оценок больше
    m = 3; // пусть больше всего пятерок
    for (i = 2; i >= 0; i--)
        if (n[i] > n[m])  m = i;

    // Пусть количеству оценок, которых больше,
    // соответствует столбик высотой 200 пикселов.
    // Вычислим высоту остальных столбиков.
    for (i = 0; i < 4; i++)
        h[i] = 200 * n[i]/n[m];
}
```

```
void diagr()
{
    int x,y; // координаты левого нижнего угла
             // столбика диаграммы
    int i;   // индекс массива

    // цвет столбиков
    int color[4] = {YELLOW, BLUE, GREEN, RED};
    char buf[10];

    outtextxy(40,50,"Результаты контрольной работы\0");
    rectangle(40,80,170,310);
    x = 50; y = 300; // левый нижний угол первого столбика
    // столбики диаграммы
    for (i = 3; i >= 0; i--)
    {
        setfillstyle(SOLID_FILL, color[i]);
        bar(x,y,x+10,y-h[i]); // столбик
        sprintf(buf,"%2.1f",p[i]);
        outtextxy(x,y-h[i]-10,buf);
        x += 20;
    }
    // численные значения
    x = 50;
    for (i = 3; i >= 0; i--)
    {
      setfillstyle(SOLID_FILL,color[i]);
      //bar(x,y,x+10,y-h[i]); // столбик
      //OutTextXY(x,y-h[i]-10,RealToStr(p[i],5,1)+'%\n");
      x = x+20;
    }
    // легенда
    x = 200;y = 100;
    for (i = 3; i >= 0; i--)
    {
      setfillstyle(SOLID_FILL,color[i]);
      bar(x,y,x+20,y+10); // столбик
      outtextxy(x+25,y,mes[i]);
      y += 20;
    }
}
```

```
void main()
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   obr(); // ввод и обработка результатов

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
     diagr();  // вывод диаграммы
   else
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
   }
   getch();
}
```

## Задача 231

```
// Круговая диаграмма
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

    #define N 4 // количество категорий

void krdiagr(char* *name, float* dol)
{
    int a1,a2;     // угол начала и конца сектора
    int color[4] = {BLUE, YELLOW, GREEN, RED};
    int x,y;       // координаты вывода легенды
    char st[25];   // изображение числа
    int i;
```

```
    // строим диаграмму
    a1 = 0;             // от оси OX
    x = 350; y = 100;  // левый верхний угол области легенды
    for (i = 0; i < N; i++)
    {
      // сектор
      a2 = a1 + 3.6 * dol[i];  // 1% - 3.6 градуса
      if (i == N-1) a2 = 360; // последний, по счету, сектор
      setfillstyle(SOLID_FILL, color[i]);
      sector(200,200,a1,a2,100,100);
      // pieslice(200,200,a1,a2,100);
      a1 = a2;  // следующий сектор - от конца текущего
      // легенда
      bar(x,y,x+30,y+10);
      rectangle(x,y,x+30,y+10);
      sprintf(st, "%s - %2.1f%\0", name[i], dol[i]);
      outtextxy(x+50,y,st);
      y += 20;
    }
}

void main(void)
{
    char *name[N] = {"Книги\0","Журналы\0",
                     "Канцтовары\0","Прочее\0"};
    float kol[N];  // количество для категории
    float dol[N];  // доля категории в общем количестве
    float sum = 0; // общее кол-во по всем категориям
    int i;

    int gdriver = DETECT; // драйвер
    int gmode;            // режим
    int errorcode;        // код ошибки

    // ввод исходных данных
    puts("Введите количество по каждой категории");
    for (i = 0; i < N; i++)
    {
      printf("%s -> ", name[i]);
      scanf("%f", &kol[i]);
```

```
      sum += kol[i];
    }

    // вычислим долю каждой категории в общей сумме
    for (i = 0; i < N; i++)
      dol[i] = kol[i]/sum*100;

    // инициализация графического режима
    initgraph(&gdriver, &gmode, PATHTODRIVER);
    errorcode = graphresult();

    if (errorcode == grOk)
    {
        krdiagr(name, dol);// строим диаграмму
        getch();
        closegraph(); // выход из графического режима
   }
   else {
        printf("Ошибка: %d\n", errorcode);
        puts("Для завершения нажмите <Enter>");
        getch();
   }
}
```

## Задача 232

```
// Светофор
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// рисует круг заданного цвета
void krug(int x,int y, int r, int fc, int bc)
{
    // x, y, r - координаты центра и радиус
    // fc, bc - цвет круга и окантовки
    setfillstyle(SOLID_FILL,fc);
    setcolor(fc);
```

```
    pieslice(x,y,0,360,r);
    setcolor(bc);
    circle(x,y,r);
}

void main(void)
{
    int gdriver = DETECT; // драйвер
    int gmode;            // режим
    int errorcode;        // код ошибки

    initgraph(&gdriver, &gmode, PATHTODRIVER);
    errorcode = graphresult();

    if (errorcode != grOk)  // ошибка инициализации
                            // графического режима
    {
        printf("Ошибка: %d\n", errorcode);
        puts("Для завершения программы нажмите <Enter>");
        getch();
    }

    outtextxy(10,10,"Соблюдайте правила уличного движения!");
    rectangle(88,88,112,152);
    // Горит красный свет
    krug(100,100,10,RED,WHITE);
    krug(100,120,10,LIGHTGRAY,WHITE);
    krug(100,140,10,LIGHTGRAY,WHITE);
    for (int i = 1; i <= 3; i++)
    {
       // Здесь горит красный
       delay(3000);  // задержка 3 сек
       krug(100,120,10,YELLOW,WHITE); // Включить желтый
       delay(1000);
       krug(100,100,10,LIGHTGRAY,WHITE); // Выключить красный
       krug(100,120,10,LIGHTGRAY,WHITE); // Выключить желтый
       krug(100,140,10,GREEN,WHITE);     // Включить зеленый
       delay(2000);
       // Мигающий зеленый сигнал
       for (int j = 1; j <= 5; j++)  // мигает 5 раз
```

```
      {
         delay(500);
         krug(100,140,10,GREEN,WHITE);     // Включить
                                           // зеленый
         delay(500);
         krug(100,140,10,LIGHTGRAY,WHITE);// Выключить
                                           // зеленый
      }
      krug(100,120,10,YELLOW,WHITE);       // Включить
                                           // желтый
      delay(1500);
      krug(100,120,10,LIGHTGRAY,WHITE);    // Выключить
                                           // желтый
      krug(100,100,10,RED,WHITE);          // Включить
                                           // красный
   }
   outtextxy(10,25,"Нажмите <Enter>!");
   getch();
   closegraph(); // выход из графического режима
}
```

## Задача 233

```
// Часы с минутной и секундной стрелками
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <dos.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// выводит вектор заданной длины из текущей точки
// используется для вывода изображения стрелки
void vector(int a,   // угол между вектором и осью OX
            int l)   // длина вектора
{
   #define G 0.0174532 // коэф. перевода из градусов в радианы
   int x0,y0; // координаты начала вектора
   int x1,y1; // координаты конца вектора
```

```
    x0 = getx();
    y0 = gety();
    x1 = x0 + l*cos(a*G);
    y1 = y0 - l*sin(a*G);
    lineto(x1,y1);
}

void clock()
{
    int x0 = 80,   // координаты центра часов
        y0 = 80;
    int d = 50;    // диаметр циферблата
    int s = 0;     // время, кол-во секунд
    int m = 0;     // время, кол-во минут
    int as = 90;   // угол наклона секундной стрелки
    int am = 90;   // угол наклона минутной стрелки

    circle(x0,y0,d+5);
    setfillstyle(SOLID_FILL, 0);
    do {
       // вывести секундную стрелку
       moveto(x0,y0);
       setcolor(YELLOW);
       vector(as,d);

       // вывести минутную стрелку
       moveto(x0,y0);
       setcolor(GREEN);
       vector(am,d-10);

       delay(1000); // задержка

       // стереть стрелки
       setcolor(0);
       // секундную
       moveto(x0,y0);
       vector(as,d);

       // минутную
       moveto(x0,y0);
       vector(am,d-10);
       s++;
```

```
      if (s > 60) {
        m++;
        s = 0;
        am -= 6; // шаг движения минутной стрелки 6 градусов
        if (am < 0)  am = 354;
      }
      as -= 6;
      if (as < 0)  as = 354;

    } while ( !kbhit() );
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
   {
      clock();
      closegraph(); // выход из графического режима
   }
   else
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
   }
}
```

## Задача 234

```
// График функции
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
```

```
#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// функции, график которых надо построить
float f1(float x)
{
    return(2 * sin(x) * exp(x/5));
}

void grafik()
{
    float x1=0,   // границы изменения аргумента функции
          x2=25;
    float y1,y2;  // границы изменения значения функции
    float x;      // аргумент функции
    float y;      // значение функции в точке x
    float dx=0.01;   // приращение аргумента
    int l=50,     // левый нижний угол области графика
        b=400;
    int w=300,    // ширина и высота области графика
        h=200;
    float mx,my;  // масштаб по осям X и Y
    int x0,y0;    // точка - начало координат
    char st[25];  // изображение числа

    // найдем максимальное и минимальное значение
    // функций на отрезке [x1,x2]
    y1 = f1(x1); // минимум
    y2 = f1(x1); // максимум
    x = x1 + dx;
    do {
        y = f1(x);
        if (y < y1)  y1 = y;
        if (y > y2)  y2 = y;
        x += dx;
    } while (x <= x2);

    // вычислим масштаб по осям
    my = h/fabs(y2-y1);
    mx = w/fabs(x2-x1);
    // оси
```

```
    x0 = l;
    y0 = b-abs(y1*my);
    line(l,b,l,b-h);
    line(x0,y0,x0+w,y0);
    // максимальное и минимальное значения функции
    sprintf(st,"%3.2f",y2);
    outtextxy(l+5,b-h,st);
    sprintf(st,"%3.2f",y1);
    outtextxy(l+5,b,st);
    // построение графика
    x = x1;
    do {
        y = f1(x);
        putpixel(x0+x*mx,y0-y*my,15);
        x += dx;
    } while (x <= x2);
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
   {
        grafik();
        getchar();
        closegraph();
   }
   else {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
   }
}
```

## Задача 235

```
// Создает на диске файл
#include "stdio.h"
#include "conio.h"

#define FNAME "a:\\numbers.txt\0" // имя файла
#define N 5                       // количество чисел

// Создает на диске A: файл и записывает в него
// целые числа, введенные пользователем
void main()
{
     char fname[20] = FNAME;
     FILE *out;  // файл чисел
     int n;      // число

     puts("\nСоздание файла");
     printf("Введенные числа будут записаны в файл");
     printf("%s\n", fname);
     puts("После ввода каждого числа нажимайте <Enter>\n");

     // Открыть файл в режиме записи (w) текста (t)
     // Если файл с таким именем уже есть, то новые
     // данные будут записаны поверх старых
     // Для дозаписи в конец файла используйте
     // режим добавления (a)
     if ((out = fopen(fname, "wt")) == NULL)
     {
        printf("Ошибка открытия файла для записи");
        getch();
        return;
     }

     for (int i = 0; i < N; i++)
     {
        printf("->");
        scanf("%i", &n);
        fprintf(out,"%i",n);
     }
     fclose(out);     // закрыть файл
```

