Никита Культин

C/C++
в задачах и примерах
2-е издание

Санкт-Петербург
«БХВ-Петербург»
2009

УДК 681.3.068+800.92С/С++
ББК 32.973.26–018.1
К90

Культин Н. Б.

К90 С/С++ в задачах и примерах: 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2009. — 368 с.: ил. + CD-ROM

ISBN 978-5-94157-406-3

Книга представляет собой сборник примеров и задач по программированию на языке С/С++, как типовых — ввод-вывод, управление вычислительным процессом, работа с массивами, поиск и сортировка, так и тех, которые обычно не входят в традиционные курсы — работа со строками и файлами, программирование графики, рекурсия. Для большинства задач приведены решения — хорошо документированные исходные тексты программ. Книга содержит также справочник по операторам языка С/С++ и наиболее часто используемым функциям. Может служить задачником для студентов и школьников, изучающих программирование в учебном заведении или самостоятельно. Во втором издании добавлены и обновлены примеры, а также прилагается компакт-диск с исходными текстами программ.

Для начинающих программистов

УДК 681.3.068+800.92С/С++
ББК 32.973.26–018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зав. редакцией	*Григорий Добин*
Редактор	*Леонид Кочин*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Виктория Пиотровская*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 28.07.09.
Формат $60\times90^{1}/_{16}$. Печать офсетная. Усл. печ. л. 23.
Тираж 2000 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-94157-406-3

Оглавление

Предисловие

Чтобы стать программистом, недостаточно прослушать курс лекций или прочитать самоучитель по языку программирования, нужно писать программы, решать конкретные задачи. Но где их взять? В учебниках, как правило, приводятся типовые задачи, в основе которых лежат расчеты по формулам. Они, несомненно, полезны, но не всем интересны.

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 двух частей и приложения.

Первая часть книги содержит примеры и задачи для самостоятельного решения. Они сгруппированы по темам и охватывают практически все разделы базового курса программирования: от объявления переменных и программ с линейной структурой до работы с массивами и файлами.

Вторая часть представляет собой краткий справочник по языку программирования C++. В нем приведено описание основных типов данных, инструкций, реализующих алгоритмические структуры выбора и циклов, наиболее часто используемых функций.

О компиляторе. Если вы только начинаете осваивать язык C++, то перед вами встанет задача выбора среды разработки, компилятора. Действительно, на практике наиболее широко используются Microsoft Visual C++ и Borland C++ Builder. Какой из этих сред

отдать предпочтение, на чем остановить свой выбор? Если в своих программах вы не собираетесь выводить сообщения на русском языке, то подойдет любая из них. Но если вы хотите, чтобы ваши консольные приложения (а именно эти приложения традиционно рассматриваются в качестве примеров при изучении языка программирования С) могли выводить сообщения на русском языке (здесь не следует путать возможность использования букв русского алфавита в тексте программы, например, в символьных константах и комментариях с возможностью отображения русских букв в сообщениях, выводимых на экран во время работы программы), то в этом случае следует выбрать Borland C++ 3.1. 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 справиться с задачей не получается, то можно посмотреть решение и затем еще раз попытаться решить задачу самостоятельно. Перед тем как начать работать на компьютере (набирать текст программы в редакторе кода), рекомендуется "набросать" блок-схему алгоритма решения на бумаге.

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

Оформление решений

Важно, чтобы программа (решенная задача) соответствовала правилам хорошего стиля программирования, была правильно оформлена. Это предполагает:

- использование несущих смысловую нагрузку имен переменных, констант и функций;
- запись инструкций выбора и циклов с применением отступов;
- наличие комментариев.

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

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

ЧАСТЬ I

Примеры и 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 на то, 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 допустимы буквы латинского алфавита и цифры (первым символом должна быть буква).
- Компилятор C++ различает прописные и строчные буквы, поэтому, например, имена `Sum` и `sum` обозначают разные переменные.
- Основными числовыми типами языка C++ являются `int` (целый) и `float` (дробный).
- После инструкции объявления переменной рекомендуется поместить комментарий — указать назначение переменной.

Задачи

1. Объявить переменные, необходимые для вычисления площади прямоугольника.

Задача 1

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

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

Задача 2

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

3. Определить исходные данные и объявить переменные, необходимые для вычисления дохода по вкладу.

Задача 3

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

4. Объявить переменные, необходимые для вычисления площади круга.

5. Объявить переменные, необходимые для вычисления площади кольца.

Задача 5

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

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

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

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
```

8. Объявить 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 служит для изменения значений переменных, в том числе и для вычислений "по формуле".
- В отличие от большинства языков программирования, в C++ инструкция присваивания, изменяющая значение переменной, может быть записана несколькими способами, например вместо `x=x+dx` можно написать `x+=dx`, а вместо `i=i+1` воспользоваться оператором инкремента: `i++`.
- Значение выражения в левой части инструкции присваивания зависит от типа операндов и операции, выполняемой над операндами. Целочисленное сложение и вычитание выполняется без учета переполнения. Например, если переменная `n`, объявленная как `int`, имеет значение `32767`, то в результате выпол-

нения инструкции `n=n+1`, значение переменной `n` будет равно `-32768`.

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

Задачи

9. Записать инструкцию, которая присваивает переменной `x` значение 1,5.

10. Написать инструкцию, которая присваивает переменной `summa` нулевое значение.

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

Задача 11

```
n++;
```

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

Задача 12

```
counter -= 2;
```

13. Написать инструкцию вычисления среднего арифметического переменных `x1` и `x2`.

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

Задача 14

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

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

Задача 15

```
x += dx;
```

16. Записать в виде инструкции присваивания формулу пересчета веса из фунтов в килограммы (один фунт — 405,9 г).

Задача 16

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

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

18. Записать в виде инструкции присваивания формулу вычисления площади прямоугольника.

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

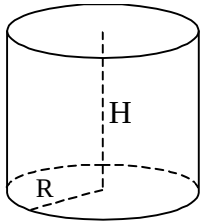
20. Записать в виде инструкции присваивания формулу вычисления площади трапеции: $s=\frac{a+b}{2}\cdot h$, где a и b — длины оснований, h — высота трапеции.

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

Задача 21

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

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



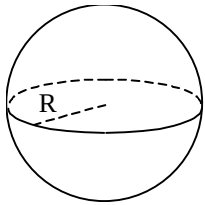
$$s = 2 \cdot \pi \cdot r(h + r)$$
$$v = \pi \cdot r^2 \cdot h$$

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;
```

23. Записать в виде инструкции присваивания формулу вычисления объема параллелепипеда.

24. Объявить необходимые переменные и записать в виде инструкции присваивания формулы вычисления объема и площади поверхности шара.



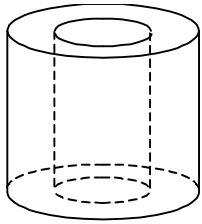
$$v = \frac{3}{4} \cdot \pi \cdot r^3$$
$$s = 4 \cdot \pi \cdot r^2$$

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 math.h
s = 4*M_PI*r*r;
```

25. Записать в виде инструкции присваивания формулу вычисления объема цилиндра.

26. Записать в виде инструкции присваивания формулу вычисления объема полого цилиндра.



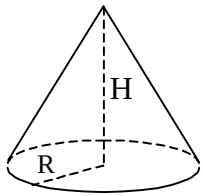
$v=\pi\cdot h\cdot(r_1^2-r_2^2)$

r_1 — радиус цилиндра

r_2 — радиус отверстия

h — высота цилиндра

27. Записать в виде инструкции присваивания формулу вычисления объема конуса.



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

28. Записать в виде инструкции присваивания формулу пересчета температуры из градусов Фаренгейта в градусы Цельсия: $C=\frac{5}{9}(F-32)$.

29. Записать в виде инструкции присваивания формулу для вычисления тока по известным значениям напряжения и сопротивления электрической цепи.

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

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

32. Записать 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. Записать в виде инструкции присваивания формулу пересчета сопротивления электрической цепи из омов в килоомы.

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

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;
```

Вывод

Общие замечания

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

- Функция `printf` обеспечивает вывод на экран монитора сообщений и значений переменных.
- Первый параметр функции `printf` — строка вывода, определяющая выводимый текст и формат отображения значений

переменных, имена которых указаны в качестве остальных параметров функции.

- Формат вывода значений переменных задается при помощи спецификатора преобразования — последовательности символов, начинающейся с `%`.
- При выводе числовых значений наиболее часто используются следующие спецификаторы:
  - `%i` — целое со знаком;
  - `%u` — беззнаковое целое;
  - `%f` — дробное, в виде числа с плавающей точкой;
  - `%n.mf` — дробное в формате с фиксированной точкой, где `n` — общее количество символов (количество цифр целой и дробной частей числа, десятичный разделитель, знак числа); `m` — количество цифр дробной части.
- 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 сообщений на экран можно использовать функцию `puts`.
- Чтобы после окончания работы программы ее окно не было сразу закрыто, в конец программы нужно поместить следующие инструкции:

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

Задачи

35. Написать программу, которая выводит на экран ваше имя и фамилию.

36. Написать программу, которая выводит на экран ваше имя, отчество и фамилию (каждую часть имени с новой строки).

Задача 36

```
#include <stdio.h>
#include <conio.h>
void main()
{
    printf("Иван\nИванович\nИванов\n");
    printf("Для завершения нажмите <Enter>");
    getch(); // ждет нажатия клавиши
}
```

37. Написать программу, которая выводит на экран приведенное далее четверостишие. Между последней строкой стихотворения и именем автора должна быть пустая строка.

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

А. С. Пушкин

Задача 37

```
// Выводит стихотворение
#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
}
```

38. Написать инструкцию вывода значений переменных `a`, `b` и `c` (типа `float`) с пятью цифрами в целой части и тремя — в дробной. Значения должны быть выведены в виде: *a = значение b = = значение c = значение.*

Задача 38

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

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

Задача 39

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

40. Написать инструкцию, которая выводит в одной строке значения переменных `m` и `n` целого типа (`int`).

Задача 40

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

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

Задача 41

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

42. 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, нужно использовать функции `cprintf` и `cputs`. Следует обратить внимание на то, что переход к новой строке в этих функциях задается последовательностью `\r\n`.
- 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

43. 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

Задача 43

```
// Выводит стихотворение
#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();
}
```

44. 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.

Задача 44

```
// 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();
}
```

Ввод

Общие замечания

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

- Для ввода исходных данных с клавиатуры предназначена функция `scanf`.
- Первым параметром функции `scanf` является управляющая строка, которая определяет формат вводимых данных. Остальные параметры задают переменные, значения которых должны быть введены с клавиатуры. Перед именем переменной нужно ставить символ `&` (фактически, в инструкции ввода указывают адреса переменных).
- Управляющая строка представляет собой заключенный в двойные кавычки список спецификаторов:
  - `%i` — для ввода целых чисел со знаком;
  - `%u` — для целых беззнаковых чисел;
  - `%f` — для дробных чисел;
  - `%c` — для ввода символа;
  - `%s` — для ввода строки.
- Отсутствие знака `&` перед именем переменной, указанной в качестве параметра функции `scanf`, является типичной ошибкой начинающих программистов (следует обратить внимание, что компилятор эту ошибку не обнаруживает).

Задачи

45. Написать инструкцию, которая обеспечивает ввод с клавиатуры переменной `kol` целого типа.

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

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

Задача 47

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

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

Задача 48

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

49. Объявить необходимые переменные и написать фрагмент программы вычисления объема цилиндра, обеспечивающий ввод исходных данных.

Задача 49

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

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

50. Объявить необходимые переменные и написать инструкции ввода исходных данных для программы вычисления дохода по вкладу. Предполагается, что процентную ставку программа определяет на основе данных о сумме и сроке вклада.

Задача 50

```
float sum;   // сумма
int period;  // срок (кол-во месяцев)
float rate;  // процентная ставка
float profit; // доход

float value; // сумма в конце срока вклада
printf("Сумма (руб.) -> ");
scanf("%f, &sum);
printf("Срок (мес.) ->");
scanf("%i", &period);
```

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 несложных вычислений по формулам.
- В программах с линейной структурой инструкции выполняются последовательно, одна за другой.

- Алгоритм программы с линейной структурой может быть представлен следующим образом:

Задачи

51. Написать программу вычисления площади прямоугольника. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Площадь прямоугольника: 67.50 кв. см.
```

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 даные:\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();
}
```

52. Написать программу вычисления площади параллелограмма.

53. Написать программу вычисления объема параллелепипеда. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Объем: 337.50 куб. см.
```

54. Написать программу вычисления площади поверхности параллелепипеда. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Площадь поверхности: 90.00 кв. см.
```

Задача 54

```
// 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 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();
}
```

55. Написать программу вычисления объема куба. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Объем куба: 857.38 куб. см.
```

56. Написать программу вычисления объема цилиндра. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Вычисление объема цилиндра
Введите исходные данные:
Радиус основания (см) -> 5
```

```
Высота цилиндра (см) -> 10

Объем цилиндра 1570.80 см. куб.
Для завершения нажмите <Enter>
```

Задача 56

```
// 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();
}
```

57. Написать программу вычисления стоимости покупки, состоящей из нескольких тетрадей и карандашей. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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.
```

Задача 57

```
// 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();
}
```

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

```
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.
```

59. Написать программу вычисления стоимости некоторого количества (по весу), например яблок. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Стоимость покупки: 19.55 руб.
```

60. Написать программу вычисления площади треугольника, если известна длина основания и высота. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Вычисление площади треугольника
Введите исходные данные:
Основание (см) -> 8.5
Высота (см) -> 10

Площадь треугольника 42.50 кв. см.
```

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

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

Задача 61

```
// 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();
}
```

62. Написать программу вычисления сопротивления электрической цепи, состоящей из двух параллельно соединенных сопротивлений. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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
```

Задача 62

```
// 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();
}
```

63. Написать программу вычисления сопротивления электрической цепи, состоящей из двух последовательно соединенных сопротивлений. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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
```

64. Написать программу вычисления силы тока в электрической цепи. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Сила тока: 0.024 Ампер
```

65. Написать программу вычисления расстояния между населенными пунктами, изображенными на карте. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Вычисление расстояния между населенными пунктами
Введите исходные данные:
Масштаб (количество километров в одном сантиметре) -> 120
Расстояние между точками (см) -> 3.5

Расстояние между точками 420 км.
```

66. Написать программу вычисления стоимости поездки на автомобиле. Исходные данные: расстояние (км); количество бензина (в литрах), которое потребляет автомобиль на 100 км пробега; цена одного литра бензина. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Вычисление стоимости поездки на автомобиле
Расстояние (км) -> 67
Расход бензина (литров на 100 км пробега) -> 8.5
Цена литра бензина (руб.) -> 19.20

Поездка обойдется в 109.34 руб.
```

Задача 66

```
// Вычисление стоимости поездки на автомобиле
#include <stdio.h>
#include <conio.h>
void main()
{
    float rast;  // расстояние
```

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

    printf("\nВычисление стоимости поездки на автомобиле\n");
    printf("Расстояние (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 обойдется в %6.2f руб.",summ);

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

67. Написать программу, которая вычисляет скорость, с которой спортсмен пробежал дистанцию. Рекомендуемый вид экрана программы приведен ниже. Данные, введенные пользователем, выделены полужирным.

```
Вычисление скорости бега
Введите длину дистанции (метров) -> 1000
Введите время (минут.секунд)-> 3.25

Дистанция: 1000
Время: 3 мин  25 сек = 205 сек
Скорость:  17.56 км/час
```

Задача 67

```
// 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("Скорость %2.2f км/час\n", v);

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

68. Написать программу вычисления объема цилиндра. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Объем цилиндра 665.23 см куб.
```

69. Написать программу вычисления площади поверхности цилиндра. Ниже приведен рекомендуемый вид экрана про-

граммы (данные, введенные пользователем, выделены полужирным).

