Никита Культин

C/C++

в задачах и примерах

3-е издание

Более 200 задач

Исходные тексты программ с комментариями

Справочник по языку программирования

Описание стандартных функций

Никита Культин

C/C++
в задачах и примерах

3-е издание,
дополненное и исправленное

Санкт-Петербург
«БХВ-Петербург»
2019

УДК 004.438 С
ББК 32.973.26-018.1
К90

Культин Н. Б.

К90 С/С++ в задачах и примерах. —
3-е изд., доп. и исправл. — СПб.: БХВ-Петербург, 2019. —
272 с.: ил.

ISBN 978-5-9775-3996-8

Сборник примеров и задач для самостоятельного решения по программированию на языке С/С++ охватывает практически все разделы начального курса программирования: от задач консольного ввода/вывода, использования циклов и операций с массивами до работы со строками, файлами и объектами. Примеры представлены в виде хорошо документированных исходных текстов программ. Книга содержит справочник — описание основных типов данных, операторов и наиболее часто используемых функций. Адресована студентам, школьникам старших классов и всем тем, кто изучает программирование в учебном заведении или самостоятельно. В третьем издании добавлены и обновлены примеры.

Для начинающих программистов

УДК 004.438 С
ББК 32.973.26-018.1

Группа подготовки издания:

Руководитель проекта	*Евгений Рыбаков*
Зав. редакцией	*Екатерина Капалыгина*
Компьютерная верстка	*Ольги Сергиенко*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Марины Дамбиевой*

Подписано в печать 09.01.19.
Формат 60×90$^{1}/_{16}$. Печать офсетная. Усл. печ. л. 17.
Тираж 1000 экз. Заказ № 8257.
"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.

Отпечатано с готового оригинал-макета
ООО "Принт-М", 142300, М.О., г. Чехов, ул. Полиграфистов, д. 1

ISBN 978-5-9775-3996-8

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 двух приложений:

- *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 С. В нем приведено описание основных типов данных, инструкций, реализующих алгоритмические структуры выбора и циклов, а также наиболее часто используемых функций.

О КОМПИЛЯТОРЕ И СРЕДЕ РАЗРАБОТКИ

Если вы только начинаете осваивать язык C/C++, то перед вами встанет задача выбора среды разработки, компилятора. Существует достаточно много инструментов разработки на C/C++, среди которых можно выделить Microsoft Visual Studio, Qt Creator и Embarcadero C++ Builder. Какой из этих сред отдать предпочтение, на чем остановить свой выбор? По большому счету, особой разницы между этими средами для решения задачи освоения языка программирования нет. Скорее всего, при выборе среды программирования вы будете ориентироваться на рекомендации преподавателя или потенциального работодателя.

Решения, приведенные в книге, представляют собой исходные тексты консольных Win32 приложений. Они не ориентированы на конкретную среду разработки, хотя тестировались в Microsoft Visual Studio, поэтому вы можете использовать любую среду разработки, которую выберете.

> ***Внимание!***
>
> Чтобы консольное Win32 приложение правильно отображало буквы русского алфавита, строковую константу, содержащую буквы русского алфавита, необходимо преобразовать в кодировку ASCII. Сделать это можно, например, так, как показано в решении *задачи 37*.
>
> В решениях (текстах программ) для наглядности все сообщения записаны русскими буквами, поэтому читателю при вводе программ в компьютер придется или заменить русские сообщения эквивалентными на английском языке, или применить к строковым константам, содержащим русские буквы, функцию rus, текст и пример использования которой приведены в решении *задачи 37*.

КАК РАБОТАТЬ С КНИГОЙ?

Группы задач следуют в книге в том порядке, в котором традиционно изучаются соответствующие разделы в курсе программирования. Прежде чем приступить к решению задач, нужно изучить соответствующую тему — прочитать посвященный ей 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 программу в редакторе кода), рекомендуется «набросать» блок-схему алгоритма решения на бумаге.