```
     printf("Введенные числа записаны в файл %s\n", fname);
     puts("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 236

```
// Добавляет данные в файл
#include "stdio.h"
#include "conio.h"

#define FNAME "a:\\numbers.txt\0" // имя файла
#define N 5                       // количество чисел

// Дописывает в находящийся на диске A: файл
// целые числа, введенные пользователем
void main()
{
     char fname[20] = FNAME;
     FILE *out;  // файл чисел
     int n;      // число

     puts("\nДобавление в файл");
     printf("Введенные числа будут добавлены в файл");
     printf("%s\n", fname);
     puts("После ввода каждого числа нажимайте <Enter>\n");

     // Открыть файл в режиме добавления (a) текста (t)
     // Если файла с таким именем нет, то он будет создан
     if ((out = fopen(fname, "at")) == NULL)
     {
        printf("Ошибка открытия файла для добавления");
        getch();
        return;
     }

     for (int i = 0; i < N; i++)
     {
        printf("->");
        scanf("%i", &n);
        fprintf(out,"%i\n",n);
    }
```

```
    fclose(out);     // закрыть файл
    printf("Введенные числа добавлены в файл %s\n", fname);
    puts("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 237

```
// Выводит на экран содержимое файла
#include "stdio.h"
#include "conio.h"

#define FNAME "a:\\numbers.txt\0" // имя файла

void main()
{
     char fname[20] = FNAME;
     FILE *in;     // текстовый файл
     char st[80];  // строка из файла

     printf("\nСодержимое файла %s\n", fname);
     puts("--------------------------------");

     // Открыть файл в режиме чтения (r) текста (t)
     if ((in = fopen(fname, "rt")) == NULL)
     {
        printf("Ошибка открытия файла для чтения");
        getch();
        return;
     }

     while (!feof(in))
     {
        fscanf(in,"%s", &st);
        printf("%s\n", st);
     }
     fclose(in);     // закрыть файл
     puts("--------------------------------");
     puts("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 238

```
// Вычисляет среднее арифметическое чисел,
// находящихся в файлах
#include "stdio.h"
#include "conio.h"

#define FNAME "a:\\numbers.txt\0" // имя файла

void main()
{
     char fname[20] = FNAME;
     FILE *in;     // текстовый файл

     int a;       // число
     int n = 0;   // количество чисел
     int sum = 0; // сумма чисел
     float sr;    // среднее арифметическое

     puts("\nВычисление среднего арифметического");
     printf("чисел, находящихся в файле %s", fname);

     // Открыть файл в режиме чтения (r) текста (t)
     if ((in = fopen(fname, "rt")) == NULL)
     {
        printf("Ошибка открытия файла для чтения");
        getch();
        return;
     }

     while (!feof(in))
     {
        fscanf(in,"%i", &a);
        sum += a;
        n++;
     }
     fclose(in);     // закрыть файл
     sr = (float) sum / n;
     printf("Введено чисел: %i\n", n);
     printf("Сумма чисел: %i\n", sum);
     printf("Среднее арифметическое: %3.2f", sr);
```

```
    puts("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 239

```
// Выводит на экран содержимое файла,
// имя которого указано пользователем
#include "stdio.h"
#include "conio.h"
#include "string.h"

#define  MAXLEN 80    // максимальная длина строки в файле
void main()
{
    char fname[40];   // имя файла
    FILE *in;         // текстовый файл

    char st[MAXLEN+2];// строка, прочитанная из файла
    int n = 0;        // кол-во строк, выведенных на экран
    char key;         // клавиша, нажатая пользователем

    puts("Просмотр текстового файла");
    puts("Введите полное имя файла и нажмите <Enter>");
    printf("->");
    scanf("%s",&fname);

    // Открыть файл в режиме чтения (r) текста (t)
    if ((in = fopen(fname, "rt")) == NULL)
    {
        printf("Ошибка при обращении к файлу %s\n", fname);
        getch();
        return;
    }

    clrscr();
    while (!feof(in))
    {
        fgets(st, MAXLEN, in);
        printf("%s", st);
        if (++n > 21)
```

```
        {
          printf("\nДля продолжения нажмите");
          printf("любую клавишу...");
          key = getch();
          gotoxy(1,wherey()); // курсор в начало текущей
                              // строки
          delline;            // удалить сообщение "Для
                              // продолжения ..."
            n = 0;
        }
    }
    fclose(in);     // закрыть файл

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 240

```
// Дописывает в файл фамилию, имя и номер телефона
#include "stdio.h"
#include "conio.h"

#define FNAME "a:\\phone.txt\0" // имя файла
void main()
{
    char fname[20] = FNAME;
    FILE *out;  // файл чисел

    char fam[15];  // фамилия
    char name[15]; // имя
    char tel[9];   // номер телефона

    puts("\nДобавление в телефонный справочник");

    // Открыть файл в режиме добавления (a) текста (t)
    // Если файла с таким именем нет, то он будет создан
    if ((out = fopen(fname, "at")) == NULL)
    {
        printf("Ошибка открытия файла для добавления");
        getch();
```

```
        return;
    }

    // получим данные от пользователя
    printf("Фамилия ->");
    scanf("%s", &fam);
    printf("Имя ->");
    scanf("%s", &name);
    printf("Телефон ->");
    scanf("%s", &tel);
    // и запишем их в файл
    fprintf(out,"%s %s %s", fam, name, tel);
    puts("Информация добавлена");
    fclose(out);     // закрыть файл

    printf("\n\nДля завершения нажмите <Enter>\n");
    getch();
}
```

## Задача 242

```
// Поиск в телефонном справочнике
#include "stdio.h"
#include "conio.h"

#define FNAME "a:\\phone.txt\0" // имя файла
void main()
{
    char fname[20] = FNAME;
    FILE *in;  // файл - телефонный справочник

    char obr[15];  // фамилия - образец для поиска в БД

    // найденная информация
    char fam[15];  // фамилия
    char name[15]; // имя
    char tel[9];   // номер телефона

    int n = 0; // количество записей, удовлетворяющих запросу

    puts("\nПоиск в телефонном справочнике");
```

```
    // Открыть файл в режиме чтения (r) текста (t)
    if ((in = fopen(fname, "rt")) == NULL)
    {
        printf("Ошибка открытия файла %s", fname);
        getch();
        return;
    }

    // получим данные от пользователя
    printf("Фамилия ->");
    scanf("%s", &obr);   // образец для поиска в БД
    while (!feof(in))
    {
        fscanf(in,"%s %s %s", &fam, &name, &tel);
        if (fam == obr)
        {
            printf("%s %s %s",fam, name, tel);
            n++;
        }
    }
    if (n )
        printf("Найдено записей: %i", n);
    else
        printf("Данных об абоненте %s в БД нет", obr);

    fclose(in);     // закрыть файл

    puts("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 244

```
// Универсальная программа тестирования
// имя файла теста задается в инструкции запуска программы
#include "stdio.h"
#include "conio.h"
#include "string.h"

void main(int argc, char* argv[])
{
```

```
char fname[40]; // имя файла теста
FILE* f;        // файл теста

int VsegoVopr = 0; // количество вопросов теста
int PravOtv = 0;   // количество правильных ответов

// для текущего вопроса
int nOtv;     // количество альтернативных ответов
int Prav;     // номер правильного ответа
int Otv;      // номер ответа, выбранного пользователем

int p;        // процент правильных ответов

char st[80];  // строка файла теста

int i; // счетчик циклов

if ( !argc )
{
    puts("\nНе задан файл вопросов теста!");
    puts("Командная строка: test ИмяФайлаТеста\n");
    return;
}

strcpy(fname,argv[1]); // имя файла из командной строки
// Открыть файл в режиме чтения (r) текста (t)
if ((f = fopen(fname, "rt")) == NULL)
{
    printf("Ошибка открытия файла %s", fname);
    getch();
    return;
}

clrscr();
puts("\nСейчас Вам будет предложен тест.");
puts("К каждому вопросу дается несколько \
вариантов ответа.");
puts("Вы должны ввести номер правильного ответа");
puts("и нажать клавишу <Enter>\n");

printf
```

```
    ("Для начала тестирования нажмите <Enter>");
    getch();
    textbackground(BLUE);
    clrscr();

    while (!feof(f))
    {
      VsegoVopr++;
      fgets(st, 80, f);          // читаем из файла вопрос
      printf("\n%s\n", st);     // вопрос на экран

      fscanf(f,"%i %i", &nOtv, &Prav);// кол-во вариантов
                                      // ответа
                                      // и номер прав. ответа
      fgets(st,80,f); // дочитать конец предыдущей строки

      //читаем и выводим альтернативные ответы
      for (i = 1; i <= nOtv; i++)
      {
            fgets(st, 80, f);
            printf("%i. %s", i, st);
      }
      printf("\nВаш выбор ->");
      scanf("%i", &Otv);
      if (Otv == Prav)  PravOtv++;
    }

    // обработка результата тестирования
    // вычислим процент правильных ответов
    p = 100 * PravOtv / VsegoVopr;
    printf("\nВаша оценка - ");
    if (p == 100) puts("ОТЛИЧНО!");
    if (p >= 99 && p <= 80) puts("ХОРОШО.");
    if (p >= 60 && p <= 79) puts("УДОВЛЕТВОРИТЕЛЬНО.");
    if (p < 60) puts("ПЛОХО!\n");

    puts("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 245

```
// Выводит имена всех файлов программ
// предполагается, что первая строка
// файла - комментарий, название программы
#include <stdio.h>
#include <dir.h>
#include <string.h>
#include <conio.h>

// #define DEBUG  // режим отладки

// в качестве параметра программе передается
// имя каталога, список файлов которого надо вывести
void main(int argc, char *argv[])
{
   struct ffblk ffblk;  // информация о файле
   int done;
   FILE *in;   // файл программы

   int n;      // обработано файлов

   char mask[MAXPATH];
   char infile[MAXPATH];
   char outfile[MAXPATH];

   if (argc < 2)
   {
    puts("В командной строке не задан путь");
    puts("к обрабатываемым файлам");
    printf("Командная строка: %s path\\\n", argv[0]);
    return;
   }

   printf("\nПостроение списка файлов\n");

   // маска обрабатываемых файлов
   strcpy(mask, argv[1]);
   strcat(mask,"*.cpp");

   // файл-список обработанных файлов
```

```
   strcpy(outfile, argv[1]);
   strcat(outfile,"filelist.txt");

   printf("Обработка: %s",  mask);
   n = 0;

   done = findfirst(mask, &ffblk,0);
   while (!done)
   {  n++;
      #ifdef DEBUG
      printf("%s ", ffblk.ff_name);
      #endif
      strcpy(infile, argv[1]);
      strcat(infile,ffblk.ff_name);
      if ((in = fopen(infile, "rt")) != NULL)
      {
        // читаем из файла первую строку
        char st[80];
        fgets(st, 80, in);
        printf("%s %s", infile, st);
        fclose(in);
      }
      done = findnext(&ffblk); // выбрать следующий файл
   }
   printf("\nОбработано файлов: %d\n", n);
   printf("Для завершения нажмите <Enter>");
   getch();
}
```