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

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

Задача 69

```
// 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();
}
```

70. Написать программу вычисления объема параллелепипеда. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Вычисление объема параллелепипеда
Введите в одной строке длину, ширину
```

```
и высоту параллелепипеда (в сантиметрах).
Числа разделяйте пробелами.
После ввода последнего числа нажмите <Enter>
-> 7.5 2.5 3

Объем параллелепипеда 56.25 см куб.
```

71. Написать программу пересчета расстояния из миль в километры (1 миля равна 1600,94 м). Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Пересчет расстояния из миль в километры
Введите расстояние в милях -> 100


100 миль -это 160.09 км
```

Задача 71

```
// Пересчет расстояния из миль в километры
#include <stdio.h>
#include <conio.h>
void main()
{
   float m; // расстояние в милях
   float k; // расстояние в километрах

   printf("\nПересчет расстояния из миль в километры\n");
   printf("Введите расстояние в милях ->");
   scanf("%f", &m);
   k = m*1.60094;
   printf("%6.2f миль - это %6.2f км\n", v,k);
   printf("\nДля завершения нажмите <Enter>");

   getch();
}
```

72. Написать программу пересчета веса из фунтов в килограммы (1 российский фунт равен 405,9 г). Ниже приведен рекомендуе-

мый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Пересчет веса из фунтов в килограммы
Введите вес в фунтах -> 5

5 фунтов - это 2.05 кг
```

73. Написать программу вычисления дохода по вкладу. Сумма, процентная ставка (% годовых) и срок вклада (в месяцах) задаются во время работы программы. Ниже приведен рекомендуемый вид экрана (данные, введенные пользователем, выделены полужирным).

```
Вычисление дохода по вкладу
Введите исходные данные:
Сумма вклада (руб.) -> 25000
Срок вклада (месяцев) -> 6
Процентная ставка (годовых) -> 10
-------------------------------------
Доход: 1250.00 руб.
Сумма по окончании срока вклада: 262500.00 руб.
```

Задача 73

```
// 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("Сумма 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();
}
```

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

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

Задача 74

```
// 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();
}
```

75. 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. Ниже приведен рекомендуемый вид экрана (данные, введенные пользователем, выделены полужирным).

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

23.6 руб. - это 23 руб. 60 коп.
```

Задача 75

```
// 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();
}
```

76. Написать программу пересчета веса из фунтов в килограммы (1 фунт — 405,9 г). Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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.
```

77. Написать программу, которая вычисляет площадь треугольника, если известны координаты его углов. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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.
```

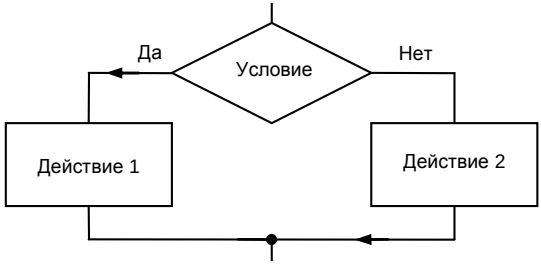
Выбор

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.
- Алгоритм, реализуемый инструкцией `if`, выглядит так:



- Выбор действия (последовательности инструкций) осуществляется в зависимости от значения *условия* — заключенного в скобки выражения, записанного после `if`.
- Инструкция, записанная после `else`, выполняется в том случае, если значение выражения *условие* равно нулю, во всех остальных случаях выполняется инструкция, следующая за условием.
- Если при выполнении (или невыполнении) условия требуется 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

78. Написать программу вычисления сопротивления электрической цепи, состоящей из двух сопротивлений, которые могут быть соединены последовательно или параллельно. Ниже приведен рекомендуемый вид экрана (данные, введенные пользователем, выделены полужирным).

```
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
```

Задача 78

```
// 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();
}
```

79. Написать программу вычисления дохода по вкладу. Исходные данные: сумма и срок вклада. Процентная ставка зависит от суммы. Если сумма меньше 5000 руб., то процентная ставка 10%, если больше, то 13%. Ниже приведен рекомендуемый вид экрана (данные, введенные пользователем, выделены полужирным).

```
Доход
Сумма, руб. -> 10000
Срок вклада, мес. -> 12
----------------------------
Сумма: 10000.00 руб.
Срок вклада: 12 мес.
Процент годовой: 13
Доход: 1300.00 руб.
Сумма в конце срока вклада: 11300.00 руб.
```

Задача 79

```
// Доход по вкладу
#include "stdio.h"
```

```
#include "conio.h"

void main()
{

  float sum;      // сумма вклада
  int period;     // срок

  float percent; // процент, зависит от суммы
  float profit;  // доход

  float total;   // сумма в конце срока вклада
  printf("\nДОХОД\n");

  printf("Сумма, руб. -> ");
  scanf("%f",&sum);

  printf("Срок вклада, мес. -> ");
  scanf("%i",&period);

  if ( sum < 5000)
    percent = 10;
  else
    percent = 13;

  profit = sum * percent/100/12 * period;
  total = sum + profit;

  printf("\nСумма: %3.2f руб.", sum);
  printf("\nСрок вклада: %i мес.", period);
  printf("\nПроцент годовой: %2.2f", percent);
  printf("\nДоход: %3.2f руб.", profit);
  printf("\nСумма в конце срока вклада: %3.2f руб.", total);

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

80. Написать программу вычисления дохода по вкладу. Исходные данные: сумма и срок вклада. Процентная ставка зависит от суммы. Если сумма меньше 5000 руб., то процентная ставка 9%, если больше 5000 руб., но меньше 10 000 руб., то 11%, а если больше 10 000, то 13%. Ниже приведен рекомендуемый вид экрана (данные, введенные пользователем, выделены полужирным).

```
Доход
Сумма, руб. -> 15000
Срок вклада, мес. -> 6
---------------------------
Сумма: 15000.00 руб.
Срок вклада: 6 мес.
Процент годовой: 13.00
Доход: 975.00 руб.
Сумма в конце срока вклада: 15975.00 руб.
```

Задача 80

```
// Доход
#include "stdio.h"
#include "conio.h"

void main()
{

  float sum;      // сумма
  int period;     // срок

  float percent; // период
  float profit;  // доход

  float total;   // сумма в конце срока вклада

  printf("\nДОХОД\n");


  printf("Сумма, руб. -> ");
  scanf("%f",&sum);

  printf("Срок, мес. -> ");
```

```
  scanf("%i",&period);

  if ( sum < 5000)
    percent = 9;
  else
    if ( sum < 10000)
      percent = 11;
    else
      percent = 13;

  profit = sum * percent/100/12 * period;

  total = sum + profit;

  printf("\n---------------------------");
  printf("\nСумма: %3.2f руб.", sum);
  printf("\nСрок: %i", period);
  printf("\nПроцент: %3.2f", percent);
  printf("\nДоход: %3.2f руб.", profit);
  printf("\nСумма в конце срока вклада: %3.2f руб.", total);

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

81. Написать программу вычисления стоимости печати фотографий. Формат фотографий 9×12 или 10×15. Если количество фотографий больше 10, то заказчику предоставляется скидка 5%. Ниже приведен рекомендуемый вид экрана (данные, введенные пользователем, выделены полужирным).

```
Фотографии
Формат (1 - 9x12; 2 - 10x15) -> 2
Количество, шт. -> 15
---------------------------
Цена: 3.20 руб.
Количество: 15 шт.
Сумма: 48.00 руб.
Скидка: 2.40 руб.
К оплате: 45.60 руб.
```

Задача 81

```
// Фото - вычисляет стоимость печати фотографий с учетом
// размера и возможной скидки
#include "stdio.h"
#include "conio.h"
void main()

{
  int format; // формат: 1 - 9x12; 2 - 10x15
  int kol;    // количество

  float cena; // цена за штуку
  float sum;  // сумма

  float discount = 0; // скидка
  float total;    // к оплате

  printf("\nФОТО\n");

  printf("Формат (1 - 9x12; 2 - 10x15) -> ");
  scanf("%i",&format);

  printf("Количество, шт. -> ");
  scanf("%i",&kol);


  if ( format == 1 )
    cena = 2.50;
  else
    cena = 3.20;

  sum = cena * kol;

  if ( kol > 10 )
```

```
  {
    discount = sum * 0.05;
    total = sum - discount;
  }


  printf("\n-------------------------");
  printf("\nЦена: %3.2f руб.", cena);
  printf("\nКоличество: %i", kol);
  printf("\nСумма: %3.2f руб.", sum);

  if ( discount != 0)
  {
    printf("\nСкидка: %3.2f руб.", discount);
    printf("\nК оплате: %3.2f руб.", total);
  }


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

82. Написать программу, которая вычисляет величину тока, потребляемого электроприбором ($I = P/U$, где: I — ток, А; P — мощность, Вт; U — напряжение, В). Программа должна проверять правильность введенных пользователем данных и, если они неверные (делитель равен нулю), выводить сообщение об ошибке. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Ток в электрической цепи
Мощность, Вт -> 60
Напряжение, В -> 0

Ошибка! Напряжение не должно быть равно нулю.
```

Задача 82

```
// Ток в электрической цепи
#include <stdio.h>
```

```
#include <conio.h>
void main()
{
    float P,U,I; // Мощность, напряжение, ток

    printf("\nТок в электрической цепи\n");

    printf("Мощность, Вт -> ");
    scanf("%f", &P);
    printf("Напряжение, В -> ");
    scanf("%f", &U);

    if (U != 0)
    {
       I = P / U;
       printf("Ток в цепи: %5.2f A, I);
    }
    else
       printf("Ошибка! Напряжение не должно быть равно нулю!\n");

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

83. Написать программу вычисления площади кольца. Программа должна проверять правильность исходных данных. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Ошибка! Радиус отверстия больше радиуса кольца
```

Задача 83

```
// 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();
}
```

84. 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 программы приведен ниже. Ошибочные данные, введенные пользователем, выделены полужирным.

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

Задача 84

```
// 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("Число 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();
}
```

85. Написать программу, которая проверяет, является ли год високосным. Ниже приведен рекомендуемый вид экрана программы. Данные, введенные пользователем, выделены полужирным.

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

Задача 85

```
// 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 - не високосный\n", year);
   else
    printf("%i год - високосный\n", year);

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

86. 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 программы (данные, введенные пользователем, выделены полужирным).

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

Задача 86

```
// 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("\nРешение 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();
}
```

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

```
Вычисление стоимости покупки с учетом скидки
Сумма покупки -> 1200
Вам предоставляется скидка 10%
Сумма покупки с учетом скидки: 1080.00 руб.
```

88. 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 при-

веден рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 88

```
// 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 сумму 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();
}
```

89. Написать программу проверки знания даты основания Санкт-Петербурга. В случае неправильного ответа пользователя, программа должна выводить правильную дату. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 89

```
// 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();
}
```

90. Написать программу проверки знания даты начала Второй мировой войны. В случае неправильного ответа пользователя, программа должна выводить правильную дату. Ниже приведен

рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

91. Написать 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 программы (данные, введенные пользователем, выделены полужирным).

```
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
```

Задача 91

```
// 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 правильного 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();
}
```

92. Написать 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 программы (данные, введенные пользователем, выделены полужирным).

```
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
```

93. 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 программы.

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

Задача 93

```
// 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();
}
```

94. 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 программы (данные, введенные пользователем, выделены полужирным).

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

Задача 94

```
// 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
                               // случайных 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();
}
```

95. 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 программы (данные, введенные пользователем, выделены полужирным).

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

96. Написать программу, которая проверяет, является ли введенное пользователем целое число четным. Ниже приведен реко-

мендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 96

```
// 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();
}
```

97. Написать программу, которая проверяет, делится ли на три введенное с клавиатуры целое число. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

98. Написать программу вычисления стоимости разговора по телефону с учетом 20%-ной скидки, предоставляемой по субботам

и воскресеньям. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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.
```

Задача 98

```
// 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 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();
}
```

99. Написать программу, которая вычисляет оптимальный вес человека, сравнивает его с реальным и выдает рекомендацию о необходимости поправиться или похудеть. Оптимальный вес вычисляется по формуле: Рост (см) – 100. Рекомендуемый вид экрана программы приведен ниже (данные, введенные пользователем, выделены полужирным).

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

Задача 99

```
// 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();
}
```

100. Написать программу, которая запрашивает у пользователя номер месяца и затем выводит соответствующее название времени года. Если пользователь введет недопустимое число, программа должна вывести сообщение "Ошибка данных". Ниже приведен рекомендуемый вид экрана программы.

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

Задача 100

```
// 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();
}
```

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

102. Написать программу, которая отображает введенное пользователем число (от 1 до 999) в денежном формате — дописывает слово "рубль" в правильной форме, например, 12 рублей, 21 рубль и т. д.

Задача 102

```
// 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();
}
```

103. 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 т. д.

104. Написать программу, которая вычисляет дату следующего дня. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Введите цифрами сегодняшнюю дату (число месяц год) -> 31 12 2008
Последний день месяца!
С наступающим Новым годом!
Завтра 1.1.2009
```

Задача 104

```
// 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++;
```

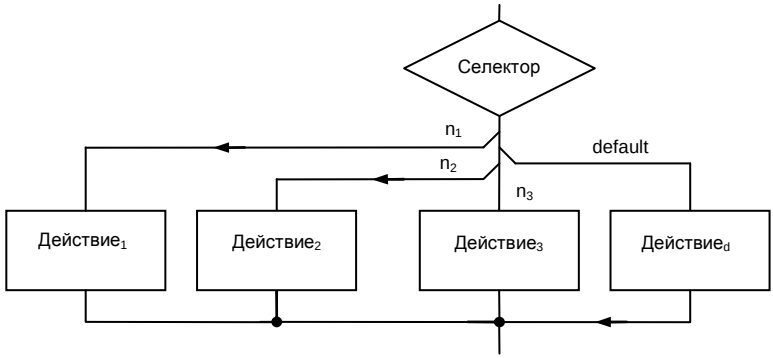
```
    }
    else day++;
    printf("Завтра %i %i %i",day,month,year);

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

Инструкция *switch*

Общие замечания

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

- Инструкция `switch` используется для выбора одного из нескольких возможных вариантов дальнейшего хода программы.
- Алгоритм, реализуемый инструкцией `switch`, выглядит так:



- Выбор действия 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, записанные после слова `default`.
- В качестве переменной-селектора можно использовать переменную целого или символьного типа.

Задачи

105. Написать программу, которая позволяет посчитать цену жалюзи. Исходные данные: размер (ширина и высота, выраженные в сантиметрах) и тип материала (пластик, текстиль, алюминий). Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Жалюзи
Ширина (см) -> 75
Высота (см) -> 150
Материал:
1 - Пластик
2 - Текстиль
3 - Алюминий
Ваш выбор -> 3
-----------------------------
Цена за кв. м: 350.00 руб.
Площадь: 1.13 кв. м.
К оплате: 393.75 руб.
```

Задача 105

```
// Жалюзи - вычисляет цену жалюзи
// в зависимости от размера и материала

#include "stdio.h"
#include "conio.h"
void main()
{
  float w,h; // ширина, высота
  int m;     // материал:
             // 1 - пластик; 2 - текстиль; 3 - алюминий;

  float cena; // цена за 1 кв. м.
  float s;    // площадь
  float sum;  // сумма

  printf("\nЖАЛЮЗИ\n");

  printf("\nШирина (см.) -> ");
```

```
  scanf("%f",&w);

  printf("Высота (см) -> ");
  scanf("%f",&h);

  printf("Материал:\n");
  printf("1 - пластик\n");
  printf("2 - текстиль\n");
  printf("3 - алюминий\n");
  printf("Ваш выбор ->");

  scanf("%i",&m);
  switch ( m )
  {
    case 1: cena = 200; break;
    case 2: cena = 250; break;
    case 3: cena = 350; break;

    default: cena = 0; break;
  }

  if ( cena != 0 )
  {
    s = (w * h) / 10000;
    sum = s * cena;
    printf("\nЦена за кв. м: %3.2f руб.", cena);
    printf("\nПлощадь: %3.2f кв. м", s);
    printf("\nК оплате: %3.2f руб.", sum);
  }
  else
    printf("\nНеправильно указан код материала");

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

106. Написать программу, которая позволяет посчитать стоимость печати фотографий. Исходные данные: размер фотографий (9×12, 10×15 или 18×24) и их количество. Если заказанных фотографий больше 10, заказчику должна предоставляться скидка 10%. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Фото
Размер:
1 - 9x12
2 - 10x15
3 - 18x24
Ваш выбор -> 1
Количество -> 12
-----------------------------
Цена: 3.50 руб.
Количество: 12 шт.
Сумма: 42.00 руб.
Скидка: 4.20 руб.
К оплате: 37.80 руб.
```

Задача 106

```
// Фото - вычисляет цену заказа печати фотографий

#include "stdio.h"
#include "conio.h"
void main()
{
  int k; // количество фотографий
  int f; // формат: 1 - 9x12; 2 - 12x15; 3 - 18x24

  float cena;     // цена за 1 шт.
  float sum;      // сумма
  float discount; // скидка
  float itog;

  printf("\nФОТО\n");

  printf("Формат:\n");
  printf("1 - 9x12\n");
```

```
printf("2 - 12x15\n");
printf("3 - 18x24\n");
printf("Ваш выбор ->");

scanf("%i",&f);
switch ( f )
{
  case 1: cena = 3.50; break;
  case 2: cena = 5.00; break;
  case 3: cena = 8.50; break;

  default: cena = 0; break;
}


printf("Количество (шт.) -> ");
scanf("%i",&k);



if ( cena != 0 )
{
  sum = k * cena;
  printf("\nЦена за шт.: %3.2f руб.", cena);
  printf("\nКоличество: %i шт.", k);
  if ( k > 10 )
  {
    discount = sum * 0.1;
    itog = sum - discount;
    printf("\nСумма: %3.2f руб.",sum);
    printf("\nСкидка: %3.2f руб.", discount);
    printf("\nК оплате: %3.2f руб.",itog);

  }
  else
```

```
      printf("\nК оплате: %3.2f руб.", sum);
  }
  else
    printf("\nНеправильно указан код материала");

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

107. Написать программу, которая позволяет посчитать стоимость заправки автомобиля. Исходные данные: тип топлива (бензин 92, 95, 98 или дизельное топливо) и количество литров. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Бензин
Марка бензина:
1 - 92
2 - 95
3 - 98
4 - DT
Ваш выбор -> 2
Литров -> 25
----------------------------
Цена за литр: 20.30 руб.
Литров: 25

К оплате: 507.50 руб.
```

Задача 107

```
// Бензин - вычисляет стоимость заправки автомобиля

#include "stdio.h"
#include "conio.h"
void main()
{
  int m; // марка бензина: 1 - 92; 2 - 95; 3 - 98; 4 - DT
  float cena; // цена за 1 литр

  float litr; // литров
  float sum;  // сумма
```

```
  printf("\nБЕНЗИН\n");

  printf("Марка бензина:\n");
  printf("1 - 92\n");
  printf("2 - 95\n");
  printf("3 - 98\n");
  printf("3 - DT\n");
  printf("Ваш выбор ->");

  scanf("%i",&m);
  switch ( m )
  {
    case 1: cena = 17.50; break;
    case 2: cena = 20.30; break;
    case 3: cena = 25.40; break;
    case 4: cena = 18.50; break;

    default: cena = 0; break;
  }

  printf("Литров -> ");
  scanf("%f",&litr);

  if ( cena != 0 )
  {
    sum = litr * cena;
    printf("\nЦена за литр: %3.2f руб.", cena);
    printf("\nЛитров: %3.2f", litr);
    printf("\nК оплате: %3.2f руб.", sum);
  }
  else
    printf("\nНеправильно указан код топлива");

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

108. Написать программу, которая запрашивает у пользователя номер дня недели и затем выводит его название. Если введены неправильные данные, программа должна вывести сообщение об ошибке.

Задача 108

```
// 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();
}
```

109. Написать программу, которая вычисляет доход по вкладу. Процентная ставка зависит от срока вклада:

Срок	3 мес.	6 мес.	12 мес.	18 мес.	24 мес.	36 мес.
Процент	9,0%	11,5%	13,5%	15,0%	18,0%	24,0%

Задача 109

```
// Доход по вкладу
#include "stdio.h"
#include "conio.h"

void main()
{

    float value;
    int period;
    float rate;
    float profit;

    printf("value->");
    scanf("%f",&value);

    printf("period->");
    scanf("%i",&period);

    switch ( period )
    {
        case 3:  rate=9.0; break;
        case 6:  rate=11.5; break;
        case 12: rate=13.5; break;
        case 18: rate=15.0; break;
        case 24: rate=18.0; break;
        case 36: rate=24; break;
        default: period = 0;
    }

    if ( period !=0 )
    {
        printf("\nrate: %6.2f", rate );
        profit = value * rate/100/12 * period;
```

```
        printf("\nprofit: %6.2f", profit);
    }
    else
        printf("Not valid period");


    printf("<Enter>");
    getch();
}
```

110. Написать программу, которая вычисляет стоимость междугороднего телефонного разговора (цена одной минуты зависит от расстояния до города, в котором находится абонент). Исходные данные для программы: код города и длительность разговора. Ниже приведены коды некоторых городов, и рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

Город	Код	Цена минуты (руб.)
Владивосток	423	2,2
Москва	495	1,0
Мурманск	815	1,2
Самара	846	1,4

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

Задача 110

```
// Определение стоимости междугороднего
// 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 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\t495");
    puts("Мурманск\t815");
    puts("Самара\t\t846");
    printf("->");
    scanf("%i", &kod);

    printf("Город: ");
    switch (kod)
    {
      case 432: puts("Владивосток");
                cena = 2.2;
                break;
      case 495:  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();
}
```

111. Написать программу, которая по дате определяет соответствующий ей день недели. Для вычисления дня недели воспользуйтесь формулой: $(d + \left[\frac{1}{5}(13m-1)\right] + Y + \left[\frac{Y}{4}\right] + \left[\frac{c}{4}\right] - 2c + 777) \bmod 7$.

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 нужно взять целую часть от заключенного в них значения. Вычисленное по формуле число определяет день недели: 1 — понедельник, 2 — вторник, ..., 6 — суббота, 0 — воскресенье.

Задача 111

```
// 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();
}
```

Циклы

Цикл *for*

Общие замечания

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

- Инструкция `for` используется для организации циклов с фиксированным числом повторений.
- Алгоритм, реализуемый инструкцией `for`, выглядит так:

- Количество повторений цикла определяется начальным значением переменной-счетчика и условием завершения цикла.
- Переменная-счетчик должна быть целого типа и может быть объявлена непосредственно в инструкции цикла.

Задачи

112. Написать программу, которая выводит на экран ваше имя 10 раз.

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 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
----------------
```

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();
}
```

114. Написать программу, которая выводит таблицу квадратов первых десяти целых положительных чисел. Ниже приведен рекомендуемый вид экрана 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
--------------
```

Задача 114

```
// 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();
}
```

115. Написать программу, которая выводит таблицу квадратов первых пяти целых положительных нечетных чисел. Ниже приведен рекомендуемый вид экрана программы.

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

116. Написать программу, которая выводит таблицу скорости (через каждые 0,5 с) свободно падающего тела ($v = g \cdot t$, где $g = 9{,}8$ м/с^2 — ускорение свободного падения). Рекомендуемый вид экрана приведен ниже.

```
------------------------
Время, с Скорость, м/с
------------------------
  0.0          0.00
  0.5          4.90
  1.0          9.80
  1.5         14.70
  2.0         19.60
  2.5         24.50
  3.0         29.40
------------------------
```

117. Написать программу, которая выводит таблицу ежемесячных платежей по кредиту. Исходные данные для расчета: сумма кредита, срок и процентная ставка. Предполагается, что кредит возвращается (выплачивается) ежемесячно равными долями. Проценты начисляются ежемесячно на величину долга. Рекомендуемый вид экрана приведен иже.

```
Сумма (руб.) -> 150000
Срок (мес.) -> 12
Процентная ставка (годовых) -> 14
```

```
-----------------------------------
      Долг     Процент    Платеж
-----------------------------------
 1  150000.00   1750.00   14250.00
 2  137500.00   1604.17   14104.17
 3  125000.00   1458.33   13958.33
 4  112500.00   1312.50   13812.50
 5  100000.00   1166.67   13666.67
 6   87500.00   1020.83   13520.83
 7   75000.00    875.00   13375.00
 8   62500.00    729.17   13229.17
 9   50000.00    583.33   13083.33
10   37500.00    437.50   12937.50
11   25000.00    291.67   12791.67
12   12500.00    145.83   12645.83
-----------------------------------
Всего процентов: 11375.00
```

Задача 117

```
// Платежи по кредиту
#include "stdio.h"
#include "conio.h"

void main()
{
    float value; // сумма кредита
    int period;  // срок
    float rate;  // процентная ставка

    float debt; // долг, на начало текущего месяца
    float interest; // плата за кредит (проценты на долг)
    float paying;   // сумма платежа
    float suminterest; // общая плата за кредит

    printf("Сумма, руб.,-> ");
    scanf("%f",&value);

    printf("Срок, мес.,-> ");
    scanf("%i",&period);

    printf("Процентная ставка, годовых-> ");
```

```
    scanf("%f",&rate);

    debt = value;
    suminterest = 0;
    printf("----------------------\n");
    printf("   Долг Процент Платеж\n");
    printf("----------------------\n");
    for ( int i = 1; i <= period; i++)
    {
        interest = debt * (rate/12/100);
        suminterest += interest;
        paying = value/period + interest;
        printf("%2i  %9.2f  %9.2f  %9.2f\n",
                i,debt,interest, paying);
        debt = debt - value/period;
    }
    printf("----------------------\n");
    printf("\nВсего процентов: %3.2f",suminterest);
    printf("\n Для завершения нажмите <Enter>");
    getch();

}
```

118. Написать программу, которая выводит на экран таблицу соответствия температуры в градусах Цельсия и Фаренгейта ($F° = 5/9 \cdot C° + 32$). Диапазон изменения температуры в градусах Цельсия и шаг должны вводиться во время работы программы. Рекомендуемый вид экрана приведен ниже.

```
t1 -> 0
t2 -> 10
dt-> 1

-----------------
   C        F
-----------------
  0.00     32.00
  1.00     33.80
  2.00     35.60
  3.00     37.40
```

```
  4.00     39.20
  5.00     41.00
  6.00     42.80
  7.00     44.60
  8.00     46.40
  9.00     48.20
 10.00     50.00
-----------------
```

Задача 118

```
// Таблица перевода температуры из градусов Цельсия
// в градусы Фаренгейта
#include "stdio.h"
#include "conio.h"
void main()
{
  float t1,t2,dt; // диапазон и шаг
  float c,f;      // градусы Цельсия и Фаренгейта
  int n;          // число строк в таблице

  printf("t1->");
  scanf("%f",&t1);

  printf("t2->");
  scanf("%f",&t2);

  printf("dt->");
  scanf("%f",&dt);

    n = (t2 - t1)/dt + 1;

  c = t1;

  printf("\n-----------------\n");
  printf("\n  C      F\n");
  printf("\n-----------------\n");
  for (int i=0; i<n; i++)
  {
```

```
    f = (float)9/5 * c + 32;
    printf("%5.2f  %5.2f\n", c,f);
    c = c + dt;
  }
    printf("\n----------------\n");

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

  getch();
}
```

119. Написать программу, которая выводит на экран таблицу перевода длины из дюймов в миллиметры (1 дюйм = 2,54 см). Диапазон длины в дюймах и шаг изменения должны вводиться во время работы программы.

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

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

Задача 120

```
// 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();
}
```

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

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

122. 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 программы (данные, введенные пользователем, выделены полужирным).

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

Задача 122

```
// 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: 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();
}
```

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

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

Задача 123

```
// 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 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();
}
```

124. Написать программу, которая выводит таблицу степеней двойки, от нулевой до десятой. Ниже приведен рекомендуемый вид экрана 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
```

Задача 124

```
// 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();
}
```

125. 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, факториал 8 — 40 320).

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

126. Написать программу, которая вводит с клавиатуры пять дробных чисел и вычисляет их среднее арифметическое. Рекомендуемый вид экрана программы приведен ниже. Данные, введенные пользователем, выделены полужирным.

```
Среднеее арифметическое 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>
```

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

```
Вычисление среднего арифметического последовательности дробных
чисел.
Введите количество чисел последовательности -> 5
Вводите последовательность. 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>
```

128. 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 программы приведен ниже.

```
Обработка последовательности дробных чисел
После ввода каждого числа нажимайте <Enter>
-> 12.3
Введено чисел: 1 Сумма: 12.30 Сред. арифм.: 12.30
-> 15
Введено чисел: 2 Сумма: 27.30 Сред. арифм.: 13.65
-> 10
Введено чисел: 3 Сумма: 37.30 Сред. арифм.: 12.43
-> 5.6
Введено чисел: 4 Сумма: 42.90 Сред. арифм.: 10.73
-> 11.5
Введено чисел: 5 Сумма: 54.40 Сред. арифм.: 10.88

Для завершения нажмите <Enter>
```

Задача 128

```
// Среднее арифметическое дробных чисел, вводимых с клавиату-
ры
#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();
}
```

129. Написать программу, которая вычисляет среднее арифметическое последовательности дробных чисел, вводимых с клавиатуры. После ввода последнего числа, программа должна вывести минимальное и максимальное числа последовательности. Количество чисел последовательности должно задаваться во время работы программы. Рекомендуемый вид экрана

приведен ниже. Данные, введенные пользователем, выделены полужирным.

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

Количество чисел: 5
Среднее арифметическое: 4.00
Минимальное число: 1.5
Максимальное число: 7.8
Для завершения нажмите <Enter>
```

Задача 129

```
// 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("Вводите 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();
}
```

130. 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 программы приведен ниже.

```
Случайные числа
 1
 3
 4
 2
 7
 4
 9
 6
 2
 1
 Сред. арифм.: 3.90
```

131. Написать программу, которая генерирует три последовательности из десяти случайных чисел (в диапазоне от 1 до 10). Для каждой последовательности программа должна вычислить среднее арифметическое. Рекомендуемый вид экрана программы приведен 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>
```

Задача 131

```
// 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;       // summa 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();
}
```

132. Написать программу, которая выводит на экран таблицу стоимости, например, яблок в диапазоне от 100 г до 1 кг с шагом 100 г. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Вес, гр.    Стоимость, 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
```

133. Написать программу, которая выводит таблицу значений функции $y = |x|$. Диапазон изменения аргумента от –4 до 4, шаг приращения аргумента 0,5.

Задача 133

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

#define  LB -4 // нижняя граница диапазона изменения аргумен-
та
#define  HB  4 // верхняя граница диапазона изменения аргу-
мента
#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();
}
```

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

135. Написать программу, которая выводит на экран таблицу умножения, например, на 7. Рекомендуемый вид экрана программы приведен ниже.

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

Задача 135

```
// 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();
}
```

136. Написать программу, которая выводит на экран квадрат Пифагора — таблицу умножения. Рекомендуемый вид экрана программы приведен 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
```

Задача 136

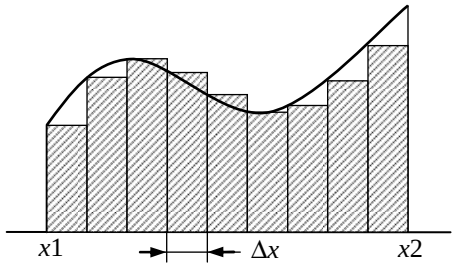
```
// Квадрат Пифагора - таблица умножения
#include <stdio.h>
#include <conio.h>
void main()
{
    int i,j; // номер строки и столбца таблицы

    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();
}
```

137. Написать программу, которая вычисляет частичную сумму ряда: 1 – 1/3 + 1/5 – 1/7 + 1/9… и сравнивает полученное значение с π/4 (при суммировании достаточно большого количества членов этого ряда, величина частичной суммы приближается к π/4).

Задача 137

```
// 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 числа ПИ = %2.6f\n", summ * 4);

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

138. Написать программу приближенного вычисления интеграла функции $f(x) = 5x^2 - x + 2$ методом прямоугольников. Диапазон $x1$, $x2$ и шаг изменения аргумента Δx должны задаваться во время работы программы.