## Задача 247

```
// Рекурсивная функция "Факториал"
#include "stdio.h"
#include "conio.h"

unsigned int factor(unsigned int k)
{
    if ( k == 1 )
        return(1);
    else
        return(k*factor(k-1));
}
```

```
void main()
{
    unsigned int n; // число, факториал которого надо
                    // вычислить
    unsigned int f; // факториал числа n

    puts("Вычисление факториала\n");
    puts("Введите число, факториал которого надо вычислить");
    printf("->");
    scanf("%u", &n);
    f = factor(n);
    printf("Факториал числа %u равен %u", n, f);

    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

## Задача 248

```
// Рекурсивный узор из окружностей
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// элемент узора
void elem(int x, int y, int r, int p)
{
     // x, y, r — координаты и радиус центра
     // основного элемента узора
     // p — порядок узора

     if (p)
     {
        circle(x, y, r);
        delay(100);
        elem(x+r, y,   r/2, p-1);
        elem(x,   y-r, r/2, p-1);
        elem(x-r, y,   r/2, p-1);
```

```
        elem(x,   y+r, r/2, p-1);
    }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
   {
        elem(320, 240, 60, 5); // рисуем узор 5-го порядка
        outtext("Для завершения нажмите <Enter>");
        getch();
        closegraph(); // выход из графического режима
   }
   else
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения нажмите <Enter>");
      getch();
   }
}
```

## Задача 249

```
// Вычисляет сопротивление
// n-звенной электрической цепи
#include <stdio.h>
#include <conio.h>

float r1,r2,r3; // величины сопротивлений,
                // из которых состоит цепь

// вычисляет сопротивление цепи n-го порядка
float rcep(int n)
{
   float r; // сопротивление цепи порядка n-1
```

```
   if (n == 1)
      return(r1 + r2 + r3);
   else
   {
      r = rcep(n-1);
      return (r1 + r2*r/(r2+r) + r3);
   }
}

void main()
{
     int n;          // количество звеньев (порядок) цепи
     float rc;       // сопротивление цепи

     puts("\nВычисление сопротивления электрической цепи");
     puts("Введите величины сопротивлений (Ом):");
     printf("r1 ->");
     scanf("%f", &r1);
     printf("r2 ->");
     scanf("%f", &r2);
     printf("r3 ->");
     scanf("%f", &r3);
     printf("Порядок цепи ->");
     scanf("%i", &n);

     rc = rcep(n); // величины сопротивлений передаются
                   // функции rcep через глобальные
                   // переменные

     printf("Сопротивление цепи:");
     if (rc > 100)
     {
         rc /= 1000;
         printf("%5.2f кОм\n", rc);
     }
     else
         printf("%5.2f Ом\n", rc);

     puts("\nДля завершения нажмите <Enter>");
     getch();
}
```

## Задача 250

```
// Вычерчивает схему электрической цепи
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

#define PATHTODRIVER "c:\\borlandc\\bgi\\"

// вычерчивает схему эл. цепи от точки
// с координатами x,y
void drcep(int k, int x, int y)
{
      #define dx 7 // шаг сетки по X
      #define dy 7 // шаг сетки по Y

      setcolor(GREEN);
      line(x,y,x+2*dx,y);
      rectangle(x+2*dx,y-dy,x+6*dx,y+dy);
      line(x+6*dx,y,x+8*dx,y);
      outtextxy(x+3*dx,y-3*dy,"R1");

      setcolor(YELLOW);
      line(x+8*dx,y,x+8*dx,y+2*dy);
      rectangle(x+7*dx,y+2*dy,x+9*dx,y+6*dy);
      line(x+8*dx,y+6*dy,x+8*dx,y+8*dy);
      outtextxy(x+10*dx,y+2*dy,"R2");

      setcolor(LIGHTGRAY);
      line(x,y+8*dy,x+2*dx,y+8*dy);
      rectangle(x+2*dx,y+7*dy,x+6*dx,y+9*dy);
      line(x+6*dx,y+8*dy,x+8*dx,y+8*dy);
      outtextxy(x+3*dx,y+5*dy,"R3");

      if ( k > 1 )  drcep(k-1, x+8*dx, y);
}

void main(void)
{
   int k; // порядок цепи

   int gdriver = DETECT; // драйвер
```

```
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
   {
      printf("Введите порядок цепи -> ");
      scanf("%i", &k);
      drcep(k, 10, 50);

      outtextxy(10,200,
                "Для завершения нажмите <Enter>");
      getch();
      closegraph(); // выход из графического режима
   }
   else
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения нажмите <Enter>");
      getch();
   }
}
```

## Задача 251

```
// Кривая Гильберта

#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <dos.h>
#define PATHTODRIVER "c:\\borlandc\\bgi\\"

#define DT  3  // задержка при выводе линий по точкам
#define u 10   // величина штриха кривой Гильберта

void Gilbert(int p); // вычерчивает кривую Гильберта
```

```
void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk) {
      outtextxy(10,10,"Кривая Гильберта ...");
      Gilbert(4);
      outtextxy(10,25,"Для завершения нажмите <Enter>");
      getch();
      closegraph();
   }
   else {
      printf("Ошибка: %d\n", errorcode);
      printf("\Для завершения программы нажмите <Enter>");
      getch();
   }
}

// Кривая Гильберта состоит из четырех элементов: a, b, c и d.
// Каждый элемент строит соответствующая функция.
void a(int i);
void b(int i);
void c(int i);
void d(int i);

void my_lineto(int x2, int y2); // вычерчивает по точкам
                                // линию

void Gilbert(int p) // p - порядок кривой Гильберта
{
   moveto(450,50);
   a(p);
}

// Элементы кривой.
```

```
void a(int i)
{
    if (i > 0) {
      d(i-1); my_lineto(getx() - u, gety());
      a(i-1); my_lineto(getx(), gety() + u);
      a(i-1); my_lineto(getx() + u, gety());
      b(i-1);
    }
}

void b(int i)
{
   if (i > 0)
   {
      c(i-1); my_lineto(getx(),gety() - u);
      b(i-1); my_lineto(getx() + u, gety());
      b(i-1); my_lineto(getx(),gety() + u);
      a(i-1);
   }
}

void c(int i)
{
   if (i > 0) {
      b(i-1); my_lineto(getx() + u,gety());
      c(i-1); my_lineto(getx(), gety() - u);
      c(i-1); my_lineto(getx() - u,gety());
      d(i-1);
   }
}

void d(int i )
{
   if (i > 0) {
      a(i-1); my_lineto(getx(),gety() + u);
      d(i-1); my_lineto(getx() - u,gety());
      d(i-1); my_lineto(getx(),gety() - u);
      c(i-1);
   }
}
```

```
// вычерчивает по точкам линию
void my_lineto(int x2, int y2)
{
    int x1,y1;  // координаты начала прямой
                // x2, y2 — координаты конца
    int x,y;    // координаты текущей точки
    int dx;     // приращение аргумента
    int dy;     // приращение y при рисовании
                // вертикальной линии
    int color;  // цвет линии
    int a,b;    // коэф-ты уравнения прямой
    int n;      // кол-во точек
    int i;

    x1 = getx();
    y1 = gety();
    if ( x1 != x2 )
    {
      // не вертикальная линия
      a = (y2-y1)/(x2-x1);
      b = y1- a * x1;
      n = abs(x2-x1)+1;
      if (x2 > x1)
        dx = 1;
      else dx = -1;
      x = x1;
      color = getcolor();
      for (i = 1; i<= n; i++)
      {
        y = a*x + b;
        putpixel(x,y,color);
        delay(DT);
        x += dx;
      }
    }
    else { // вертикальная линия
       n = abs(y2-y1);
       if (y2 > y1)
          dy = 1;
       else dy = -1;
          x = x1;
```

```
        y = y1;
        color = getcolor();
        for (i = 1; i<=n; i++)
        {
            putpixel(x, y, color);
            delay(DT);
            y += dy;
        }
    }
    putpixel(x2, y2, color);
    moveto(x2, y2);
}
```

## Задача 252

```
// Кривая Серпинского
#define u 5   // Длина штриха, определяет величину кривой
#define DT 25 // определяет скорость вычерчивания кривой
#define PATHTODRIVER "c:\\borlandc\\bgi\\"

#include <graphics.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>

// кривая Серпинского состоит из четырех
// элементов: a, b, c и d
// каждый элемент строит соответствующая функция
void a(int i);
void b(int i);
void c(int i);
void d(int i);

// вычерчивает прямую из текущей точки в заданную
// координаты конца задаются в приращениях
#define linetodxy(dx,dy) lineto(getx()+dx,gety()+dy)

void far lineto(int x2, int y2); // вычерчивает линию по
                                 // точкам
// заменяет стандартную функцию, чтобы
// процесс вычерчивания кривой Серпинского можно было видеть
```

```
// элементы кривой
void a(int i)
{
    if (i > 0)
    {
        a(i-1); linetodxy(u, u);
        b(i-1); linetodxy(2*u,0);
        d(i-1); linetodxy(u, -u);
        a(i-1);
    }
}

void b(int i)
{
    if (i > 0)
    {
        b(i-1);linetodxy(-u,u);
        c(i-1);linetodxy(0, 2*u);
        a(i-1);linetodxy(u,u);
        b(i-1);
    }
}

void c(int i)
{
    if (i > 0)
    {
        c(i-1);linetodxy(-u,-u);
        d(i-1);linetodxy(-2*u,0);
        b(i-1);linetodxy(-u,u);
        c(i-1);
    }
}

void d(int i)
{
    if (i > 0)
    {
        d(i-1);linetodxy(u,-u);
        a(i-1);linetodxy(0,-2*u);
        c(i-1);linetodxy(-u,-u);
```

```
        d(i-1);
    }
}

void main(void)
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   initgraph(&gdriver, &gmode, PATHTODRIVER);
   errorcode = graphresult();

   if (errorcode == grOk)
   {
        int p; // порядок кривой
        puts("Программа строит кривую Серпинского.");
        puts("Введите порядок кривой (1-4) \
        и нажмите <Enter>");
        printf("->");
        scanf("%i", &p);
        printf("Кривая Серпинского %i-го порядка\n", p);

        moveto(100,100);

        // кривая Серпинского
        a(p); linetodxy(u,u);
        b(p); linetodxy(-u,u);
        c(p); linetodxy(-u,-u);
        d(p); linetodxy(u,-u);

        puts("Для завершения нажмите <Enter>");
        getch();
        closegraph();
    }

    else
    {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения нажмите <Enter>");
```