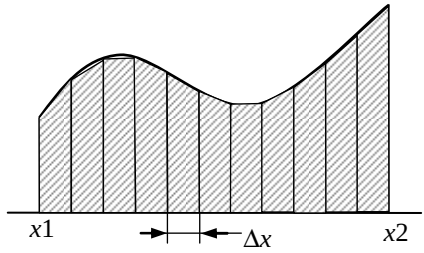


Задача 138

```
// 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();
}
```

139. Написать программу приближенного вычисления интеграла функции $f(x) = 5x^2 - x + 2$ методом трапеций. Границы интервала $x1$, $x2$ и шаг приращения аргумента Δx должны задаваться во время работы программы.



Задача 139

```
// 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 функции в начале интервала
      x += dx;
      y2 = x*x + 2;   // значение функции в конце интервала
      s += (y1 + y2)*dx/2;
   }
   printf("Значение интеграла: %6.3f", s);

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

140. Написать программу, которая выводит на экран изображение шахматной доски. Черные клетки отображать "звездочкой",

белые — пробелом. Рекомендуемый вид экрана программы приведен ниже.

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

141. Написать программу, которая преобразует введенное пользователем десятичное число в двоичное. Рекомендуемый вид экрана программы приведен 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>
```

Задача 141

```
//   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 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();
}
```

Факультатив

142. 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 — "неудовлетворительно". Ниже приведен рекомендуемый вид экрана программы. Ответы пользователя выделены полужирным.

```
*** 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.
```

Задача 142

```
// 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();
}
```

143. 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 программы. Ответы пользователя выделены полужирным.

```
Проверка умения складывать и 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
```

Задача 143

```
// 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();
}
```

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

Задача 144

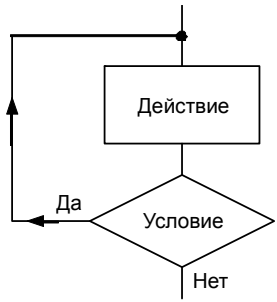
```
// Электронные часы
#include <stdio.h>
#include <conio.h>
#include "dos.h"    // для доступа к delay
void main()
{
    int min,sec; // минуты, секунды
    clrscr();      // очистить экран
    _setcursortype(_NOCURSOR);  // убрать курсор
    printf("Чтобы остановить таймер, нажмите любую клавишу");
    for (min = 0; min <= 2; min++)
   {
      for (sec = 0; sec <= 59; sec++)
      {
        delay(1000);     // задержка 1000 ms
        gotoxy(1,3);     // курсор в 1-ю колонку 1-й строки
        printf("%i:%2i", min, sec);
        if (kbhit()) break;
      }
      if (kbhit()) break;
    }
    _setcursortype(_NORMALCURSOR);
    getch(); // клавиша, остановившая часы
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

Цикл *do ... while*

Общие замечания

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

- Цикл `do while` — это цикл с постусловием, т. е. инструкции тела цикла (между `do` и `while`) будут выполнены хотя бы один раз.
- Алгоритм, реализуемый инструкцией `do while`, выглядит так:



- После слова `while` записывается условие повторного выполнения инструкций цикла.
- Число повторений инструкций цикла `do while` определяется ходом выполнения программы.
- Для завершения цикла `do while` в теле цикла обязательно должны быть инструкции, выполнение которых влияет на условие завершения цикла.
- Цикл `do while`, как правило, используется для организации приближенных вычислений, в задачах поиска и обработки данных, вводимых с клавиатуры или из файла.

Задачи

145. Написать программу, которая выводит на экран таблицу значений функции $y = -2{,}4x^2 + 5x - 3$. Диапазон и шаг изменения

аргумента должны задаваться во время работы программы. Ниже приведен рекомендуемый вид экрана программы.

```
x1 -> -2
x2 -> 2
dx -> 0.5

---------------
   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
----------------
```

Задача 145

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

void main()
{
    float x,y;  // аргумент и значение функции

    float x1,x2,dx; // диапазон и шаг изменения аргумента


    printf("x1 -> ");
    scanf("%f",&x1);
    printf("x2 -> ");
    scanf("%f",&x2);
    printf("dx -> ");
    scanf("%f",&dx);

    x = x1;
```

```
    printf("------------------\n");
    printf("   x    |   y\n");
    printf("------------------\n");
    do
       {
         y = -2.4*x*x+5*x-3;
         printf("%6.2f  | %6.2f\n" ,x ,y);
         x += dx;
        }
    while ( x <= x2 );

    printf("------------------\n");

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

146. Написать программу, которая выводит на экран таблицу соответствия температуры в градусах Цельсия и Фаренгейта ($F°= 5/9 \cdot C° + 32$). Диапазон изменения температуры в градусах Цельсия и шаг должны вводиться во время работы программы. Рекомендуемый вид экрана приведен ниже.

```
t1 -> 0
t2 -> 10
dt-> 1

-----------------
   C        F
-----------------
  0.00     32.00
  1.00     33.80
  2.00     35.60
  3.00     37.40
  4.00     39.20
  5.00     41.00
  6.00     42.80
  7.00     44.60
  8.00     46.40
  9.00     48.20
 10.00     50.00
-----------------
```

Задача 146

```
// Таблица перевода температуры из градусов Цельсия
// в градусы Фаренгейта
#include "stdio.h"
#include "conio.h"
void main()
{
  float t1,t2,dt;
  float c,f;

  printf("\nПеревод температуры из градусов Цельсия");
  printf("\nв градусы Фаренгейта\n");

  printf("t1->");
  scanf("%f",&t1);

  printf("t2->");
  scanf("%f",&t2);

  printf("dt->");
  scanf("%f",&dt);

  printf("\n----------------------\n");
  printf("\n  C      F\n");
  printf("\n----------------------\n");

  c = t1;
  do
  {
    f = (float)9/5 * c + 32;
    printf("%5.2f  %5.2f\n", c,f);
    c = c + dt;
  }
  while ( c <= t2);
```

```
  printf("\n----------------------\n");

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

  getch();
}
```

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

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

Задача 147

```
// 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();
}
```

148. Написать программу, которая определяет максимальное число из введенной с клавиатуры последовательности положительных чисел (длина последовательности не ограничена). Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 148

```
// 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();
}
```

149. Написать программу, которая определяет минимальное число во введенной с клавиатуры последовательности положительных чисел (длина последовательности не ограничена). Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 149

```
// Определение минимального числа
// в 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: %i\n", min);

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

150. Написать программу, которая проверяет, является ли введенное пользователем целое число простым (простым называется число, которое делится только на единицу и на само себя). Рекомендуемый вид экрана программы приведен ниже. Данные, введенные пользователем, выделены полужирным.

```
Введите целое число и нажмите <Enter>
-> 17
17 - простое число.
```

Задача 150

```
// 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();
}
```

151. Написать программу приближенного вычисления интеграла методом прямоугольников. Интервал и точность вычисления должны задаваться во время работы программы, функция — в программе. После каждого цикла вычислений программа должна выводить вычисленное значение интеграла.

Задача 151

```
// Вычисление интеграла методом прямоугольников
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
```

```
float x1,x2; // границы отрезка
float e;     // точность вычисления

float dx;    // приращение аргумента
int n;       // количество интервалов
float x;     // аргумент
float y;     // значение функции в начале интервала

float st;    // значение интеграла на текущем шаге
float sp;    // значение интеграла на текущем шаге

int i;

printf("\Приближенное вычисление интеграла\n");
printf("Нижняя граница интервала -> ");
scanf("%f", &x1);
printf("Верхняя граница интервала -> ");
scanf("%f", &x2);

printf("Требуемая точность вычисления -> ");
scanf("%f", &e);

dx = 0.5;
st = 0;

do
{
   sp = st;
   dx = dx / 2;
   n = (x2 - x1) / dx;
   x = x1;
   st = 0;
   for (i = 0; i <= n; i++)
   {
     y = x + 2; // значение функции в начале интервала
```

```
        st = st + (y * dx);
        x += dx;
      }
      printf("Значение интеграла: %6.3f\n", st);
   }
   while ( abs(sp - st) > e);

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

152. Написать программу, которая "задумывает" число в диапазоне от 1 до 10 и предлагает пользователю угадать его за пять попыток. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 152

```
// 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 пять попыток");
   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();
}
```

Факультатив

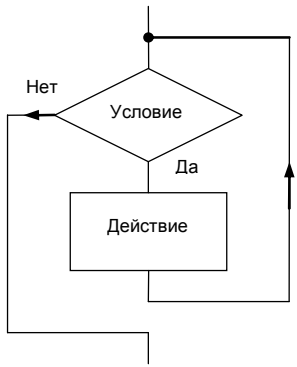
153. Написать программу приближенного вычисления интеграла методом трапеций. Интервал и точность вычисления должны задаваться во время работы программы. После каждого цикла вычислений программа должна выводить вычисленное значение интеграла, число интервалов и шаг изменения аргумента.

Цикл *while*

Общие замечания

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

- `While` — это цикл с предусловием, т. е. возможна ситуация, при которой инструкции тела цикла не будут выполнены ни разу.
- Алгоритм, реализуемый инструкцией `do while`, выглядит так:



- Инструкции цикла `while` выполняются до тех пор, пока условие истинно (значение выражения `Условие` не равно нулю).
- Для того чтобы цикл `while` завершился, в его теле должны быть инструкции, влияющие на значения переменных, входящих в условие выполнения цикла.
- Цикл `while` обычно используется для организации приближенных вычислений, в задачах поиска и обработки данных, вводимых с клавиатуры или из файла.

Задачи

154. Написать программу, которая выводит на экран таблицу значения функции $y = 2x^2 - 5x - 8$ в диапазоне от –4 до 4. Шаг изменения аргумента равен 0,5.

Задача 154

```
// 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();
}
```

155. Написать программу, которая вычисляет число "Пи" с заданной пользователем точностью. Для вычисления числа π воспользуйтесь тем, что значение частичной суммы ряда 1 – 1/3 + 1/5 – 1/7 + 1/9 ..., при суммировании достаточно большого количества членов, приближается к π/4. Рекомендуемый

вид экрана программы приведен ниже (данные, введенные пользователем, выделены полужирным).

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

Задача 155

```
// 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();
}
```

156. Написать программу, которая вычисляет наибольший общий делитель двух целых чисел.

Задача 156

```
// 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();
}
```

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 его элементов.
- Элементы массива нумеруются с нуля.
- Доступ к элементу массива осуществляется путем указания индекса (номера) элемента. В качестве индекса можно использовать константу, переменную или выражение целого типа. Индекс может меняться от 0 до $n - 1$, где n — число 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 работе с массивами является обращение к несуществующему элементу, т. е. выход индекса за допустимые пределы значений.

Задачи

157. Написать программу, которая записывает введенные с клавиатуры данные в одномерный массив целого типа, состоящий из семи элементов. Перед вводом каждого элемента должна выводиться подсказка с номером элемента. После ввода последнего

элемента программа должна вывести введенный массив и вычислить среднее арифметическое его элементов.

```
Ввод массива целых чисел
После ввода каждого числа нажмите <Enter>
a[0] -> 10
a[1] -> 16
a[2] -> 14
a[3] -> 5
a[4] -> 10
a[5] -> 22
a[6] -> 22

Массив: 10   16   14   5   10   22   22
Среднее арифметическое: 14.00
```

Задача 157

```
// Ввод, вывод и обработка массива
#include <stdio.h>
#include <conio.h>
void main()
{
int a[7]; // массив
   int sum;  // сумма элементов массива
   float m;  // среднее арифметическое

   int j;

   printf("\nВвод массива целых чисел");
   printf("После ввода каждого числа нажмите <Enter>\n");

   // ввод массива
   for ( j = 0; j < 7; j++)
   {
           printf("na[%i] -> ", j);
           scanf("%i",&a[j]);
   }

   // вывод массива
   printf("\nМассив: \n");
```

```
    for ( j = 0; j < 7; j++)
    {
            printf("%i   ", a[j]);
    }

    sum = 0;
    // вычислить сумму элементов
    for ( j = 0; j < 7; j++)
    {
             sum = sum + a[j];
    }

    m = sum / 7;

    printf("Среднее арифметическое: %f", m);

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

158. Написать программу, которая вводит с клавиатуры данные в одномерный массив дробного типа, состоящий из пяти элементов, 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
------------------------
Ненулевых элементов: 3
```

Задача 158

```
// 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();
}
```

159. Написать программу, которая выводит минимальный элемент введенного с клавиатуры массива целых чисел. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 159

```
// Поиск минимального элемента массива
#include <stdio.h>
#include <conio.h>
```

```
#define НВ  5    // размер массива
void main()
{
    int a[НВ];  // массив
    int min;    // номер минимального элемента
    int i;      // индекс массива

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

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

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

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

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

Задача 160

```
// Поиск минимального элемента массива
// (доступ к элеменам при помощи указателя)
#include <stdio.h>
#include <conio.h>
```

```
#define НВ  5    // размер массива
void main()
{
    int a[НВ];   // 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", НВ);
    printf("-> ");
    p = a;
    for (i = 1; i <= НВ; 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 <= НВ; i++)
    {
        if (*p < *min) min = p;
        p++;  // к следующему элементу
    }
    printf("Минимальный элемент массива: %i\n", *min);

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

161. Написать программу, которая вычисляет среднее арифметическое ненулевых элементов введенного с клавиатуры массива целых чисел. Ниже приведен рекомендуемый вид экрана

программы (данные, введенные пользователем, выделены полужирным).

```
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
```

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

```
Среднее арифметическое без учета min 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:
```

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

```
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.
```

Задача 163

```
// 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();
}
```

164. Написать программу, которая проверяет, находится ли введенное с клавиатуры число в массиве. Массив должен вводиться в процессе работы программы.

Задача 164

```
// Поиск в массиве методом перебора элементов
#include <stdio.h>
#include <conio.h>
#define NB 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();
}
```

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

Задача 165

```
// Проверяет, отсортирован ли массив по возрастанию
#include <stdio.h>
#include <conio.h>
#define НВ 5
void main()
{
    int a[НВ];   // 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 ", НВ);
    printf("в одной строке) и нажмите <Enter>\n");
    for (k = 0; k < НВ; 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 ");
    printf("упорядочены по возрастанию\n");

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

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

Задача 166

```
// 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();
}
```

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

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

Задача 168

```
// 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();
}
```

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

Задача 169

```
// 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();
}
```

170. Написать программу, которая объединяет два упорядоченных по возрастанию массива в один, который также должен быть упорядочен по возрастанию. Рекомендуемый вид экрана программы приведен ниже, данные, введенные пользователем, выделены полужирным.

```
Объединение двух массивов, упорядоченных по возрастанию
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 нажмите <Enter>
```

Задача 170

```
// 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 двухупорядоченных ");
    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();
}
```

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

Задача 171

```
// 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 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();
}
```

172. Написать программу, которая определяет количество учеников в классе, чей рост превышает средний. Рекомендуемый вид экрана программы приведен ниже. Введенные пользователем данные выделены полужирным.

```
*** Анализ роста учеников ***
Введите рост (см) учеников
Для завершения введите 0 и нажмите <Enter>
-> 175
-> 170
-> 180
-> 168
-> 170
-> 0
------------------------------
Средний рост: 172.6 см
У 2 человек рост превышает средний
```

Задача 172

```
// 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("Введите 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();
}
```

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

Задача 173

```
// 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();
}
```

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

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

```
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 нажмите <Enter>
```

Задача 175

```
// 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();
}
```

176. Написать программу, которая вводит по строкам с клавиатуры двумерный массив дробного типа (3×5 — три строки по пять элементов) и вычисляет среднее арифметическое элементов строк. Рекомендуемый вид экрана приведен ниже.

```
Ввод и обработка массива (3x5)
Введите массив по строкам
Строка 1 -> 12 15 10 22 3
Строка 2 -> 10 10 3 5 12
Строка 3 -> 11 17 10 9 7
---------------------------------------
Массив:
   12   15   10   22    3
   10   10    3    5   12
   11   17   10    9    7

Среднее арифметическое:
Строка 1: 12.40
Строка 2:  8.00
Строка 3: 10.80

Для завершения нажмите <Enter>
```

Задача 176

```
#include <stdio.h>
#include <conio.h>
void main()
{
int a[3][5]; // массив 3x5 - 3 строки по 5 элементов
   int r;       // номер строки
   int c;       // номер столбца

   int sum;     // сумма элементов строки
   float m;     // среднее арифметическое

   int k;

   printf("\nВвод по строкам\n");
   printf("Введите элементы строки (5 чисел) и нажмите <Enter>\n");

   // ввод массива
   k =1;
```

```
for ( r = 0; r < 3; r++)
{
    printf("Строка %i -> ", k);
    for ( c = 0; c < 5; c ++)
    {
           scanf("%i",&a[r][c]);
    }
    k++;
}

printf("\n--------------------------\n");
// вывод массива
printf("\nМассив\n");
for ( r = 0; r < 3; r++)
{
    for ( c = 0; c < 5; c ++)
    {
           printf("%5i",a[r][c]);
        }
    printf("\n");
}

// обработка массива
printf("Среднее арифметическое:\n");
k = 1;
for ( r = 0; r < 3; r++)
{
    // вычислить среднее арифметическое
    // элементов строки
    sum = 0;
    for ( c = 0; c < 5; c ++)
    {
           sum = sum + a[r][c];
    }
    m = (float) sum / 5;
```

```
        printf("Строка %i: %5.2f\n", k, m);
        k++;
    }

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

177. Написать программу, которая определяет номер строки двумерного массива, сумма элементов которой максимальна.

Задача 177

```
// 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();
}
```

Факультатив

178. 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

Задача 178

```
// 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();
}
```

179. Написать программу, которая вычисляет доход по вкладу. Процентная ставка по вкладу вычисляется на основе данных, приведенных в таблице.

Сумма, тыс. руб.	Срок вклада и процентная ставка					
	3 мес.	6 мес.	12 мес.	18 мес.	24 мес.	36 мес.
до 50	15,0%	16,5%	18,0%	19,5%	21,0%	22,0%
до 250	16,5%	18,0%	19,5%	21,0%	22,5%	24,0%
свыше 250	18,5%	19,5%	21,0%	22,5%	24,0%	27,0%

Задача 179

```
// Доход по вкладу

#include "stdio.h"
#include "conio.h"

void main()
{

  // процентная ставка
  float rate[3][6] =
     {
        15.0,16.5,18.0,19.5,21.0,24.0,
        16.5,18.0,19.5,21.0,22.5,24.0,
```

```
      18.0,19.5,21.0,22.5,24.0,27.0
   };

float value;  // сумма
int period;   // срок (месяцев)
float profit; // доход

int r,c;

// вывести таблицу
for (int r=0; r<3 ;r++)
{
  for(int c=0; c<6; c++)
    printf("%8.2f",rate[r][c]);
  printf("\n");
}

printf("Сумма, руб.-> ");
scanf("%f",&value);

printf("Срок, мес. -> );
scanf("%i",&period);

if ( value <= 10000)
  r = 0;
else
  if (value <= 300000 )
    r = 1;
  else
     r = 3;

switch ( period )
{
  case 3:  c=0; break;
  case 6:  c=1; break;
  case 12: c=2; break;
```

```
    case 18: c=3; break;
    case 24: c=4; break;
    case 36: c=6; break;
    default: period = 0;
  }

  if ( period !=0 )
  {
    printf("\Процентная ставка: %5.2f, rate[r][c]);
    profit = value * rate[r][c]/100/12 * period;
    printf("\Доход: %6.2f руб.", profit);
  }
  else
    printf("Неправильно указан срок");


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

180. Написать программу, которая обрабатывает результаты спортивных соревнований, — выстраивает команды в соответствии с количеством набранных очков, вычисляемым по формуле $K = 7N_G + 6N_S + 5N_B$, где N_G, N_S и N_B — количество золотых, серебряных и бронзовых медалей. Рекомендуемый вид экрана приведен ниже (данные, введенные пользователем, выделены полужирным).

```
Результат соревнований
Австрия-> 0 1 1
Германия-> 1 0 2
Норвегия-> 4 2 1
Россия-> 2 3 2
Финляндия-> 1 2 2
-------------------------------------
  Команда  Зол. Сер. Бр. Всего Очков
1 Норвегия  4    2    1    7     45
2 Россия    2    3    2    7     42
3 Финляндия 1    2    2    5     29
4 Германия  1    0    2    3     17
5 Австрия   0    1    1    2     11
```

Задача 180

```
// Результат соревнований
#include "stdio.h"
#include "conio.h"
#include "string.h"

#define NC 5      //число команд

void main()
{
char *team[] = {
                  "Австрия","Германия","Норвегия",
                   "Россия","Финляндия"
                };

   // таблица результатов
   int result[NC+1][6];
   // result[i][0] - золотых
   // result[i][1] - серебряных
   // result[i][2] - бронзовых
   // result[i][3] - всего
   // result[i][4] - очков
   // result[i][5] - номер команды
   // NC+1 -ая строка используется как буфер
   // при сортировке таблицы

   int i,j;
   int max;     // номер строки таблицы, в которой
                // количество очков максимально

   printf("Введите в одной строке");
   printf("количество золотых,\n");
   printf("серебряных и бронзовых медалей\n");

   // ввод исходных данных
   for (i = 0; i < NC; i++)
```

```
  {
     printf("%s ->", team[i]);
     scanf("%i%i%i", &result[i][0],   // золотых
                     &result[i][1],   // серебряных
                     &result[i][2]);  // бронзовых
      result[i][5] = i; // номер команды
  }

  // 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 строк (методом простого выбора) в
// соответствие с количеством очков
for (i = 0; i < NC+1; i++)
{
    // в части таблицы начиная со строки i
    // найти j-ю 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;
    }
    // поменяем местами i-ю строку со строкой
    // с номером max
    // в качестве буфера используем последнюю
    // строку таблицы
```

```
      for (j = 0; j < 6; j++)
           result[NC][j] = result[i][j];
      for (j = 0; j < 6; j++)
           result[i][j] = result[max][j];
      for (j = 0; j < 6; j++)
           result[max][j] = result[NC][j];
    }

    // здесь таблица упорядочена
    printf("%3s%12s%8s%8s%8s%8s%8s",
           "","Команда",
           "Золото","Серебро","Бронза",
           "Всего","Очков");
    for (i = 0; i < NC; i++)
    {
        printf("\n%12s", team[ result[i][5]   ]);
        for (j = 0; j < 5; j++)
            printf("%8i", result[i][j]);
    }

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

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 инициализировать 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 `gets` — всю строку, в том числе и соответствующий клавише <Enter> символ "" `\n`.

Задачи

181. Написать программу, которая запрашивает имя пользователя и здоровается с ним. Рекомендуемый вид экрана программы приведен ниже (данные, введенные пользователем, выделены полужирным).

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

Задача 181

```
// Приветствие
#include <stdio.h>
#include <conio.h>
void main()
{
    char name[15]; // имя
    char fam[20];  // фамилия

    printf("Как Вас зовут?\n");
    printf("Ведите свое имя и фамилию, 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();
}
```

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

Задача 182

```
// 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 <Enter>
    {  putch(ch);
       name[i++] = ch;
    }
    name[i] = '\0';
    printf("\nЗдравствуйте, %s!\n", name);

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

183. Написать программу, которая вычисляет длину введенной с клавиатуры строки.

Задача 183

```
// 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();
}
```

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

Задача 184

```
// 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();
}
```

185. Написать программу, которая выводит код введенного пользователем символа. Программа должна завершать работу в результате ввода, например, точки. Рекомендуемый вид экрана программы приведен ниже (данные, введенные пользователем, выделены полужирным).

```
Введите символ и нажмите <Enter>.
Для завершения введите точку.
->1
Символ: 1 Код: 49
->2
Символ: 2 Код: 50
->ы
Символ: ы Код:235
->.
```

Задача 185

```
// 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();
}
```

186. 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 т. д.

Задача 186

```
// 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();
}
```

187. Написать программу, которая во введенной с клавиатуры строке преобразует строчные буквы русского алфавита в прописные (учтите, что стандартная функция `upcase` с символами русского алфавита не работает). Рекомендуемый вид экрана программы приведен ниже (данные, введенные пользователем, выделены полужирным).

```
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
```

Задача 187

```
// 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] >= 'а' && st[i] <= 'п'))
        st[i] -= 32;
      else if (st[i] >= 'р' && st[i] <= 'я')
         st[i] -= 80;
      i++;
    }
    printf("\n%s\n", st);

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

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

Задача 188

```
// 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();
}
```

189. Написать программу, которая из введенного в одной строке полного имени человека выделяет имя, отчество и фамилию. Рекомендуемый вид экрана программы приведен ниже (данные, введенные пользователем, выделены полужирным).

```
Введите в одной строке имя, отчество и фамилию
-> Иван Иванович Иванов
Имя: Иван
Отчество: Иванович
Фамилия: Иванов
Для завершения нажмите <Enter>
```

Задача 189

```
// Разбивает строку на подстроки
#include "stdafx.h"
#include "string.h"
#include "stdio.h"
```

```
#include "conio.h"

void main()
{
    char full[80]; // исходная строка

    char first[80]; // имя
    char mid[80];   // отчество
    char last[80];  // фамилия

    char* p1; // указатель на символ исходной строки
    char* p2; // указатель на символ формируемой строки

    printf("Полное имя (имя отчество фамилия) ->");
    gets(full);

    p1 = full;

    // извлечь имя
    p2 = first;
    *p2 = '\0';
    while ( (*p1 != ' ') && ( *p1 != NULL ) )
    {
        *p2 = *p1;
        p1++;
        p2++;
    }
    *p2 = '\0';

    // извлечь отчество
    p2 = mid;
    *p2 = '\0';
    if ( *p1 == ' ')
    {
        p1++;
```

```
    while ( (*p1 != ' ') && ( *p1 != NULL ) )
    {
        *p2 = *p1;
        p1++;
        p2++;
    }
    *p2 = '\0';
}

// извлечь фамилию
p2 = last;
*p2 = '\0';
if ( *p1 == ' ')
{
    p1++;
    while ( (*p1 != ' ') && ( *p1 != NULL ) )
    {
        *p2 = *p1;
        p1++;
        p2++;
    }
    *p2 = '\0';
}

// Проверить, сколько подстрок удалось извлечь
// из строки. Если две, то вторая подстрока
// это фамилия, а не отчество

if ( *last == '\0' )
{
    strcpy(last,mid);
    *mid = '\0';
}

printf("\nИмя:%s",first);
```

```
    printf("\nОтчество:%s",mid);
    printf("\nФамилия:%s",last);

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

190. Написать программу, которая проверяет, является ли введенная с клавиатуры строка целым числом. Рекомендуемый вид экрана программы приведен ниже (данные, введенные пользователем, выделены полужирным).

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

Задача 190

```
// 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();
}
```

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

192. Написать программу, которая проверяет, является ли введенная с клавиатуры строка шестнадцатеричным числом.

Задача 192

```
// 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();
}
```

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

Задача 193

```
// 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();
}
```

194. Написать программу, которая преобразует введенное с клавиатуры восьмиразрядное двоичное число в десятичное. Рекомендуемый вид экрана программы приведен ниже (введенные пользователем данные выделены полужирным).

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

Задача 194

```
// 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();
}
```

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

Задача 195

```
// 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 сим-
вол
    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();
}
```

196. Написать программу, которая преобразует введенное пользователем десятичное число в число в указанной системе счисления (от 2-х до 10). Рекомендуемый вид экрана программы приведен ниже.

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

Задача 196

```
// 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 сист. счисл.
    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();
}
```

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

Задача 197

```
// 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
сист. счисл.
    char st[5];     // Представление числа в заданной сист.
счисл.
    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(" соответствует 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();
}
```

198. 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 программы, данные, введенные пользователем, выделены полужирным.

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

Задача 198

```
// 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();
}
```

Факультатив

199. Написать программу, реализующую игру "Угадай число". Правила игры следующие. Играют двое. Первый игрок задумывает число, второй должен угадать число, задуманное первым. 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 сделано предположение, что задуманное число 523, то угаданы две цифры (5 и 2), но только одна из них (2) занимает верную позицию.

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

```
Игра "Угадай число"
Компьютер задумал трехзначное число. Вы должны его угадать.
После ввода числа, вы узнаете, сколько цифр
угадано и сколько из них находятся на своих местах.
После ввода числа нажимайте <Enter>

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

Поздравляю! Вы угадали число, задуманное компьютером!
Для завершения нажмите <Enter>
```

Задача 199

```
// 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-я 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("\nИгра Угадай число");
   printf("\nКомпьютер задумал трехзначное \n");
   printf("число. Вы должны его угадать.\n");
   printf("После ввода числа вы узнаете, ");
   printf("сообщено, сколько цифр угадано, и \n");
   printf("сколько из них находятся на своих ");
   printf("местах.");
   printf("После ввода числа нажимайте <Enter>\n");

   srand((unsigned)time(&t) ); // инициализация ГСЧ

   // компьютер "задумывает" число
   for (i = 0; i < N; i++)
       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);

      for (i = 0; i < N; i++)
            a[i] = 0;

      // проверим, сколько цифр угадано
      ugad = 0;

      // каждую цифру игрока
      //сравним с цифрами компьютера
      for (i = 0; i < N; i++)
         for (j = 0; j < N; j++)
         {
            if ((igrok[i] == comp[j]) && !a[j])
            {
              ugad++;
              a[j] = 1; // запретим сравнивать
                        // эту цифру компьютера с
                        // 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("\nПоздравляю! Вы угадали число,\n");
    printf("задуманное компьютером.");

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

200. Написать 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	.–.–

Задача 200

```
// Телеграф – формирует сообщение 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 прописные русские буквы)");
    printf("->");
    gets(mes);
    for (i = 0; i < strlen(mes); i++)
```

```
    {
        if (mes[i] >= 'A' && mes[i] <='Я')
        {
        // определим код очередной буквы (функция 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();
}
```

Функции

Общие замечания

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

- Передавать данные в функцию следует только с помощью параметров. Глобальные переменные, т. е. переменные, объявленные вне функции, использовать не рекомендуется.
- Тип каждого фактического параметра (константы или переменной) в инструкции вызова функции должен соответствовать типу соответствующего формального параметра, указанного в инструкции объявления функции.
- Если параметр функции служит для возврата результата, то в объявлении функции этот параметр должен быть ссылкой, а в инструкции вызова функции в качестве фактического параметра должен использоваться адрес переменной.

Задачи

201. Написать функцию пересчета температуры из градусов Фаренгейта в градусы Цельсия ($C° = 5/9 \cdot (F° - 32)$) и программу, использующую эту функцию, которая выводит на экран таблицу соответствия температур в шкалах Фаренгейта и Цельсия.

Задача 201

```
// Функция Fahr2Cels пересчитывает температуру
// из градусов Фаренгейта в градусы Цельсия
#include "stdio.h"
#include "conio.h"

// пересчитывает температуру
// из градусов Фаренгейта в градусы Цельсия
float Fahr2Cels(float f)
```

```
{
    float c;

    c = (float) 5/9*(f - 32);
    return (c);

    // Вместо приведенных выше инструкций
    // можно написать:
    // return ( (float)5/9*(f - 32));
    //
}


void main()
{
    float f; // температура в градусах Фаренгейта
    float c; // температура в градусах Цельсия

    float f1,f2; // диапазон изменения температуры
    float df;    // шаг изменения температуры

    f1 = 0;
    f2 = 5;
    df = 0.5;

    printf("\n----------------");
    printf("\n  F        C");
    printf("\n----------------");
    f = f1;
    do
    {
        c = Fahr2Cels(f);
        printf("\n%5.2f   %5.2f", f, c);
        f = f + df;
    }
```

```
    while ( f <= f2 );
    printf("\n----------------");

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

202. Написать функцию пересчета длины из дюймов в миллиметры (1 дюйм = 2,54 см).

203. Написать функцию пересчета расстояния из миль в километры (1 миля = 1,60094 км).

204. Написать функцию пересчета цены нефти за баррель в цену за тонну (1 нефтяной баррель марки Urals равен 136,4 кг). Для проверки работоспособности функции написать программу, использующую эту функцию для пересчета цены за баррель в цену за тонну.