```
      getch();
    }
}

// вычерчивает по точкам линию
// подменим этой функцией стандартную, чтобы
// видеть процесс вычерчивания кривой
#include <dos.h>

void far lineto(int x2, int y2)
{
    int x1,y1;  // координаты начала прямой,
                // x2,y2 — координаты конца
    int x,y;    // координаты текущей точки
    int dx;     // приращение аргумента
    int dy;     // приращение y при рисовании
                // вертикальной линии
    int color;  // цвет линии
    int a,b;    // коэф-ты уравнения прямой
    int n;      // кол-во точек
    int i;

    x1 = getx();
    y1 = gety();
    color = getcolor();

    if ( x1 != x2 )
    {
        // не вертикальная линия
        a = (y2-y1)/(x2-x1);
        b = y1- a * x1;
        n = abs(x2-x1)+1;
        if (x2 > x1)
           dx = 1;
        else
           dx = -1;

      x = x1;
      for (i = 1; i<= n; i++)
      {
```

```
        y = a*x + b;
        putpixel(x,y,color);
        delay(DT);
        x += dx;
      }
    }

    else // вертикальная линия
    {
       n = abs(y2-y1);
       if (y2 > y1)
          dy = 1;
       else dy = -1;

       x = x1;
       y = y1;
       for (i = 1; i<=n; i++)
       {
           putpixel(x, y, color);
           delay(DT);
           y += dy;
       }
    }
    putpixel(x2, y2, color);
    moveto(x2, y2);
}
```

# ЧАСТЬ III. СПРАВОЧНИК

## Структура программы

Программа на языке C++ представляет собой набор функций, одна из которых имеет имя `main`.

В простейшем случае программа представляет собой одну единственную функцию `main`.

Если функция `main` получает параметры и возвращает результат, то она объявляется так:

```
int main(int argc, char* argv[])
{
    /*

    здесь инструкции

    */
    return(значение);
}
```

Если функция `main` не получает параметры и не возвращает результат, то она объявляется так:

```
void main()
{
    /*

    здесь инструкции

    */

}
```

## Основные типы данных

К основным типам данных языка C/C++ относятся:

- целые числа (**int** и др.);

- дробные (действительные) числа (**float** и др.);
- символы (**char**).

Целые числа и числа с плавающей точкой могут быть представлены в различных форматах.

## Целые числа

| Формат | Бит | Диапазон значений |
|---|---|---|
| **int** | 16 | –32 768 ... 32 767 |
| **short int** | 16 | –32 768 ... 32 767 |
| **unsigned int** | 16 | 0 ... 65 535 |
| **enum** | 16 | –32 768 ... 32 767 |
| **long** | 32 | –2 147 483 648 ... 2 147 483 647 |
| **unsigned long** | 32 | 0 ... 4 294 967 295 |

## Дробные числа

| Формат | Бит | Диапазон значений |
|---|---|---|
| **float** | 32 | $3{,}4 \times 10^{-38}$ ... $3{,}4 \times 10^{+38}$ |
| **double** | 64 | $1{,}7 \times 10^{-308}$ ... $1{,}7 \times 10^{+308}$ |
| **long double** | 80 | $3{,}4 \times 10^{-4932}$ ... $1{,}1 \times 10^{+4932}$ |

## Символы

| Тип | Бит | Диапазон значений |
|---|---|---|
| **unsigned char** | 8 | 0 ... 255 |
| **char** | 8 | –128 ... 127 |

**Примечание**

При использовании типа **char** символы русского алфавита кодируются отрицательными числами. Чтобы коды символов однозначно соответствовали кодировке ASCII (*см. Приложение*), следует использовать тип **unsigned char**.

## Строки

Строка — это массив символов.

Объявление:

**char** *Имя*[*Длина*];

# Массивы

Объявление одномерного массива:

```
Тип Имя[КоличествоЭлементов];
```

Объявление двумерного массива:

*Тип Имя*[*КоличествоЭлементов1*][*КоличествоЭлементов2*];

**Примечание**

Элементы массива нумеруются с нуля. При обращении к элементу массива индекс может меняться от 0 до (N-1), где N — количество элементов, указанное в инструкции объявления массива.

# Инструкция присваивания

| Инструкция | Соответствующая "обычная" инструкция присваивания |
|---|---|
| x++ | x = x + 1 |
| x-- | x = x - 1 |
| x += y | x = x + y |
| x -= y | x = x – y |
| x *= y | x = x*y |
| x %= y | x = x % y |

# Выбор

## Инструкция *if*

Вариант 1

```
if ( Условие )
{
```

```
    // Здесь инструкции, которые будут
    // выполнены, если значение
    // выражения Условие не равно нулю
}
```

Вариант 2

```
if ( Условие )
{
    // Здесь инструкции, которые будут
    // выполнены, если значение
    // выражения Условие не равно нулю
}
else
{
    // Здесь инструкции, которые будут
    // выполнены, если значение
    // выражения Условие равно нулю
}
```

## Инструкция *switch*

Вариант 1

```
switch ( выражение )
{
    case константа1: инструкция1; break;
    case константа2: инструкция2; break;

    case константаj: инструкцияj; break;
    default:         инструкция;  break;
}
```

Вариант 2

```
switch ( выражение )
{
    case константа1: инструкция1; break;
    case константа2: инструкция2; break;

    case константаj: инструкцияj; break;
}
```

# Циклы

## Инструкция *for*

Синтаксис:

```
for ( Инициализация; УсловиеВыполнения; Изменение )
{
    // Здесь инструкции цикла (тело цикла)
}
```

*Инициализация* — инструкция инициализации счетчика циклов.

*УсловиеВыполнения* — выражение, значение которого определяет условие выполнения инструкций цикла. Инструкции цикла выполняются до тех пор, пока *УсловиеВыполнения* истинно, т. е. не равно нулю.

*Изменение* — инструкция изменения параметра цикла. Как правило, эта инструкция изменяет значение переменной, которая входит в *УсловиеВыполнения*.

## Инструкция *do while*

Синтаксис:

```
do
{
    // Инструкции цикла (тело цикла)
}
while ( УсловиеПовторения );
```

Сначала выполняются инструкции цикла (тело цикла), затем проверяется значение выражения *УсловиеПовторения*, и если условие истинно, не равно нулю, то инструкции цикла выполняются еще раз. И так до тех пор, пока *УсловиеПовторения* не станет ложным, т. е. равным нулю.

## Инструкция *while*

Синтаксис:

```
while ( УсловиеВыполнения )
{
    // Инструкции цикла (тело цикла)
}
```

Сначала проверяется значение выражения *УсловиеВыполнения*. Если оно не равно нулю, т. е. условие истинно, то выполняются инструкции цикла (тело цикла). Затем снова проверяется значение выражения *УсловиеВыполнения*, и если оно не равно нулю, инструкции цикла выполняются еще раз. И так до тех пор, пока значение выражения *УсловиеВыполнения* не станет равным нулю.

# Объявление функции

```
Тип Имя(Тип1 Параметр1, ... Типj Параметрj)
{
    // Объявления переменных
    // и инструкции функции

    return ( Значение );
}
```

*Тип* — тип функции, тип значения, которое функция возвращает. Если функция не возвращает значение, то ее тип — **void**. В теле функции инструкцию **return** в этом случае не пишут.

*Имя* — имя функции.

*Типj*, *Параметрj* — тип и параметр функции. Если параметр используется для возврата результата, то параметр должен быть ссылкой, т. е. перед именем параметра должен быть символ *.

# Стандартные функции

При описании функций приняты следующие обозначения:

- имена функций выделены шрифтом Courier;
- перед именем функции указан ее тип, т. е. тип значения, которое функция возвращает;
- параметры выделены курсивом. В качестве параметра могут использоваться константы, переменные или выражения соответствующих типов;
- после описания функции указано имя заголовочного файла, ссылка на который должна быть включена в текст программы для того, чтобы функция была доступна.

# Математические функции

## *abs, fabs*

Синтаксис:

```
int     abs(int x);
double fabs(double x);
```

Возвращает целое (`abs`) или дробное (`fabs`) абсолютное значение аргумента, в качестве которого можно использовать выражение соответствующего типа.

Заголовочный файл: `<math.h>`

## *acos, asin, atan, acosl, asinl, atanl*

Синтаксис:

```
double acos (double x);
double asin (double x);
double atan (double x);

long double acosl(long double x);
long double asinl(long double x);
long double atanl(long double x);
```

Возвращает выраженную в радианах величину угла, косинус, синус или тангенс которого передан соответствующей функции в качестве аргумента. Аргумент функции должен находиться в диапазоне от –1 до 1.

Заголовочный файл: `<math.h>`

## *cos, sin, tan*
## *cosl, sinl, tanl*

Синтаксис:

```
double cos (double x);
double sin (double x);
double tan (double x);

long double cosl(long double x);
long double sinl(long double x);
long double tanl(long double x);
```

Возвращает синус, косинус или тангенс угла. Величина угла должна быть задана в радианах.

Заголовочный файл: `<math.h>`

## exp, expl

Синтаксис:

```
     double exp(double x);
long double exp(long double (x));
```

Возвращает значение, равное экспоненте аргумента ($e^x$, где e — основание натурального логарифма).

Заголовочный файл: `<math.h>`

## pow, powl

Синтаксис:

```
     double pow (double x, double y);
long double powl(long double (x), long double (y));
```

Возвращает значение, равное $x^y$.

Заголовочный файл: `<math.h>`

## sqrt

Синтаксис:

```
double sqrt(double x);
```

Возвращает значение, равное квадратному корню из аргумента.

Заголовочный файл: `<math.h>`

## rand

Синтаксис:

```
int rand(void);
```

Возвращает случайное целое число в диапазоне от `0` до `RAND_MAX`. Перед первым обращением к функции `rand` необходимо инициализировать генератор случайных чисел. Для этого надо вызвать функцию **`srand`**.

### srand

Синтаксис:

```
void srand(unsigned x);
```

Инициализирует генератор случайных чисел. Обычно в качестве параметра функции используют переменную, значение которой предсказать заранее нельзя, например это может быть текущее время.

Заголовочный файл: <stdlib.h>

## Функции преобразования

Приведенные ниже функции выполняют преобразование строк в числовое значение и чисел в строковое представление.

### atof

Синтаксис:

```
double atof(const char* s);
```

Возвращает дробное число, значение которого передано функции в качестве аргумента. Функция обрабатывает строку до тех пор, пока символы строки являются допустимыми. Строка может быть значением числа как в формате с плавающей точкой, так и в экспоненциальном формате.

Заголовочный файл: <stdlib.h>

### atoi, atol

Синтаксис:

```
int atoi(const char* s);
long atol(const char* s);
```

Возвращает целое соответствующего типа, изображение которого передано функции в качестве аргумента. Функция обрабатывает символы строки до тех пор, пока не встретит символ, не являющийся десятичной цифрой.

Заголовочный файл: <stdlib.h>

## gcvt

Синтаксис:

```
char *gcvt(double Значение, int Цифр, char* Строка);
```

Преобразует дробное число в строку. При преобразовании делается попытка получить указанное количество значащих цифр, а если это сделать невозможно, то число изображается в форме с плавающей точкой.

Заголовочный файл: `<stdlib.h>`

## itoa, ltoa, ultoa

Синтаксис:

```
char* itoa (int Значение, char* Строка, int Основание);
char* ltoa (long Значение, char* Строка, int Основание);
char* ultoa(unsigned long Значение, char* Строка, int
Основание);
```

Соответственно преобразуют целое, длинное целое и длинное беззнаковое целое в строку. Число изображается в указанной при вызове функции системе счисления.

*Строка* — указатель на строку, куда будет помещено изображение числа. *Основание* — задает основание системы счисления (от 2 до 36).

Максимальная длина строки, формируемой функцией `itoa`, — 17 байт, функциями `ltoa` и `ultoa` — 33 байта.

Заголовочный файл: `<stdlib.h>`

## sprintf

Синтаксис:

```
int sprintf(char *Строка, const char* Формат, СписокПеременных);
```

Выполняет форматированный вывод в строку.

*СписокПеременных* — разделенные запятыми имена переменных, задает переменные, значения которых должны быть выведены. Параметр *Формат* задает способ отображения значений переменных.

Действие функции `sprintf` аналогично действию функции **`printf`**, но вывод выполняется в строку-буфер, а не на экран.

Заголовочный файл: `<stdio.h>`

# Функции ввода-вывода

## *printf*

Синтаксис:

```
int printf(Формат, СписокПеременых);
```

Выводит на экран значения переменных. Формат вывода задается в строке форматирования, которая помимо спецификатора формата может содержать текст и управляющие символы. Значение первой переменной выводится в соответствии с первым спецификатором формата, второй — со вторым, и т. д.

Спецификаторы формата (необязательный параметр `n` задает ширину поля вывода).

| Спецификатор | Форма вывода |
|---|---|
| `%ni`<br>`%nd` | Десятичное число со знаком |
| `%nu` | Беззнаковое целое десятичное число |
| `%n.mf` | Дробное число с десятичной точкой. Необязательный параметр `m` задает количество цифр дробной части |
| `%ne` | Дробное число с десятичной точкой или, если число не может быть представлено в форме с десятичной точкой, в экспоненциальной форме |
| `%ns` | Строка символов |
| `%nc` | Символ |

Управляющие и специальные символы.

| Символ | Действие |
|---|---|
| `\n` | Переводит курсор в начало следующей строки |
| `\t` | Переводит курсор в очередную позицию табуляции |
| `\\` | Бэкслэш |
| `\'` | Кавычка |

Заголовочный файл: `<stdio.h>`

## scanf

Синтаксис:

```
int scanf(const char* Формат, СписокАдресовПеременных);
```

Вводит с клавиатуры значения переменных, в соответствии с указанным спецификатором формата. Первая переменная получает значение в соответствии с первым спецификатором формата, вторая — со вторым и т. д.

**Замечание**

В качестве параметра функции `scanf` должны передаваться *адреса* переменных, а не их имена.

| Спецификатор | Вводит |
|---|---|
| %i<br>%d | Десятичное число со знаком |
| %u | Беззнаковое целое десятичное число |
| %f<br>%e | Дробное число |
| %s | Строка символов |
| %c | Символ |

Заголовочный файл: `<stdio.h>`

## puts

Синтаксис:

```
puts(const char* Строка);
```

Выводит на экран строку символов и переводит курсор в начало следующей строки экрана. В качестве параметра функции можно использовать строковую константу или строковую переменную.

Заголовочный файл: `<stdio.h>`

## gets

Синтаксис:

```
char *gets(char* s);
```

Вводит с клавиатуры строку символов. Вводимая строка может содержать пробелы.

Заголовочный файл: `<stdio.h>`

## putch

Синтаксис:

```
int putch(int c);
```

Выводит на экран символ.

Заголовочный файл: `<conio.h>`

## getch

Синтаксис:

```
int getch(void);
```

Возвращает код символа нажатой клавиши. Если нажата служебная клавиша, то функция `getch` возвращает 0. В этом случае, для того, чтобы определить, какая служебная клавиша нажата, нужно обратиться к функции `getch` еще раз.

**Замечание**

Функция `getch` не выводит на экран символ, соответствующий нажатой клавише.

Заголовочный файл: `<conio.h>`

## cputs

Синтаксис:

```
cputs(const char* Строка);
```

Выводит на экран строку. Цвет выводимых символов можно задать при помощи функции `textcolor`, цвет фона — при помощи функции `textbackground`.

**Замечание**

Для перехода к началу следующей строки вместо `\n` следует использовать символы `\n\r`, иначе курсор лишь переводится на но-

вую строку, но не возвращается к левой границе окна. То же самое относится и к функции **cprintf**.

Заголовочный файл: <conio.h>

## cprintf

Как и функция printf, функция cpfintf используется для вывода на экран сообщений и значений переменных. При этом имеется возможность задать цвет выводимых символов (функция textcolor) и цвет фона (textbackground).

Заголовочный файл: <conio.h>

## textcolor

Синтаксис:

```
void textcolor(int Цвет);
```

Задает цвет для выводимого функциями cputs и cprintf текста. В качестве параметра *Цвет* обычно используют одну из перечисленных ниже именованных констант.

| Цвет | Константа | Значение константы |
|---|---|---|
| Черный | BLACK | 0 |
| Синий | BLUE | 1 |
| Зеленый | GREEN | 2 |
| Бирюзовый | CYAN | 3 |
| Красный | RED | 4 |
| Сиреневый | MAGENTA | 5 |
| Коричневый | BROWN | 6 |
| Светло-серый | LIGHTGRAY | 7 |
| Серый | DARKGRAY | 8 |
| Голубой | LIGHTBLUE | 9 |
| Светло-зеленый | LIGHTGREEN | 10 |
| Светло-бирюзовый | LIGHTCYAN | 11 |
| Алый | LIGHTRED | 12 |

*(окончание)*

| Цвет | Константа | Значение константы |
|---|---|---|
| Светло-сиреневый | LIGHTMAGENTA | 13 |
| Желтый | YELLOW | 14 |
| Белый (яркий) | WHITE | 15 |

Заголовочный файл: <conio.h>

## textbackground

Синтаксис:

```
void textbackground(int Цвет);
```

Задает цвет фона, на котором появляется текст, выводимый функциями cputs и cprintf. В качестве параметра *Цвет* обычно используют одну из перечисленных ниже именованных констант.

| Цвет | Константа | Значение константы |
|---|---|---|
| Черный | BLACK | 0 |
| Синий | BLUE | 1 |
| Зеленый | GREEN | 2 |
| Бирюзовый | CYAN | 3 |
| Красный | RED | 4 |
| Сиреневый | MAGENTA | 5 |
| Коричневый | BROWN | 6 |
| Светло-серый | LIGHTGRAY | 7 |

Заголовочный файл: <conio.h>

## gotoxy

Синтаксис:

```
void gotoxy(int x, int y)
```

Переводит курсор в позицию с указанными координатами. Координата *x* задает номер колонки, координата *y* — номер стро-

ки, на пересечении которых находится знакоместо, куда переводится курсор.

Заголовочный файл: <conio.h>

### clrscr

Синтаксис:

```
void clrscr(void)
```

Очищает экран и закрашивает его цветом, заданным функцией textbackground.

Заголовочный файл: <conio.h>

### window

Синтаксис:

```
void window(int x1, int y1, int x2, int y2);
```

Определяет окно — область экрана. Параметры *x1*, *y1* задают координаты левого верхнего угла окна относительно экрана, параметры *x2*,*y2* — правого нижнего.

Заголовочный файл: <conio.h>

## Функции работы с файлами

### fopen

Синтаксис:

```
FILE* fopen(const char * Имя, const char* Режим)
```

Открывает файл с указанным именем для действия, которое задается параметром *Режим*.

| Режим | Действие |
|---|---|
| r | Только запись. Файл открывается только для чтения |
| w | Чтение. Файл открывается для записи. Если файл с указанным в качестве первого параметра функции fopen уже существует, то новые данные записываются поверх старых, т. е. старый файл фактически уничтожается |

*(окончание)*

| Режим | Действие |
|---|---|
| A | Добавление. Файл открывается для записи данных в конец существующего файла. Если файл с указанным в качестве первого параметра функции `fopen` не существует, то он будет создан |

Если файл открывается как текстовый, то после символьной константы, определяющей режим открытия файла, нужно добавить символ `t`. Например, строка `rt` задает, что для чтения открывается текстовый файл.

В случае успешного открытия файла функция `fopen` возвращает указатель на поток, из которого можно читать или в который можно записывать. Если по какой-либо причине операция открытия файла не была выполнена, `fopen` возвращает `NULL`. В этом случае, чтобы получить информацию о причине ошибки, следует обратиться к функции `ferror`.

Заголовочный файл: `<stdio.h>`

## fprintf

Синтаксис:

```
int fprintf(FILE *Поток, Формат, СписокПеременых);
```

Выполняет форматированный вывод (см. `printf`) в файл, связанный с потоком, указанным в качестве первого параметра.

Файл, связанный с потоком, должен быть открыт как текстовый, в режиме, допускающем запись (см. `fopen`).

Заголовочный файл: `<stdio.h>`

## fscanf

Синтаксис:

```
int fscanf(FILE *Поток,
           const char* Формат, СписокАдр);
```

Выполняет форматированное (см. `scanf`) чтение значений переменных из файла, связанного с потоком, указанным в качестве первого параметра.

Файл, связанный с потоком, должен быть открыт как текстовый, в режиме, допускающем чтение (см. fopen).

Заголовочный файл: <stdio.h>

## fgets

Синтаксис:

```
char* fgets(char *Строка,
            int КолСимволов, FILE *Поток)
```

Читает из указанного потока символы и записывает их в строку, указанную при вызове функции. Чтение заканчивается, если прочитан символ с номером КолСимволов-1 или если очередной символ является символом новой строки.

Прочитанный из файла символ новой строки заменяется нулевым символом.

Файл, связанный с потоком, должен быть открыт как текстовый, в режиме, допускающем чтение (см. fopen).

Заголовочный файл: <stdio.h>

## fputs

Синтаксис:

```
char* fputs(char *Строка, FILE *Поток)
```

Записывает в указанный поток строку символов. Символ конца строки, нуль-символ, в поток не записывается.

Файл, связанный с потоком, должен быть открыт как текстовый, в режиме, допускающем запись (см. fopen).

Заголовочный файл: <stdio.h>

## ferror

Синтаксис:

```
int ferror(FILE* Поток)
```

Возвращает ненулевое значение, если последняя операция с указанным потоком завершилась ошибкой.

Заголовочный файл: <stdio.h>

### feof

Синтаксис:

```
int feof(FILE* Поток)
```

Возвращает ненулевое значение, если в результате выполнения последней операции чтения из потока достигнут конец файла.

Заголовочный файл: <stdio.h>

### fclose

Синтаксис:

```
int fclose(FILE* Поток)
```

Закрывает указанный поток.

Заголовочный файл: <stdio.h>

## Функции работы со строками

### strcat

Синтаксис:

```
char *strcat(char* Строка1, const char* Строка2)
```

Объединяет строки *Строка1* и *Строка2* и записывает результат в строку *Строка1*.

Заголовочный файл: <string.h>

### strcpy

Синтаксис:

```
char *strcpy(char* Строка1, const char* Строка2)
```

Копирует строку *Строка1* в строку *Строка2*.

Заголовочный файл: <string.h>

### strlen

Синтаксис:

```
int strlen(const char* Строка)
```

Возвращает длину строки. Нулевой символ не учитывается.

Заголовочный файл: <string.h>

## strcmp

Синтаксис:

```
int strcmp (const char* Строка1,
            const char* Строка2)
```

Сравнивает строки *Строка1* и *Строка2*. Возвращает 0, если строки равны, число меньше нуля, если *Строка1* < *Строка2* и число больше нуля, если *Строка1* > *Строка2*.

Заголовочный файл: <string.h>

## strlwr

Синтаксис:

```
char* strlwr(char* Строка)
```

Преобразует строчные символы строки в прописные (обрабатывает только буквы латинского алфавита).

Заголовочный файл: <string.h>

## strupr

Синтаксис:

```
char* strupr(char* Строка)
```

Преобразует прописные символы строки в строчные (обрабатывает только буквы латинского алфавита).

Заголовочный файл: <string.h>

## strset

Синтаксис:

```
char* strset(char* Строка, char Символ)
```

Заполняет строку указанным при вызове функции символом.

Заголовочный файл: <string.h>

### strchr

Синтаксис:

```
char* strchr(const char* Строка, int Символ)
```

Выполняет поиск символа в строке и возвращает указатель на первый найденный символ или, если символ найден, NULL.

Заголовочный файл: <string.h>

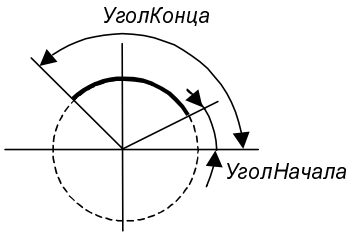
## Функции графического режима

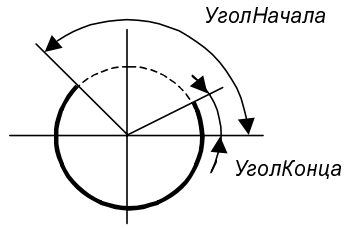
### arc

Синтаксис:

```
void arc(int x, int y, int УголНачала, int УголКонца,
int Радиус);
```

Вычерчивает дугу с центром в точке с координатами (*x*, *y*). Параметры *УголНачала* и *УголКонца* задают круговые координаты начальной и конечной точек линии дуги, которая вычерчивается против часовой стрелки от начальной точки к конечной. Угловые координаты задаются в градусах. Значение угловой координаты возрастает против часовой стрелки. Параметр *Радиус* задает радиус дуги.




Линия дуги вычерчивается цветом, заданным функцией setcolor.

Заголовочный файл: <graph.h>

### bar

Синтаксис:

```
void bar(int x1, int y1, int x2, int y2);
```

Вычерчивает закрашенный прямоугольник. Параметры *x1* и *y1* задают положение левого верхнего угла прямоугольника, *x2* и *y2* — правого нижнего.

Цвет и стиль заливки прямоугольника задаются функцией `setfillstyle`.

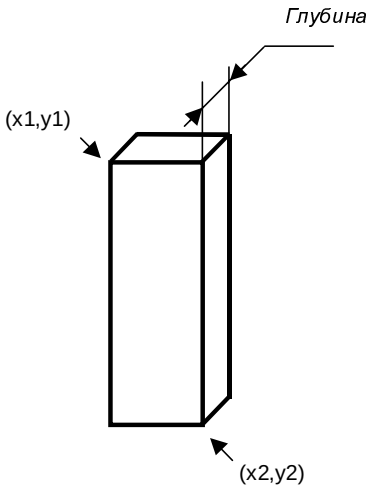
Заголовочный файл: `<graph.h>`

## bar3d

Синтаксис:

```
void bar3d(int x1,int y1,int x2, int y2,
           int Глубина, int В_Грань);
```

Вычерчивает параллелепипед. Параметры *x1* и *y1* задают положение левого верхнего, а *x2* и *y2* — правого нижнего угла ближней грани параллелепипеда. Параметр *Глубина* задает расстояние между передней и задней гранями, параметр *В_Грань* определяет, нужно ли вычерчивать границу верхней грани. Если параметр *В_Грань* равен нулю, то линия границы верхней грани не вычерчивается.



Цвет и стиль закраски ближней грани параллелепипеда можно задать при помощи функции `setfillstyle`, цвет линий границы — при помощи функции `setcolor`.

Заголовочный файл: `<graph.h>`

## circle

Синтаксис:

```
void circle(int x, int y, int r)
```

Вычерчивает окружность радиуса *r* с центром в точке с координатами (*x*, *y*).

Цвет окружности можно задать при помощи функции `setcolor`.

Заголовочный файл: `<graph.h>`

## drawpoly

Синтаксис:

```
void drawpoly(int КолТочек, int * Координаты);
```

Вычерчивает замкнутую ломаную линию, состоящую из отрезков прямых. Параметр *КолТочек* задает количество точек в результате последовательного соединения которых получается ломаная. Параметр *Координаты* задает массив координат узловых точек ломаной. Нулевой и первый элементы массива *Координаты* содержат координаты первой точки (*x* и *y*), второй и третий элементы содержат координаты второй точки и т. д.

Заголовочный файл: `<graph.h>`

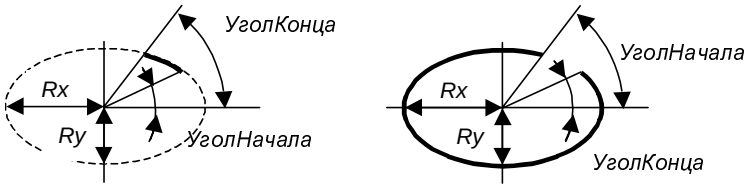
## ellipse

Синтаксис:

```
void ellipse(int x, int y, int УголНачала,int УголКонца,
             int РадиусX, int РадиусY );
```

Вычерчивает эллипс или дугу эллипса с центром в точке с координатами (*x*, *y*). Параметры *УголНачала* и *УголКонца* задают круговые координаты начальной и конечной точек линии эллипса, которая вычерчивается против часовой стрелки от на-

чальной точки к конечной. Угловые координаты задаются в градусах. Значение угловой координаты возрастает против часовой стрелки. Параметры *РадиусX* и *РадиусY* задают горизонтальный и вертикальный радиусы эллипса.




Линия эллипса или дуги вычерчивается цветом, установленным функцией `setcolor`.

Заголовочный файл: `<graph.h>`

## getmaxx, getmaxy

Синтаксис:

```
int getmaxx(void);
int getmaxy(void);
```

Функция `getmaxx` возвращает координату *x* крайней правой точки экрана, функция `getmaxy` — координату *y* крайней нижней точки экрана.

Заголовочный файл: `<graph.h>`

## getx, gety

Синтаксис:

```
int getx(void);
int gety(void);
```

Возвращает координату `x` (`y`) указателя вывода.

Заголовочный файл: `<graph.h>`

## graphresult

Синтаксис:

```
int graphresult(void);
```

Возвращает результат (код ошибки) последней выполненной графической операции. Если операция выполнена успешно, функция возвращает ноль. Код ошибки выполнения графической операции устанавливают функции: `bar`, `bar3d`, `initgraph`, `pieslice`, `setfillpattern`, `setfillstyle`, `setlinestyle`, `settextstyle` и др.

Заголовочный файл: `<graph.h>`

## grapherrormsg

Синтаксис:

```
char* grapherrormsg(int КодОшибки);
```

Возвращает указатель на строку, содержащую сообщение, соответствующее коду ошибки выполнения графической операции, указанному при вызове функции.

Заголовочный файл: `<graph.h>`

## initgraph

Синтаксис:

```
void initgraph(int* Driver, int* Mode, char* Path);
```

Инициализирует графический режим. Параметр *Driver* определяет драйвер видеосистемы, параметр *Mode* — режим работы видеосистемы, параметр *Path* — путь к файлу драйвера.

**Замечание**

Обычно в качестве параметра *Driver* используют указатель на целую константу, значение которой равно DETECT. В этом случае функция `initgraph` сама определяет тип графического адаптера и устанавливает для него наилучший режим.

Заголовочный файл: `<graph.h>`

## line

Синтаксис:

```
void line(int x1, int y1, int x2, int y2);
```

Вычерчивает линию из точки с координатами *x1*, *y1* в точку с координатами *x2*, *y2*.

Цвет линии можно задать при помощи функции `setcolor`, стиль — при помощи функции `setlinestyle`.

Заголовочный файл: `<graph.h>`

## lineto

Синтаксис:

```
void lineto(int x, int y);
```

Вычерчивает линию от текущего положения указателя вывода до точки, координаты которой указаны при вызове. Линия вычерчивается стилем, установленным функцией `setlinestyle`. Цвет линии можно задать, вызвав функцию `setcolor`.

Заголовочный файл: `<graph.h>`

## linerel

Синтаксис:

```
void linerel(int dx, int dy);
```

Вычерчивает линию из точки текущего положения указателя вывода (*xt*,*yt*) в точку с координатами (*xt+dx*, *yt+dy*), т. е. координаты конца линии задаются в приращениях относительно текущих координат указателя вывода.

Линия вычерчивается стилем, который устанавливается функцией `setlinestyle`. Цвет линии можно задать, вызвав функцию `setcolor`.

**Замечание**

Координаты указателя вывода можно получить при помощи функций `getx` и `gety`.

Заголовочный файл: `<graph.h>`

## moveto

Синтаксис:

```
void moveto(int x, int y);
```

Перемещает указатель вывода в точку с указанными координатами.

Заголовочный файл: <graph.h>

## moverel

Синтаксис:

```
void moverel(int dx, int dy);
```

Перемещает указатель вывода на *dx* и *dy* пикселов. Если значение параметра *dx* (*dy*) положительное, то указатель перемещается вниз (влево), если отрицательное, то — вверх (вправо).

Заголовочный файл: <graph.h>

## outtext

Синтаксис:

```
void outtext(const char* Текст);
```

Выводит строку символов *Текст* от текущего положения указателя вывода и перемещает указатель вывода в точку, расположенную за последним выведенным символом.

**Замечание**

Строка, передаваемая функции `outtext`, не должна содержать символов форматирования, например `\n`.

Цвет выводимых символов можно задать при помощи функции `setcolor`, шрифт — `settextstyle`.

Заголовочный файл: <graph.h>

## outtextxy

Синтаксис:

```
void outtextxy(int x, int y, const char* Текст);
```

Устанавливает указатель вывода в точку с координатами (*x*, *y*) и выводит от нее строку *Текст*, при этом указатель вывода своего положения не меняет, т. е. остается в точке с координатами (*x*, *y*).

Цвет выводимых символов можно задать при помощи функции `setcolor`, шрифт — `settextstyle`.

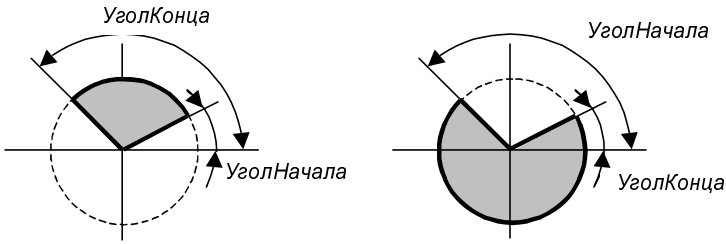
Заголовочный файл: `<graph.h>`

## pieslice

Синтаксис:

```
void pieslice(int x, int y, int УголНачала, int Уголконца,
int Радиус);
```

Вычерчивает круговой сектор радиуса *Радиус* с центром в точке с координатами (*x*, *y*). Параметры *УголНачала* и *УголКонца* задают круговые координаты начальной и конечной точек линии окружности, которая вычерчивается против часовой стрелки от начальной к конечной точке. Угловые координаты задаются в градусах. Значение угловой координаты возрастает против часовой стрелки. Нулевому углу соответствует горизонтальный отрезок, проведенный из точки (*x*, *y*) в сторону возрастания координаты *x*. Если *УголНачала*=0, а *УголКонца*=360, то функция `pieslice` вычерчивает круг.




Сектор закрашивается стилем и цветом, установленными функцией `setfillslyle`, линия границы вычерчивается цветом, установленным функцией `setcolor`.

Заголовочный файл: `<graph.h>`

## putpixel

Синтаксис:

```
void putpixel(int x, int y, int Цвет);
```

Окрашивает пиксел, точку с координатами (*x*, *y*), цветом *Цвет*. В качестве параметра *Цвет* обычно используют именованную константу (см. `setcolor`).

Заголовочный файл: `<graph.h>`

## rectangle

Синтаксис:

```
void rectangle(int x1, int y1, int x2, int y2);
```

Вычерчивает прямоугольник. Параметры *x1* и *y1* задают положение левого верхнего угла прямоугольника, *x2* и *y2* — правого нижнего.

Вид (стиль линии) контура прямоугольника можно задать при помощи функции `setlinestyle`, цвет — при помощи функции `setcolor`.

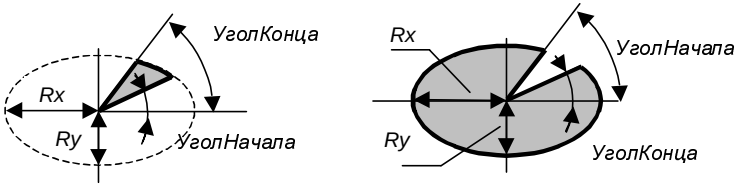
Заголовочный файл: `<graph.h>`

## sector

Синтаксис:

```
void sector(int x, int y, int Угол1, int Угол2, int РадиусX,
int РадиусY);
```

Вычерчивает эллиптический (*РадиусX* ≠ *РадиусY*) или круговой (*РадиусX* = *РадиусY*) сектор. Параметры *x* и *y* задают координаты центра сектора. Параметры *Угол1* и *Угол2* — углы прямых, ограничивающих сектор, параметры *РадиусX* и *РадиусY* — радиусы эллипса по осям *X* и *Y*, из которого "вырезается" сектор. Нулевому углу соответствует горизонтальный отрезок, проведенный из точки (x, y) в сторону возрастания координаты *x*. Если *Угол1*=0, а *Угол2*=360, то функция `sector` вычерчивает полный круг (эллипс).



Цвет и стиль заливки можно задать при помощи функции `setfillstyle`, цвет границы сектора — при помощи функции `setcolor`.

Заголовочный файл: `<graph.h>`

## *setcolor*

Синтаксис:

```
void setcolor(int Цвет);
```

Задает цвет вывода текста (функции `outtextxy` и `outtext`), вычерчивания линий и фигур (функции `line`, `circle`, `recktangle` и др.). В качестве параметра *Цвет* обычно используют именованную константу.

| Цвет | Константа | Значение константы |
|---|---|---|
| Черный | BLACK | 0 |
| Синий | BLUE | 1 |
| Зеленый | GREEN | 2 |
| Бирюзовый | CYAN | 3 |
| Красный | RED | 4 |
| Сиреневый | MAGENTA | 5 |
| Коричневый | BROWN | 6 |
| Светло-серый | LIGHTGRAY | 7 |
| Серый | DARKGRAY | 8 |
| Голубой | LIGHTBLUE | 9 |
| Светло-зеленый | LIGHTGREEN | 10 |
| Светло-бирюзовый | LIGHTCYAN | 11 |
| Алый | LIGHTRED | 12 |
| Светло-сиреневый | LIGHTMAGENTA | 13 |
| Желтый | YELLOW | 14 |
| Белый (яркий) | WHITE | 15 |

Заголовочный файл: `<graph.h>`

## setfillstyle

Синтаксис:

```
void setfillstyle(int Стиль, int Цвет);
```

Устанавливает стиль и цвет заливки (закрашивания), используемый функциями вывода областей (bar, bar3d, sector и др.). В качестве параметра *Стиль* обычно используют одну из именованных констант, список которых приведен ниже. Параметр *Цвет* также задается именованной константой (см. setcolor).

| Константа | Стиль заполнения области |
|---|---|
| EMPTY_FILL | Без заливки (сплошная заливка цветом фона) |
| SOLID_FILL | Сплошная заливка текущим цветом |
| LINE_FILL | Горизонтальная штриховка |
| LTSLASH_FILL | Штриховка под углом 45 градусов влево тонкими линиями |
| SLASH_FILL | Штриховка под углом 45 градусов влево |
| BKSLASH_FILL | Штриховка под углом 45 градусов вправо тонкими линиями |
| LTBKSLASH_FILL | Штриховка под углом 45 градусов вправо |
| HATCH_FILL | Штриховка клеткой |
| XHATCH_FILL | Штриховка под углом 45 градусов редкой косой клеткой |
| INTERLEAVE_FILL | Штриховка под углом 45 градусов частой косой клеткой |
| WIDEDOT_FILL | Заполнение редкими точками |
| CLOSEDOT_FILL | Заполнение частыми точками |
| USER_FILL | Тип заполнения определяется программистом |

Заголовочный файл: <graph.h>

## setlinestyle

Синтаксис:

```
void setlinestyle(int ТипЛинии, int Образец, int Толщина);
```

Устанавливает стиль вычерчиваемых контуров и линий (см. функции `line`, `circle` и др.).

Параметр *ТипЛинии*, в качестве которого обычно используется одна из перечисленных ниже именованных констант, определяет вид линии.

| Константа | Тип линии |
|---|---|
| SOLID_LINE | Сплошная, непрерывная |
| DOTTED_LINE | Пунктирная, с постоянной длиной штрихов |
| CENTER_LINE | Штрих-пунктирная линия |
| DASHED_LINE | Пунктирная, длина штрихов чуть больше, чем у линии типа DOTTED_LINE |
| USERBIT_LINE | Определенный программистом тип линии |

Параметр *Толщина* определяет толщину линии. Линия может быть обычной толщины (константа NORM_WIDTH) или утолщенная (константа THICK_WIDTH).

Параметр *Образец* используется в том случае, если функция `setlinestyle` устанавливает тип линии, определяемый программистом. Значением параметра *Образец* должна быть четырехразрядная шестнадцатеричная константа, кодирующая отрезок линии длиной в 16 пикселов.

Заголовочный файл: `<graph.h>`

## settexstyle

Синтаксис:

```
void settexstyle(int Шрифт, int Ориентация, int Размер);
```

Устанавливает шрифт, размер и ориентацию текста, выводимого функциями `outtextxy` и `outtext`. В качестве параметра *Шрифт* можно использовать одну из перечисленных ниже констант.

| Константа | Значение | Шрифт |
|---|---|---|
| DEFAULT_FONT | 0 | Стандартный. Каждый выводимый символ формируется в квадрате размером 8 на 8 пикселов. |

*(окончание)*

| Константа | Значение | Шрифт |
|---|---|---|
| TRIPLEX_FONT | 1 | Шрифт Triplex |
| SMALL_FONT | 2 | Мелкий |
| SANSSERIF_FONT | 3 | Шрифт SansSerif |
| GOTHIC_FONT | 4 | Готический |

**Замечание**

В шрифтах, отличных от стандартного (DEFAULT_FONT), букв русского алфавита нет.

Параметр *Ориентация* задает ориентацию текста, выводимого функциями outtext и outtextxy. Текст может быть ориентирован обычным образом (значение параметра *Ориентация* в этом случае должно быть равно именованной константе HORIZ_DIR) или вертикально, снизу вверх (в этом случае значение параметра *Ориентация* должно быть равно VERT_DIR).

Заголовочный файл: <graph.h>

# Прочие функции

## delay

Синтаксис:

```
void delay(unsigned Задержка);
```

Обеспечивает задержку на указанное количество миллисекунд.

Заголовочный файл: <dos.h>

## sound

Синтаксис:

```
void sound(unsigned Частота);
```

Обеспечивает вывод звукового сигнала с использованием внутреннего динамика компьютера. Частота сигнала задается в герцах. Динамик будет издавать сигнал до тех пор, пока программа его не выключит при помощи функции nosound.

Ниже приведены частоты, соответствующие первым двум октавам пианино.

| Нота | Частота, герц |
|---|---|
| до | 130 |
| до-диез, ре-бемоль | 138,6 |
| ре | 146,8 |
| ре-диез, ми-бемоль | 155,6 |
| ми | 164,8 |
| фа | 174,6 |
| фа-диез, соль-бемоль | 185 |
| соль | 196 |
| соль-диез, ля-бемоль | 207,7 |
| ля | 220 |
| ля-диез, си-бемоль | 233,1 |
| си | 246,9 |
| до (среднее) | 261,7 |
| до-диез, ре-бемоль | 277,2 |
| ре | 293,7 |
| ре-диез, ми-бемоль | 311,1 |
| ми | 329,6 |
| фа | 349,2 |
| фа-диез, соль-бемоль | 370 |
| соль | 392,0 |
| соль-диез, ля-бемоль | 415,3 |
| ля | 440 |
| ля-диез, си-бемоль | 466,2 |
| си | 493,9 |

Заголовочный файл: `<dos.h>`

## *nosound*

Выключает звуковой сигнал, издаваемый внутренним динамиком компьютера.

Заголовочный файл: `<dos.h>`

# Приложение

## Вывод иллюстраций

В библиотеке graph нет функции, обеспечивающей вывод на экран иллюстрации, находящейся в файле. Программист должен сам разработать такую функцию. Однако эта задача является довольно сложной. Ниже приведен текст разработанной автором функции draw, которая выводит на экран 16-цветную картинку — содержимое bmp-файла. Картинка должна быть создана в среде Microsoft Windows, например, при помощи графического редактора Paint.

```
#include <stdio.h>
#include <graphics.h>

/* Функция draw выводит на экран шестнадцатицветную
   картинку, находящуюся в bmp-файле
   (c) Культин Н. Б., 2001
*/
int draw(int x0, int y0, char* fname)
{
   /*
      x0,y0 — координаты левого верхнего угла
              области вывода
      fname — имя файла картинки;

      Значения функции:
        >0 — высота иллюстрации;
        -1 — не найден файл;
        -2 — картинка не является
             шестнадцатицветной.
   */

   // таблица преобразования кодировки
   // цвета Windows -> DOS
```

```
unsigned char color[16] =
          {0,4,2,6,1,5,3,7,
           8,12,10,14,9,13,11,15};

// прочитав из bmp-файла эту структуру,
// можно получить информацию о картинке:
// ее размере и количестве цветов
struct bmpinfo
{
  char h1,h2;    // файл должен начинаться буквами BM
  unsigned long
     size,     // размер файла, байт
     reserved, // резерв, не используется
     offset,   // смещение данных относительно
               // начала файла
     b,        // не используется
     width,    // ширина картинки
     height;   // высота картинки
  unsigned int
    plans,    // кол-во планов, должно содержать 1
    bpp;      // кол-во бит на пиксел: 1, 4, 8 или 24
};

bmpinfo info;    // информация о картинке

FILE *f;         // файл иллюстрации

int x,y;          // координаты пиксела
unsigned char b;  // байт, прочитанный из файла
unsigned char bh; // сдвинутый на 4 разряда вправо
                  // старший полубайт
unsigned char bl; // четыре младшие бита
                  // прочитанного байта
int nb;           // кол-во байт (кратное четырем)
                  // соответствующее строке
int np;           // кол-во выведенных пикселов
int i,j;

if ((f = fopen(fname, "rb")) == NULL)
   return -1;
// читаем информацию о картинке
```