Задача 204

```
// Функция Barrel2Ton пересчитывает
// цену за баррель в цену за тонну

#include "stdafx.h"
#include "stdio.h"
#include "conio.h"

// пересчитывает цену за баррель
// в цену за тонну
float Barrel2Ton(float b)
{
    // b - цена барреля
    float t; // цена тонны

    t = b* (1000/136.4);

    //return (t);
```

```
    // Вместо приведенных выше инструкций
    // можно написать:
     return ( b* (1000/136.4) );
    //
}


void main()
{
    float barrel; // цена барреля
    float ton;    // цена тонны

    printf("\nЦена барреля-> ");
    scanf("%f", &barrel);

    ton = Barrel2Ton(barrel);

    printf("\Цена тонны:%6.2f", ton);

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

205. Написать функцию, которая вычисляет объем цилиндра.

Задача 205

```
#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();
}
```

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

Задача 206

```
// Функция max возвращает максимальное из двух чисел
int max(int a, int b)
{
    if (a > b)
        return(a);
    else
        return(b);
}
```

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

Задача 207

```
// 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();
}
```

208. Написать функцию, которая вычисляет сопротивление цепи, состоящей из двух резисторов, которые могут быть соединены последовательно или параллельно. Функция должна проверять корректность параметров: если неверно указан тип соединения, то функция должна возвращать –1.

Задача 208

```
// 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);
}
```

209. Написать функцию `percent`, которая возвращает процент от числа, полученного в качестве аргумента.

210. Написать функцию "Факториал" и программу, использующую эту функцию для вывода таблицы факториалов.

Задача 210

```
// Функция "факториал"
#include "stdio.h"
#include "conio.h"

unsigned long factorial(int x)
{
    unsigned long f = 1;
    for (int i = 2; i <= x; i++)
        f *= i;
    return(f);
}

void main()
{
    unsigned long f;
```

```
    puts("\nТаблица факториалов");
    for (int n = 1; n <= 12; n++)
    {
        f = factorial(n);
        printf("%2i  %10u\n", n, f);
    }

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

211. Написать функцию `profit`, которая вычисляет доход по вкладу. Исходные данные для функции: величина вклада, процентная ставка (годовых) и срок вклада (количество дней).

Задача 211

```
// Функция вычисляет доход по вкладу
float profit(float sum,     // сумма вклада
             float stavka,  // процентная ставка (годовых)
             int srok)      //срок вклада (дней)
{
    return(sum*(stavka/100/365)*srok);
}
```

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

Задача 212

```
// 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);
}
```

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

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

Задача 214

```
// 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();
}
```

215. Написать функцию, обеспечивающую решение квадратного уравнения. Параметрами функции должны быть коэффициенты и корни уравнения. Значением функции должна быть информация о корнях уравнения: 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.

Задача 215

```
// 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();
}
```

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

Задача 216

```
// 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 программы нажмите
<Enter>");
    getch();
}
```

217. Написать функцию, которая возвращает длину строки.

Задача 217

```
// Функция length вычисляет длину строки символов
#include "stdio.h"
#include "conio.h"


// возвращает длину строки
int length(char* st)
{
    int l = 0;  // длина строки
```

```
    char* p = st;  // указатель на символ

    while ( *p++  )
        l++;
    return(l);
}

void main()
{
    char st[80]; // строка
    int len;     // длина строки

    printf("Введите строку символов и нажмите <Enter>\n");
    printf("->");

    // если во введенной строке есть пробелы, то scanf
    // введет только часть строки - до первого пробела
    // поэтому будем использовать функцию gets
    gets(st);
    len = length(st);

    printf("Длина введенной строки: %i",len);

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

218. Написать функцию, которая выводит на экран строку символов. Длина строки и символ являются параметрами функции.

Задача 218

```
// Функция line выводит на экран строку из символов
#include "stdio.h"
#include "conio.h"
```

```
// выводит на экран строку, состоящую
// из n ch-символов
void line(char ch, int n)
{
    for (int i = 0; i < n; i++)
        putch(ch);
}

// возвращает длину строки
int length(char* st)
{
    int l = 0;  // длина строки
    char* p = st;  // указатель на символ

    while ( *p++  )
        l++;
    return(l);
}

void main()
{
    char mes[] = "Hello, World!\0";
    int len;

    len = length(mes);

    line('*', len);

    printf("\nHello, World!\n");
    line('*',len);
    printf("\nДля завершения нажмите <Enter>");
    getch();
}
```

Факультатив

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

Задача 219

```
// 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 <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 обеспечивает ввод ");
  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();
}
```

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

Задача 220

```
// 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();
}
```

221. Написать программу, реализующую игру "21". Действия по выдаче карты игроку или компьютеру реализуйте в виде функции.

Задача 221

```
// 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 карту? (введите y или n) ");
        otv = getchar();

        // Игрок нажимает две клавиши: с буквой и <Enter>.
        // Предыдущий вызов getchar прочитал букву.
        // В буфере клавиатуры остался 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;
}
```

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 отсчитываются от левого верхнего угла и возрастают слева направо и сверху вниз. Левая верхняя точка имеет координаты (0, 0), правая нижняя — (639, 479).

- Для того чтобы программа могла вывести на экран графику, например, нарисовать линию, окружность или прямоугольник, необходимо инициализировать графический режим.

Ниже приведен шаблон графической программы.

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

222. Написать программу, которая рисует на экране флаг России.

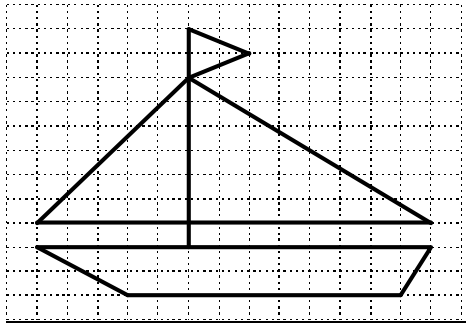
Задача 222

```
// 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 граф. 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
}
```

223. Написать программу, которая рисует на экране кораблик:



Задача 223

```
// Рисует кораблик (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
}
```

224. Написать программу, которая рисует на экране узор из 100 окружностей случайного диаметра и цвета.

Задача 224

```
// 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 графиче-
ского 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
}
```

225. Написать программу, которая рисует на экране узор из 50 произвольно размещенных прямоугольников случайного размера и цвета.

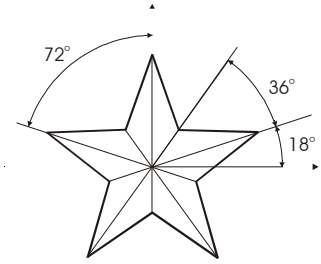
226. Написать программу, которая вычерчивает на экране узор — ломаную линию, состоящую из 100 произвольно расположенных звеньев случайного цвета.

Задача 226

```
// 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
граф.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
}
```

227. Написать программу, которая рисует на экране контур пятиконечной звезды.



Задача 227

```
// 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 граф. 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
}
```

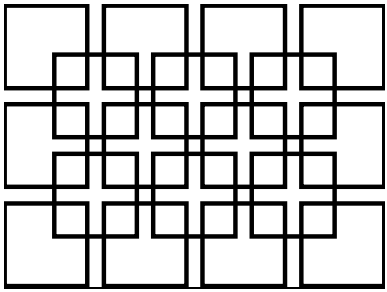
228. Написать программу, которая выводит на экран пятиконечную звезду с закрашенной внутренней областью.

Задача 228

```
// 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 граф. режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
   }
   else {
      star(100, 100, 20);
      getch();
      closegraph(); // выход из графического режима
   }
}
```

229. Написать программу, которая рисует изображенный ниже узор.



Задача 229

```
// 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 графиче-
ского режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   uzor();

   getch();
   closegraph(); // выход из графического режима
}
```

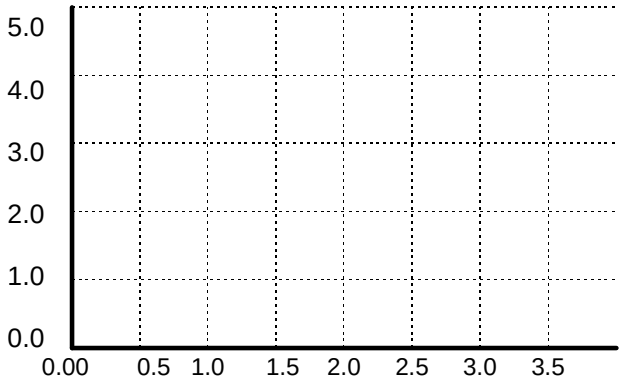
230. Написать программу, которая выводит на экран изображение шахматной доски.

Задача 230

```
// 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 граф. режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   doska();

   getch();
   closegraph(); // выход из графического режима
}
```

231. Написать программу, которая рисует на экране оцифрованную координатную сетку.



Задача 231

```
// 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  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;            // начало координат помечается метками 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 граф. режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   grid();

   getch();
   closegraph(); // выход из графического режима
}
```

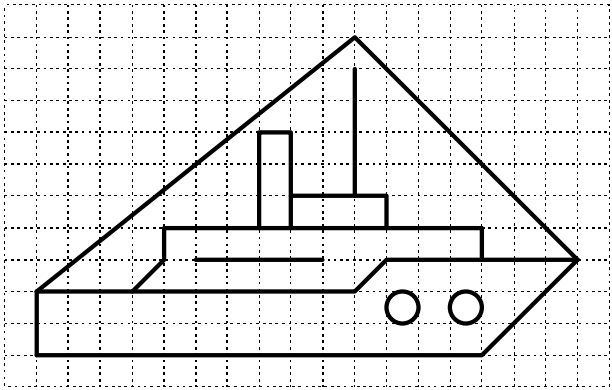
232. Написать программу, которая чертит на экране точечный график функции $y = 0{,}5x^2 + 4x - 3$. Диапазон изменения аргумента: от –15 до 5; шаг приращения аргумента: 0,1. График вывести на фоне координатных осей, точка пересечения которых должна находиться в центре экрана.

Задача 232

```
// 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 граф. режима
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
   }

   grafik();

   getch();
   closegraph(); // выход из графического режима
}
```

233. Написать программу, в окне которой "плывет" кораблик.



Задача 233

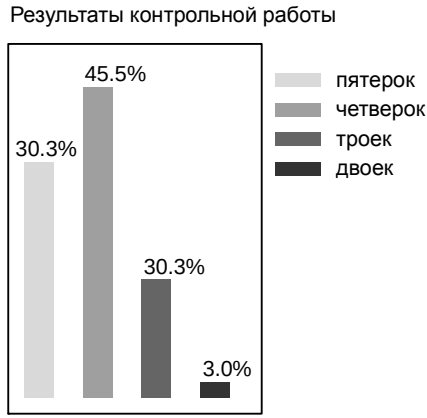
```
// Плывущий корабль
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>

#define dx  5  // шаг сетки по X
#define dy  5  // шаг сетки по Y

// Рисует кораблик
void ship(int x, int y, int color )
{
    // x,y - координаты базовой точки
    // color – цвет линий

    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 графиче-
ского режима
    {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения программы нажмите <Enter>");
      getch();
      return;
    }

    maxx = getmaxx();
    x = -10 ; // корабль выплывает из-за правой границы экра-
на
    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
}
```

234. Написать программу, которая рисует на экране диаграмму, отражающую результат контрольной работы. Исходные данные (количество пятерок, четверок, троек и двоек) должны вводиться во время работы программы с клавиатуры.



Задача 234

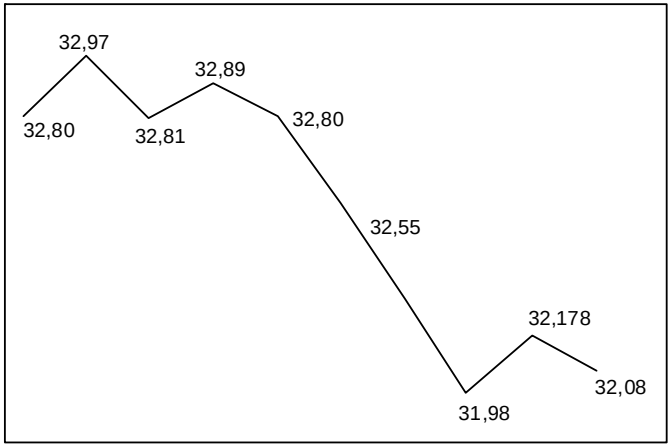
```
// Столбчатая диаграмма
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

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

    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

// ввод и обработка исходных данных
void obr()
{
    int s;      // всего оценок
    int m;      // номер максимального эл-та массива n
    int i;      // индекс массива

    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];
}

// строит диаграмму
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();
}
```

235. Написать программу, которая строит график изменения курса доллара. Данные должны загружаться из файла. При построении графика необходимо учесть, что диапазон изменения отображаемых данных незначителен. Поэтому, для того чтобы график был наглядным, его следует строить в отклонениях от минимального значения. Рекомендуемый вид графика приведен ниже.



Задача 235

```
#include "stdio.h"
#include "conio.h"
#include "graphics.h"

#define HB 10 // максимальное количество значений
#define WS 300
#define HS 150

   char  head[40];    // заголовок
   float kurs[HB];    // данные
   char* date[HB][5]; // подписи данных - дата в формате dd.mm

   int nrec; // количество элементов данных
```

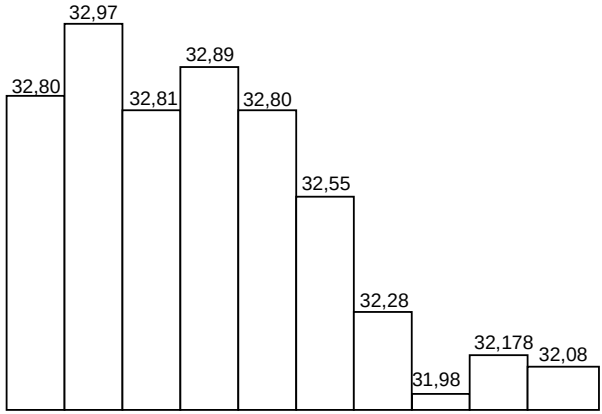
```
   int dx;     // шаг по x
   int py[HB];  // y-координаты точек на графике

// ввод и обработка исходных данных
void  LoadData()
{

     FILE* f; // файл данных

     int i;

    if ( (f = fopen("c:\\borlandc\\cpp\\kurs.txt", "rt")) ==
NULL)
    {
        printf("Ошибка доступа к файлу данных\n");
        getch();
        return;
    }

    fscanf(f,"%s",head);
    printf("%s\n",head);

    i = 0;
    while ((! feof(f) ) && ( i < HB ))
    {
        fscanf(f,"%s",&date[i]);
        printf("%s\n",date[i]);

        fscanf(f,"%f",&kurs[i]);
        printf("%5.2f\n",kurs[i]);
        i++;
    }
    nrec = i;
    fclose(f);

    // найти минимальный элемент массива
    float min = kurs[0];
```

```
    float max = kurs[0];

    for ( i = 1; i < nrec; i++ )
    {
        if ( kurs[i] < min)
            min = kurs[i];

        if ( kurs[i] > max)
            max = kurs[i];
     }

    // вычислить y-координаты точек
    for (i=0; i < nrec; i++)
        py[i] = (HS-20) * (kurs[i]- min)/(max - min)+2;

    return;
}

void Graphic()
{
    int x0, y0;
    int x;

    int dx;
    int i;

    char st[20];

    x0 = 10;
    y0 = HS + 20;
    dx = (WS - 2* x0)/ (nrec );

    outtextxy(x0,10, head );
    rectangle(1,1,WS,y0+10);

    x =x0;
```

```
    moveto(x,y0-py[0]);
    x =x +dx;

    for ( i = 1; i < nrec; i++)
    {
      lineto(x,y0-py[i]);
       x= x+dx;
    }

    x = x0;
    for ( i = 0; i < nrec; i++)
    {
     sprintf(st,"%2.2f", kurs[i]);
     outtextxy(x,y0 - py[i]-10,st);
       x= x+dx;
    }
}

void main(void)
{
   int gdriver = DETECT;
   int gmode;
   int errorcode;

   initgraph(&gdriver, &gmode, "c:\\borlandc\\bgi\\");
   errorcode = graphresult();

   if (errorcode != grOk)  // ®иЁЎЄ  Ё-ЁжЁ «Ё§ жЁЁ Ja д. аЃ¦Ё¬
   {
      printf("Ошибка: %d\n", errorcode);
      puts("Для завершения нажмите <Enter>");
      getch();
      return;
   }