```
   fread(&info, sizeof(info),1, f);

   if (info.bpp != 4 )
     return -2;        // картинка не 16-цветная

   x = x0;
   y = y0 + info.height;

   nb = (info.width / 8)*4;
   if ((info.width / 8) != 0) nb += 4;

   fseek(f, info.offset, SEEK_SET);

   // вывод иллюстрации
   for (i = 0; i < info.height; i++)
   {
       np = 0;   // кол-во выведенных пикселов
       for (j = 0; j < nb; j++) // вывод строки
       {
           b = fgetc(f);
           if ( np < info.width)
           {
             bh = b >> 4;
             putpixel(x,y,color[bh]);
             x++;
             np++;
           }
           if (np < info.width)
           {
             bl = b & 15;
             putpixel(x,y,color[bl]);
             x++;
             np++;
           }
       }
       x=x0;
       y--;
   }
   fclose(f);
   return info.height;
}
```

# Таблица кодировки символов

Символы с кодами 0—127.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0- | 16- ► | 32- | 48- 0 | 64- @ | 80- P | 96- ` | 112- p |
| 1- ☺ | 17- ◄ | 33- ! | 49- 1 | 65- A | 81- Q | 97- a | 113- q |
| 2- ☻ | 18- ↕ | 34- " | 50- 2 | 66- B | 82- R | 98- b | 114- r |
| 3- ♥ | 19- ‼ | 35- # | 51- 3 | 67- C | 83- S | 99- c | 115- s |
| 4- ♦ | 20- ¶ | 36- $ | 52- 4 | 68- D | 84- T | 100- d | 116- t |
| 5- ♣ | 21- § | 37- % | 53- 5 | 69- E | 85- U | 101- e | 117- u |
| 6- ♠ | 22- ▬ | 38- & | 54- 6 | 70- F | 86- V | 102- f | 118- v |
| 7- | 23- ↨ | 39- ' | 55- 7 | 71- G | 87- W | 103- g | 119- w |
| 8- | 24- ↑ | 40- ( | 56- 8 | 72- H | 88- X | 104- h | 120- x |
| 9- | 25- ↓ | 41- ) | 57- 9 | 73- I | 89- Y | 105- i | 121- y |
| 10- | 26- → | 42- * | 58- : | 74- J | 90- Z | 106- j | 122- z |
| 11- | 27- ← | 43- + | 59- ; | 75- K | 91- [ | 107- k | 123- { |
| 12- | 28- ∟ | 44- , | 60- < | 76- L | 92- \ | 108- l | 124- ¦ |
| 13- | 29- ↔ | 45- - | 61- = | 77- M | 93- ] | 109- m | 125- } |
| 14- ♫ | 30- ▲ | 46- . | 62- > | 78- N | 94- ^ | 110- n | 126- ~ |
| 15- ☼ | 31- ▼ | 47- / | 63- ? | 79- O | 95- _ | 111- o | 127- ⌂ |
| 16- ► | 32- | 48- 0 | 64- @ | 80- P | 96- ` | 112- p | |

Символы с кодами 128—255.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 128- А | 144- Р | 160- а | 176- ░ | 192- └ | 208- ╨ | 224- р | 240- Ё |
| 129- Б | 145- С | 161- б | 177- ▒ | 193- ┴ | 209- ╤ | 225- с | 241- ё |
| 130- В | 146- Т | 162- в | 178- ▓ | 194- ┬ | 210- ╥ | 226- т | 242- Є |
| 131- Г | 147- У | 163- г | 179- │ | 195- ├ | 211- ╙ | 227- у | 243- є |
| 132- Д | 148- Ф | 164- д | 180- ┤ | 196- ─ | 212- ╘ | 228- ф | 244- Ї |
| 133- Е | 149- Х | 165- е | 181- ╡ | 197- ┼ | 213- ╒ | 229- х | 245- ї |
| 134- Ж | 150- Ц | 166- ж | 182- ╢ | 198- ╞ | 214- ╓ | 230- ц | 246- Ў |
| 135- З | 151- Ч | 167- з | 183- ╖ | 199- ╟ | 215- ╫ | 231- ч | 247- ў |
| 136- И | 152- Ш | 168- и | 184- ╕ | 200- ╚ | 216- ╪ | 232- ш | 248- ° |
| 137- Й | 153- Щ | 169- й | 185- ╣ | 201- ╔ | 217- ┘ | 233- щ | 249- ∙ |
| 138- К | 154- Ъ | 170- к | 186- ║ | 202- ╩ | 218- ┌ | 234- ъ | 250- · |
| 139- Л | 155- Ы | 171- л | 187- ╗ | 203- ╦ | 219- █ | 235- ы | 251- √ |
| 140- М | 156- Ь | 172- м | 188- ╝ | 204- ╠ | 220- ▄ | 236- ь | 252- № |
| 141- Н | 157- Э | 173- н | 189- ╜ | 205- ═ | 221- ▌ | 237- э | 253- ¤ |
| 142- О | 158- Ю | 174- о | 190- ╛ | 206- ╬ | 222- ▐ | 238- ю | 254- ■ |
| 143- П | 159- Я | 175- п | 191- ┐ | 207- ╧ | 223- ▀ | 239- я | 255- |
| 144- Р | 160- а | 176- ░ | 192- └ | 208- ╨ | 224- р | 240- Ё | |

# Представление информации в компьютере

## Десятичные, двоичные и шестнадцатеричные числа

В повседневной жизни человек имеет дела с десятичными числами. В *десятичной системе* счисления для представления чисел используются цифры от 0 до 9. Значение числа определяется как сумма произведений цифр числа на их весовые коэффициенты, определяемые местами цифр в числе. Весовой коэффициент самой правой цифры равен единице, цифры перед ней — десяти, затем ста и т. д. Например, число 2703 равно $2 \times 1000 + 7 \times 100 + 0 \times 10 + 3 \times 1$.

Если места цифр (разряды) пронумеровать справа налево и самой правой позиции присвоить номер ноль, то можно заметить, что вес *i*-ого разряда равен *i*-й степени десятки (рис. П1).

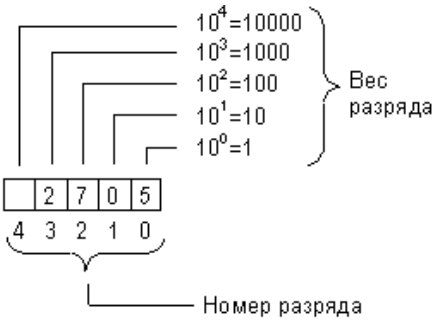
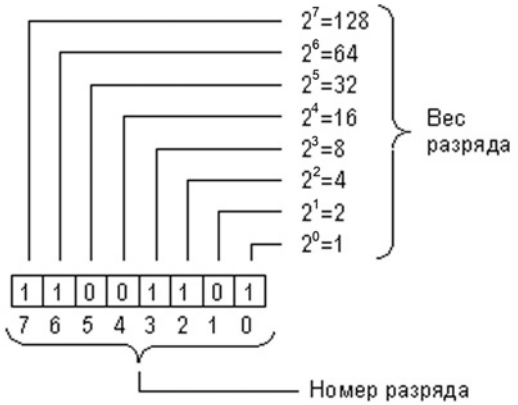



**Рис. П1.** Вес разрядов в десятичной системе счисления

Для внутреннего представления чисел в компьютере используется *двоичная система* счисления. Двоичные числа записываются при помощи двух цифр — нуля и единицы. Как и десятичная, двоичная система — позиционная. Весовой коэффициент разряда *i*-го равен двум в *i*-й степени (рис. П2).

Двоичные числа наиболее точно отражают состояние памяти, регистров процессора и внешних устройств компьютера. Вместе

с тем, работать с двоичными числами не совсем удобно — слишком много цифр приходится записывать. Поэтому была разработана шестнадцатеричная система счисления и записи чисел, позволяющая компактно записывать двоичные числа и обеспечивающая простой способ перевода двоичного числа в шестнадцатеричное и обратно.




Можно задать одно и то же число так:
1×128 + 1×64 +0×32 + 0×16 + 1×8 + 1×4 + 0×2 + 1×1 = 205
или
$(11001101)_2 = (205)_{10}$

**Рис. П2.** Вес разрядов в двоичной системе счисления

В основе шестнадцатеричной системы счисления лежит тот факт, что, используя четыре двоичные цифры, можно записать шестнадцать чисел (максимальное значение четырехразрядного двоичного числа равно пятнадцати).

Шестнадцатеричное число получается из двоичного следующим образом (рис. П3).

Цифры двоичного числа делятся на группы по четыре. Каждой группе ставится в соответствие сначала десятичное число, являющееся десятичным эквивалентом четырехзначного двоичного, затем полученное десятичное число записывается шестнадцатеричной цифрой. В табл. П1 приведены десятичные числа от нуля до 15 и соответствующие им шестнадцатеричные цифры.

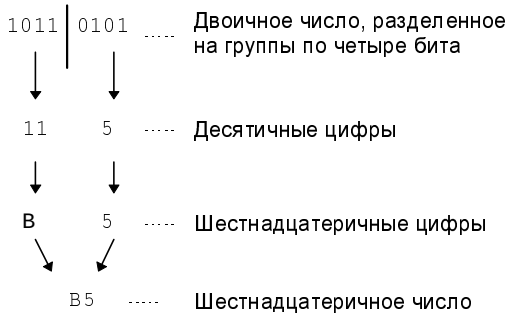



**Рис. П3.** Перевод двоичного числа в шестнадцатеричное

***Таблица П1.*** *Перевод десятичных чисел в шестнадцатеричные*

| Десятичное число | Шестнадцатеричная цифра |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| 10 | A |
| 11 | B |
| 12 | C |
| 13 | D |
| 14 | E |
| 15 | F |

В тексте программы первая цифра шестнадцатеричного числа предваряется символами `0x`. Вот примеры шестнадцатеричных чисел: `0x2A`, `0xFF`, `0x01`.

# Список дополнительной литературы


1. Керниган Б., Ритчи Д., Фьюэр А. Язык программирования Си. — М.: Финансы и статистика, 1985.
2. Уинер Р. Язык Turbo Си: Пер. с англ. — М.: Мир, 1991. — 384 с.: ил.
3. Уэйт М., Прата С., Мартин Д. Язык Си. Руководство для начинающих: Пер. с англ. — М.: Мир, 1988. — 512 с.: ил.
4. Вирт Н. Алгоритмы и структуры данных: Пер. с англ. — М.: Мир, 1989. — 360 с.: ил.
5. Зелковиц М., Шоу А., Гэннон Дж. Принципы разработки программного обеспечения: Пер. с англ. — М.: Мир, 1982. — 386 с.: ил.
6. Мик Б. и др. Практическое руководство по программированию: Пер. с англ. — М.: Радио и связь, 1986. — 168 с.: ил.
7. Фокс Дж. Программное обеспечение и его разработка: Пер. с англ. — М.: Мир, 1985. — 368 с.: ил.
8. Язык компьютера/ Под ред. и с предисл. В. М. Курочкина. Пер. с англ. — М.: Мир, 1989. — 240 с.: ил.

# Предметный указатель


*(продолжение рубрики см. на стр. 280)*

## С



## Т



## Ф

## Ч



## Ш