   LoadData();
```

```
   Graphic();

   getch();
   closegraph();
}
```

236. Написать программу, которая отображает в виде столбчатой диаграммы динамику изменения курса доллара. Данные должны загружаться из файла. При построении диаграммы необходимо учесть, что диапазон изменения отображаемых данных незначителен. Поэтому, для того чтобы диаграмма была наглядной, ее следует строить в отклонениях от минимального значения. Рекомендуемый вид диаграммы приведен ниже.



Задача 236

```
// Столбчатая диаграмма
#include "stdio.h"
#include "conio.h"
#include "graphics.h"

#define HB 10 // максимальное количество значений
#define WS 300
```

```
#define HS 150

   char  head[40];    // заголовок
   float kurs[HB];    // данные
   char* date[HB][5]; // подписи данных - дата в формате dd.mm

   int nrec; // количество элементов данных (пар дата-курс)

   int dx;     // шаг по x
   int py[HB]; // y-координаты точек на графике

// ввод и обработка исходных данных
void  LoadData()
{

     FILE* f; // файл данных

     int i;

    if ( (f = fopen("c:\\borlandc\\cpp\\kurs.txt",
                    "rt")) == NULL)
    {
        printf("Ошибка доступа к файлу данных\n");
        getch();
        return;
    }

    // заголовок
    //fgets(head,sizeof(head),f);

    fscanf(f,"%s",head);
    printf("%s\n",head);

    i = 0;
    while ((! feof(f) ) && ( i < HB ))
```

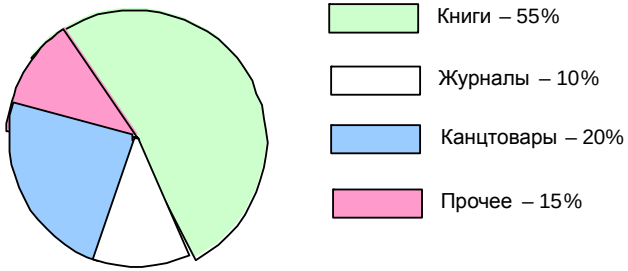
```
    {
        fscanf(f,"%s",&date[i]);
        printf("%s\n",date[i]);

        fscanf(f,"%f",&kurs[i]);
        printf("%5.2f\n",kurs[i]);
        i++;
    }
    nrec = i;
    fclose(f);

    // обработка данных:

    // найти минимальный элемент массива
    float min = kurs[0];
    float max = kurs[0];

    for ( i = 1; i < nrec; i++ )
    {
        if ( kurs[i] < min)
            min = kurs[i];

        if ( kurs[i] > max)
            max = kurs[i];
     }

    // вычислить y-координаты точек
    for (i=0; i < nrec; i++)
        py[i] = (HS-20) * (kurs[i]- min)/(max - min)+2;

    return;
}

void Diagr()
```

```
{
    int x0, y0;
    int x;

    int dx;
    int i;

    char st[20];

    x0 = 10;
    y0 = HS + 20;
    dx = (WS - 2* x0)/ (nrec );

    outtextxy(x0,10, head ); // заголовок
    rectangle(1,1,WS,y0+10); // рамка

    // столбики
    x = x0;
    setfillstyle(SOLID_FILL,GREEN);
    for ( i = 0; i < nrec; i++)
    {
       bar(x,y0,x+dx,y0-py[i]);
       rectangle(x,y0,x+dx,y0-py[i]);
       x= x+dx;
    }

    // подписи данных
    x=x0;
    for ( i = 0; i < nrec; i++)
    {
       sprintf(st,"%5.2f", kurs[i]);
       outtextxy(x,y0 - py[i]-10,st);
       x= x+dx;
    }
}
```

```
void main(void)
{
   int gdriver = DETECT;
   int gmode;
   int errorcode;
   initgraph(&gdriver, &gmode, "c:\\borlandc\\bgi\\");
   errorcode = graphresult();

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

   LoadData(); // загрузить данные

   Diagr();    // построить диаграмму

   getch();
   closegraph();
}
```

237. Измените программу отображения результатов экзамена в виде столбчатой диаграммы (см. задачу 235) так, чтобы исходные данные, в том числе заголовок и подписи данных, загружались из файла.

238. Написать программу, которая выводит на экран 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 далее.



Задача 238

```
// 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();
   }
}
```

Факультатив

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

Задача 239

```
// 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. угла 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();
   }
}
```

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

Задача 240

```
// 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 значение 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();
   }
}
```

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 стал доступен, его нужно открыть, указав, для выполнения какой операции открывается файл (чтение, запись или обновление данных) и тип файла (двоичный или текстовый).
- При работе с файлами возможны ошибки. Поэтому рекомендуется проверять результат выполнения потенциально опасных, с точки зрения возникновения ошибок, операций с файлами (например, `fopen`).
- Если в инструкции открытия файла путь к файлу не указан, то по умолчанию предполагается, что файл находится в том же каталоге, что и выполняемый файл программы.
- При записи в тексте программы пути к файлу, следует помнить, что символ `\` в строковых константах обозначает начало специальной последовательности символов (например, `\n`). Поэтому, чтобы путь к файлу был интерпретирован правильно, символ `\` следует продублировать (например, `c:\\temp`).

- Чтение данных из текстового файла обеспечивает функция `fscanf`, запись — `fprintf`.
- По завершении работы с файлом его нужно обязательно закрыть (функция `fclose`).

Задачи

241. Написать программу, которая создает на диске компьютера файл numbers.txt и записывает в него 5 целых чисел, введенных пользователем с клавиатуры. Откройте созданный программой файл и убедитесь, что каждое число находится в отдельной строке.

Задача 241

```
// Создает на диске файл
#include "stdio.h"
#include "conio.h"

#define FNAME "numbers.txt\0" // имя файла
#define N 5                   // количество чисел

// Создает на диске файл и записывает в него
// целые числа, введенные пользователем
void main()
{
     char fname[20] = FNAME;
     FILE *f;    // файл чисел
     int n;      // число

     puts("\nСоздание файла");
     printf("Веденные числа будут записаны в файл %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 добавления в конец файла, используйте
    // режим добавления (a)
    if ((f = fopen(fname, "wt")) == NULL)
    {
       printf("Ошибка открытия файла для записи");
       getch();
       return;
    }

    for (int i = 0; i < N; i++)
    {
       printf("->");
       scanf("%i", &n);
       fprintf(f,"%i",n);
    }
    fclose(f);     // закрыть файл
    printf("Введенные числа записаны в файл %s\n", fname);
    puts("\nДля завершения нажмите <Enter>");
    getch();
}
```

242. Написать программу, которая дописывает в файл numbers.txt три целых числа, введенных пользователем. Убедитесь, открыв файл при помощи редактора текста, что в файле находятся 8 чисел.

Задача 242

```
// добавляет данные в файл
#include "stdio.h"
#include "conio.h"

#define FNAME " numbers.txt\0" // имя файла
#define N 3                    // количество чисел
```

```
// Дописывает в находящийся на диске файл numbers.txt
// целые числа, введенные пользователем
void main()
{
     char fname[20] = FNAME;
     FILE *f;     // файл чисел
     int n;       // число

     puts("\nДобавление в файл");
     printf("Веденные числа будут добавлены в файл %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 ((f = fopen(fname, "at")) == NULL)
     {
        printf("Ошибка открытия файла для добавления");
        getch();
        return;
     }

     for (int i = 0; i < N; i++)
     {
        printf("->");
        scanf("%i", &n);
        fprintf(f,"%i\n",n);
    }
    fclose(out);     // закрыть файл
    printf("Введенные числа добавлены в файл %s\n", fname);
    puts("\nДля завершения нажмите <Enter>");
    getch();
}
```

243. Написать программу, которая выводит на экран содержимое файла numbers.txt.

Задача 243

```
// Выводит на экран содержимое файла
#include "stdio.h"
#include "conio.h"

#define FNAME "numbers.txt\0" // имя файла

void main()
{
     char fname[20] = FNAME;
     FILE *f;     // текстовый файл
     char st[80];  // строка из файла

     printf("\nСодержимое файла %s\n", fname);
     puts("--------------------------------");

     // Открыть файл в режиме чтения (r) текста (t)
     if ((f = fopen(fname, "rt")) == NULL)
     {
        printf("Ошибка открытия файла для чтения");
        getch();
        return;
     }

     while (!feof(a))
     {
        fscanf(f,"%s", &st);
        printf("%s\n", st);
     }
     fclose(f);     // закрыть файл

     puts("--------------------------------");
```

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

244. Написать программу, которая вычисляет среднее арифметическое чисел, находящихся в файле numbers.txt.

Задача 244

```
// Вычисляет среднее арифметическое чисел,
// находящихся в файлах
#include "stdio.h"
#include "conio.h"

#define FNAME "numbers.txt\0" // имя файла

void main()
{
    char fname[20] = FNAME;
    FILE *f;     // 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 ((f = fopen(fname, "rt")) == NULL)
    {
       printf("Ошибка открытия файла для чтения");
       getch();
       return;
    }
```

```
    while (!feof(f))
    {
       fscanf(f,"%i", &a);
       sum += a;
       n++;
    }
    fclose(f);     // 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();
}
```

245. Написать программу, которая записывает в файл данные, находящиеся в двумерном массиве дробного типа.

Задача 245

```
// Запись данных из массива в файл
#include "stdio.h"
#include "conio.h"
#include "string.h"

void main()
{
#define NR 3
#define NC 6

    // массив данных
    float a[NR][NC] =
     {
        15.0,16.5,18.0,19.5,21.0,24.0,
```

```
   16.5,18.0,19.5,21.0,22.5,24.0,
   18.0,19.5,21.0,22.5,24.0,27.0
};

FILE *f; // файл

int r,c; // номер строки и столбца

// показать данные
printf("\nДанные:\n");
for ( r = 0; r < NR; r++)
{
   for (c = 0; c < NC; c++)
   {
       printf("%5.2f   ", a[r][c]);
   }
   printf("\n");
}

// предполагается, что строки текстового файла
// будут содержать значения строк элементов массива

 // Открыть для записи (w) текстовый файл (t)
 // Если файл с указанным именем уже существует,
 // он будет заменен новым
 if ((f = fopen("a.dat", "wt")) == NULL)
 {
     printf("Ошибка создания файла\n");
     printf("Для завершения нажмите <Enter>");
     getch();
     return;
 }

 for ( r = 0; r < NR; r++)
 {
    for (c = 0; c < NC; c++)
```

```
        {
           fprintf(f, "%5.2f ", a[r][c]);
        }
        if ( r != NR-1)
           fprintf(f, "\n");
     }
     fclose(f);
     printf("Данные записаны в файл\n");

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

246. Написать программу, которая загружает из файла данные в двумерный массив дробного типа.

Задача 246

```
#include "stdio.h"
#include "conio.h"
#include "string.h"

void main()
{
#define NR 3
#define NC 6

   float a[NR][NC]; // массив NRxNC - NR строк из NC элемен-
тов

   FILE *f; // файл

   // предполагается, что строки текстового файла
   // содержат значения строк элементов массива

    // Открыть для чтения (r) текстовый файл (t)
    if ((f = fopen("c:\\borlandc\\cpp\\a.dat", "rt")) == NULL)
```

```
    {
       printf("Нет файла a.dat");
       printf("\nДля завершения нажмите <Enter>");
       getch();
       return;
    }

    int r,c; // номер строки и столбца

    for ( r = 0; r < NR; r++)
    {
       for (c = 0; c < NC; c++)
       {
           fscanf(f, "%f", &a[r][c]);
       }
    }
    fclose(f);

    printf("\nДанные, загруженные в массив из файла:\n");
    // показать прочитанные данные
    for ( r = 0; r < NR; r++)
    {
       for (c = 0; c < NC; c++)
       {
           printf("%7.2f  ", a[r][c]);
       }
       printf("\n");
    }

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

247. Написать программу, которая позволяет просматривать текстовые файлы (выводит на экран их содержимое), например,

файлы исходных программ C++. Имя просматриваемого файла должно вводиться пользователем во время работы программы.

Задача 247

```
// 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 *a;         // 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 ((f = fopen(fname, "rt")) == NULL)
    {
        printf("Ошибка при обращении к файлу %s\n", fname);
        getch();
        return;
    }

    clrscr();
```

```
    while (!feof(f))
    {
        fgets(st, MAXLEN, f);
        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(f);     // закрыть файл

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

248. Написать программу, которая дописывает в находящийся на диске компьютера файл contacts.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 находиться в отдельной строке. Ниже приведен рекомендуемый вид экрана во время работы программы.

```
Добавление информации в телефонный справочник
Фамилия -> Сидоров
Имя -> Вася
Телефон -> 234-84-37

Информация добавлена
Для завершения нажмите <Enter>
```

Задача 248

```
// Дописывает в файл contacts.txt фамилию, имя и номер телефона
#include "stdio.h"
#include "conio.h"

#define FNAME "contacts.txt\0" // имя файла
void main()
{
    char fname[20] = FNAME;
    FILE *f;  // файл

    char fam[15];  // фамилия
    char name[15]; // имя
    char tel[9];   // номер телефона

    puts("\nДобавление информации в телефонный справочник\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;
    }

    // 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(f,"%s %s %s", fam, name, tel);
```

```
    puts("\nИнформация добавлена");
    fclose(f);     // закрыть файл

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

249. Усовершенствуйте программу работы с телефонным справочником (см. задачу 247) так, чтобы за один сеанс работы в файл contacts.txt можно было добавить информацию о нескольких людях. Рекомендуемый вид экрана во время работы программы приведен ниже.

```
Добавление информации в телефонный справочник
Для завершения вместо ввода фамилии нажмите <Enter>

Фамилия -> Иванов
Имя -> Иван
Телефон -> 234-84-37
Информация добавлена

Фамилия -> Орлов
Имя -> Андрей
Телефон -> 552-18-40
Информация добавлена

Фамилия ->
Ввод завершен
Для завершения нажмите <Enter>
```

250. Написать программу, которая позволяет найти в телефонном справочнике (в файле contacts.txt) нужную информацию. Программа должна запрашивать у пользователя фамилию человека и выводить информацию о нем. 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.

```
Телефонный справочник
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
->
```

Задача 250

```
// Поиск в телефонном справочнике
#include "stdio.h"
#include "conio.h"

#define FNAME "contacts.txt\0" // имя файла
void main()
{
    char fname[20] = FNAME;
    FILE *f;  // 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 ((f = 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(f))
    {
        fscanf(f,"%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(f);     // закрыть файл

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

251. Написать программу, которая объединяет возможности программ "Добавление в телефонный справочник" и "Поиск в телефонном справочнике". При запуске программы на экране должно отображаться меню, вид которого приведен ниже.

```
*** Телефонный справочник ***
1 - Добавление
2 - Поиск
0 - Завершение работы

Ваш выбор ->
```

Факультатив

252. Написать программу, при помощи которой можно автоматизировать процесс проверки знаний. Тест, последовательность вопросов и вариантов ответа, должен находиться в текстовом файле. Имя файла теста программа должна получать из команд-

ной строки ее запуска. 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 мене чем на 80% вопросов, удовлетворительно — если правильных ответов более 60%, и неудовлетворительно — если правильных ответов меньше 60%.

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

Структура файла теста

```
Вопрос1
N1 K1
Ответ1
...
Ответk
Вопрос2
N2 K2
Ответ1
...
Ответk

Вопросm
Nm Km
Ответ1
...
Ответk
```

Пример файла теста

```
В инструкции do wile после while записывают
2 2
условие завершения цикла
условие повторения инструкций цикла
Переменная unsigned int может принимать значение
2 1
0 … 65535
0 … 32767
```

```
Какая из приведенных ниже инструкций содержит ошибку
2 2
scanf ("%f", &kurs);
scanf("%f", kurs);


Сейчас Вам будет предложен тест.
К каждому вопросу дается несколько вариантов ответа.
Вы должны ввести номер правильного ответа и нажать <Enter>

В инструкции do wile после while записывают
1 – условие завершения цикла
2 – условие повторения инструкций цикла
Ваш выбор-> 2

Переменная unsigned int может принимать значение
1 – 0 … 65535
2 – 0 … 32767
Ваш выбор-> 1

Какая из приведенных ниже инструкций содержит ошибку
1 – scanf ("%f", &kurs);
2 – scanf("%f", kurs);
Ваш выбор-> 2

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

Задача 252

```
// Универсальная программа проверки знаний
// 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 процет 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();
}
```

253. Написать программу, которая выводит на экран список Си-программ, находящихся в указанном пользователем каталоге. Предполагается, что первая строка программы является комментарием и содержит информацию о программе.

Задача 253

```
// Выводит имена всех файлов программ
// предполагается, что первая строка
// файла - комментарий
#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();
}
```

Рекурсия

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

- Рекурсивной называется такая функция, которая может вызывать сама себя.
- Для окончания процесса рекурсии, в алгоритме рекурсивной функции обязательно должна быть веточка, обеспечивающая непосредственное завершение функции.

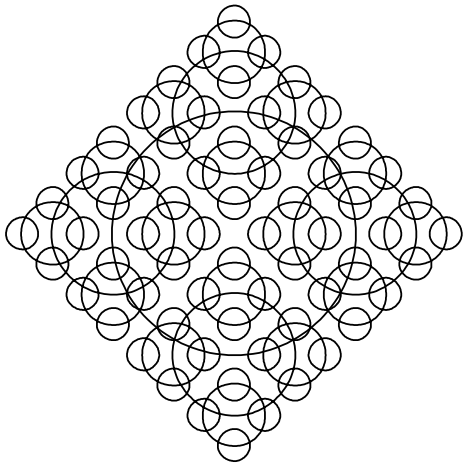
254. Написать рекурсивную функцию вычисления факториала и программу, проверяющую ее работоспособность.

Задача 254

```
// 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();
}
```

255. Написать программу, которая выводит на экран узор, приведенный ниже.



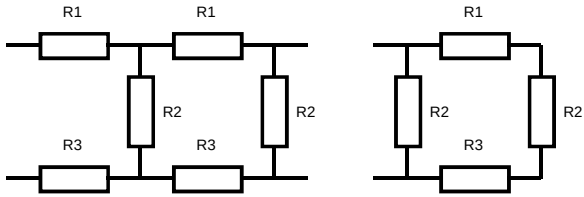
Задача 255

```
// 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();
   }
}
```

256. Написать программу, которая вычисляет сопротивление электрической цепи. Величины сопротивлений и порядок цепи (количество сопротивлений R2) должны вводиться во время работы программы.



Задача 256

```
// Вычисляет сопротивление
// рекурсивной 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 (n-1)-го порядка

   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();
}
```

257. Написать программу, которая рисует на экране схему, приведенную в задаче 255. Порядок цепи должен вводиться во время работы программы.

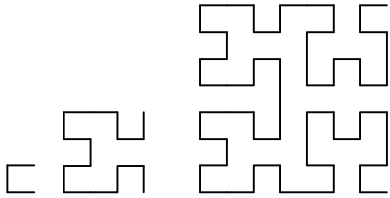
Задача 257

```
// Вычерчивает схему рекурсивной 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();
   }
}
```

Факультатив

258. Написать программу, которая рисует на экране кривую Гильберта (вид кривых первого, второго и третьего порядков приведен ниже).



259. Обратите внимание, что кривая второго порядка получается путем соединения четырех кривых первого порядка, две из которых повернуты на 90 градусов: одна по часовой стрелке, другая — против. Аналогичным образом получается кривая третьего порядка, но при этом в качестве "кирпичиков" используются кривые второго порядка.

Задача 259

```
// 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)
```

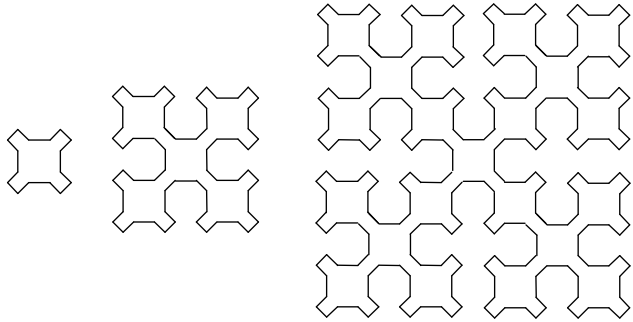
```
{
   int gdriver = DETECT; // драйвер
   int gmode;            // режим
   int errorcode;        // код ошибки

   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;    // коэффициенты 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);
}
```

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



Задача 260

```
// 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
```

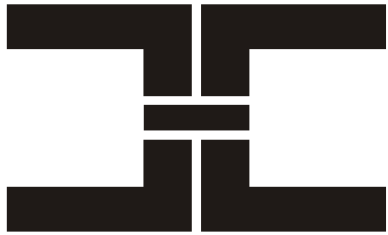
```
// координаты конца задаются в приращениях
#define linetodxy(dx,dy) lineto(getx()+dx,gety()+dy)

void far lineto(int x2, int y2); // вычерчивает линию по точкам
// заменяет стандартную функцию, чтобы
// видеть процесс вычерчивания кривой Серпинского

// 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;  // коэффициенты 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 II

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 русского алфавита отображались правильно, 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` — число элементов, указанное в инструкции объявления массива.

Инструкция присваивания

Инструкция	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 условие истинно
    // ( значение выражения Условие не равно нулю )

}
```

Вариант 2

```
if ( Условие )
{
    // Здесь инструкции, которые будут
    // выполнены, если условие истинно
    // ( значение выражения Условие не равно нулю )
}
else
{
    // Здесь инструкции, которые будут
    // выполнены, если условие не выполняется
    // (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 (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 истинно (не равно нулю), то выполняются инструкции цикла (тело цикла). Затем снова проверяется значение выражения *УсловиеВыполнения*, и если оно истинно (не равно нулю), инструкции цикла выполняются еще раз. И так до тех пор, пока значение выражения *УсловиеВыполнения* не станет ложным (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.

Функции ввода-вывода

printf

Синтаксис:

```
int printf(Формат, СписокПеременных);
```

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 %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

(окончание)

Спецификатор	Форма вывода
%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 %d	Десятичное число со знаком
%u	Беззнаковое целое десятичное число

(окончание)

Спецификатор	Вводит
%f %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 новую 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>`

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>`

textcolor

Синтаксис:

```
void textcolor(int Цвет);
```

Задает цвет для выводимого функциями `cputs` и `cprintf` текста. В качестве параметра *Цвет* обычно используют одну из перечисленных далее 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

(окончание)

Цвет	Константа	Значение константы
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: `<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 далее 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 строки, 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 чтение. Файл открывается только для чтения
w	Только запись. Файл открывается для записи. Если файл с именем, указанным в качестве первого параметра функции fopen, уже существует, то новые данные записываются поверх старых, т. е. старый файл фактически уничтожается
A	Добавление. Файл открывается для записи данных в конец существующего файла. Если файл с именем, 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 *Строка2* в строку *Строка1*.

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 не найден, NULL.

Заголовочный файл: <string.h>

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, вызвав 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 далее 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>`

Функции графического режима

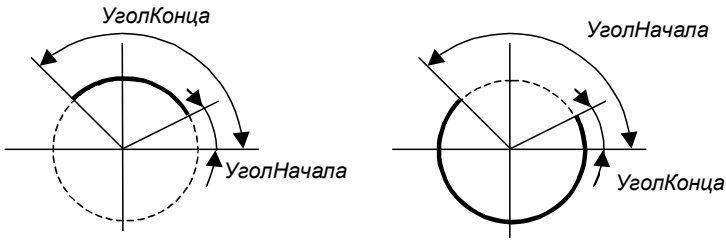
arc

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>`

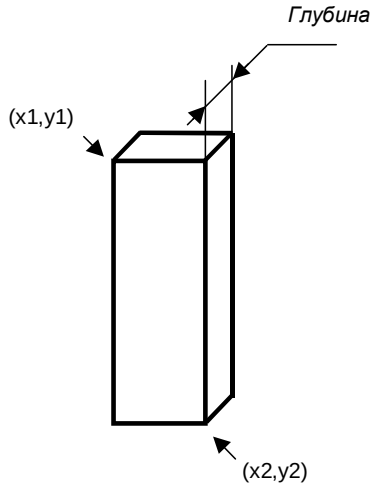
bar3d

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 *B_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.

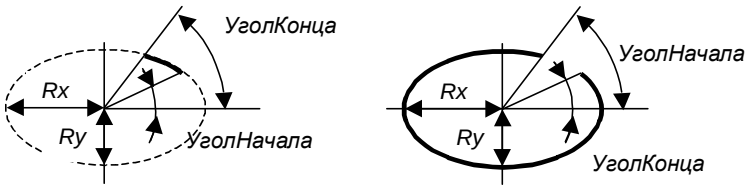
Заголовочный файл: `<graph.h>`

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 начальной 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 вычерчи-

вается 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>`

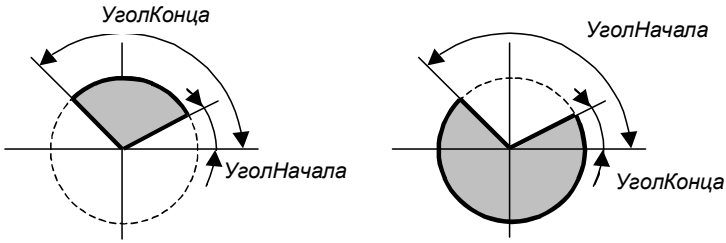
pieslice

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 окруж-

ности, 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`.

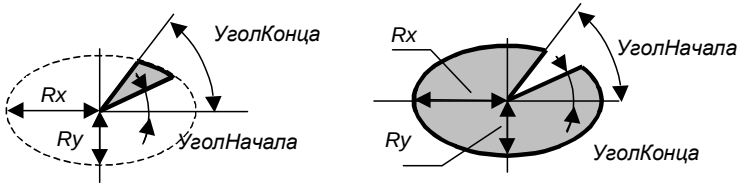
Заголовочный файл: `<graph.h>`

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 *УголНачала* и *УголКонца* — 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 далее. 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° влево тонкими линиями
SLASH_FILL	Штриховка под углом 45° влево
BKSLASH_FILL	Штриховка под углом 45° вправо тонкими линиями
LTBKSLASH_FILL	Штриховка под углом 45° вправо
HATCH_FILL	Штриховка клеткой
XHATCH_FILL	Штриховка под углом 45° косой клеткой
INTERLEAVE_FILL	Штриховка под углом 45° 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	Штрихпунктирная 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 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 8x8 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 гер-

цах. Динамик будет издавать сигнал до тех пор, пока программа его не выключит при помощи функции nosound.

Далее приведены частоты, соответствующие первым двум октавам пианино.

Нота	Частота, Гц
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>

ПРИЛОЖЕНИЯ

Приложение 1

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 16-цветную
   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
           16-цветной.
*/

// 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;
```

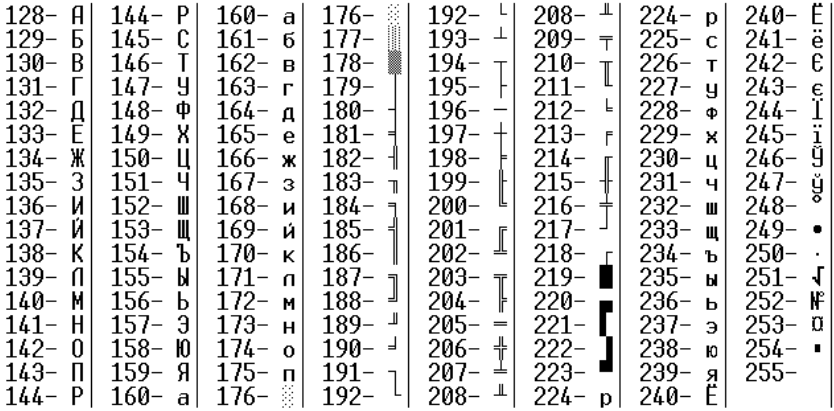
```
            putpixel(x,y,color[bh]);
            x++;
            np++;
          }
          if (np < info.width)
          {
            bl = b & 15;
            putpixel(x,y,color[bl]);
            x++;
            np++;
          }
        }
        x=x0;
        y--;
    }
    fclose(f);
    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.



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-го разряда равен i-й степени десятки (рис. П1).

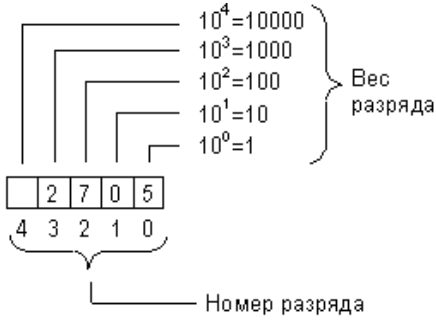
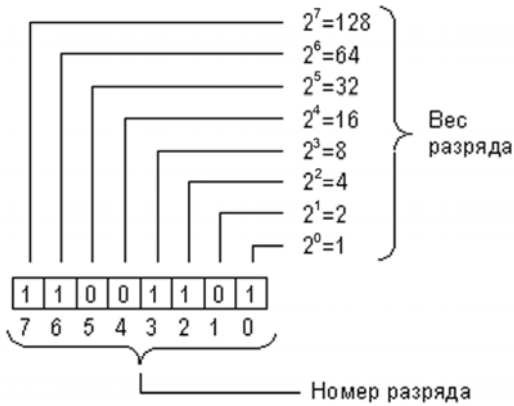


Рис. П1. Вес разрядов в десятичной системе счисления

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 *i*-го разряда равен двум в *i*-й степени (рис. П2).



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

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.

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.

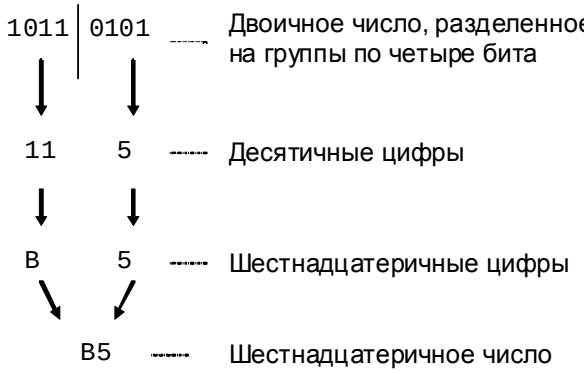


Рис. П3. Перевод двоичного числа в шестнадцатеричное

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`.

Приложение 2

Описание компакт-диска

Прилагаемый компакт-диск содержит приведенные в книге примеры — исходные тексты программ. Для активной работы, чтобы иметь возможность вносить изменения в программы, скопируйте содержимое компакт-диска на жесткий диск своего компьютера.

Список дополнительной литературы

1. Керниган Б., Ритчи Д., Фьюэр А. Язык программирования Си. — М.: Финансы и статистика, 1985. — 279 с.
2. Уинер Р. Язык Turbo Си: Пер. с англ. — М.: Мир, 1991. — 384 с.
3. Уэйт М., Прата С., Мартин Д. Язык Си. Руководство для начинающих: Пер. с англ. — М.: Мир, 1988. — 512 с.
4. Вирт Н. Алгоритмы и структуры данных: Пер. с англ. — М.: Мир, 1989. — 360 с.
5. Зелковиц М., Шоу А., Гэннон Дж. Принципы разработки программного обеспечения: Пер. с англ. — М.: Мир, 1982. — 386 с.
6. Мик Б. и др. Практическое руководство по программированию: Пер. с англ. — М.: Радио и связь, 1986. — 168 с.
7. Фокс Дж. Программное обеспечение и его разработка: Пер. с англ. — М.: Мир, 1985. — 368 с.
8. Язык компьютера / Под ред. В. М. Курочкина. Пер. с англ. — М.: Мир, 1989. — 240 с.

Предметный указатель

(окончание рубрики см. на стр. 348)

Н, П

Р, С

Т

Ф

Ц

Ш