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

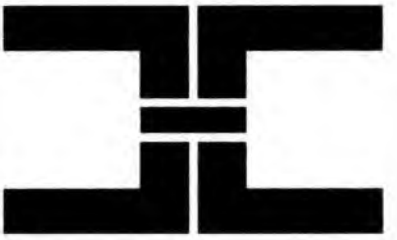
ОФОРМЛЕНИЕ РЕШЕНИЙ

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;
- комментирование назначения переменных, функций и ключевых точек программы.

Правильно оформленную программу легче читать, в ней легче найти ошибку, кроме того, программа, соответствующая правилам хорошего стиля программирования, характеризует уровень профессионализма ее разработчика.

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



Часть 1

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. Обратите внимание: 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 выполняются без учета переполнения. Например, если переменная `n`, объявленная как `int`, имеет значение `32767`, то в результате выполне-

ния 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.

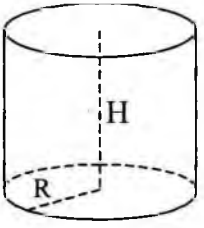
20. Записать в виде инструкции присваивания формулу вычисления площади трапеции: $s=\frac{a+b}{2}\cdot h$, где a и b — длины оснований, h — высота трапеции.

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.



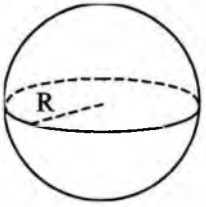
$$s = 2\cdot\pi\cdot r(h+r)$$
$$v = \pi\cdot r^2\cdot h$$

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.



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

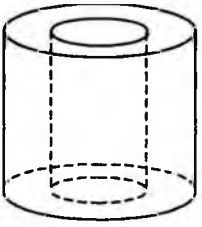
$$s=4\cdot\pi\cdot r^2$$

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.



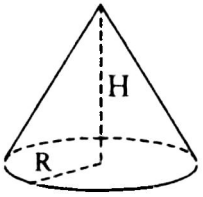
$$v=\pi\cdot h\cdot(r_1^2-r_2^2)$$

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 НА КОНСОЛЬ

Общие замечания

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

- функция `printf` обеспечивает вывод на экран (консоль) сообщений и значений переменных;
- первый параметр функции `printf` — строка вывода (форматирования), определяющая выводимый текст и формат отображения значений переменных, имена которых указаны в качестве остальных параметров функции;
- формат вывода значения переменной задается при помощи спецификатора преобразования (формата) — последовательности символов, начинающейся с `%`;
- при выводе числовых значений наиболее часто используются следующие спецификаторы:
  - `%i` — целое со знаком;
  - `%u` — беззнаковое целое;
  - `%f` — вещественное, в виде числа с плавающей точкой;
  - `%n.mf` — вещественное 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("To exit press  <Enter>");
getchar();
```

- чтобы программа могла выводить сообщения на русском языке, необходимо программно изменить кодировку строки сообщения — преобразовать ANSI-строку в ASCII-строку. Сделать это можно, например, при помощи функции rus, текст и пример использования которой приведен в решении *задачи 37*.

> ***Внимание!***
>
> Чтобы консольное Win32 приложение правильно отображало буквы русского алфавита, строковую константу, содержащую буквы русского алфавита, необходимо преобразовать в кодировку ASCII. Сделать это можно, например, так, как показано в решении *задачи 37*.

Задачи

35. Написать программу, которая выводит на экран имя сына Гомера Симпсона.

36. Написать программу, которая выводит на экран имена детей Гомера и Мардж Симпсонов (имя каждого — с новой строки).

Задача 36

```
#include <stdio.h>

void main()
{
    printf("Bart Simpson\nLisa Simpson\nMaggi Simpson\n");
    printf("To exit press <Enter>");
    getchar(); // ждет нажатия клавиши
}
```

37. Написать программу, которая выводит на экран имена детей Гомера и Мардж Симпсонов по-русски.

Задача 37

```
#include <stdio.h>

#include <string.h>

// преобразует ANSI-строку в ASCII-строку
char* rus(char st[])
{
    char* st2 = new char[strlen(st)  + 2];

    int i = 0;

    while (st[i] != 0)
    {
        unsigned char ch = st[i];
        if (( ch >= 192) && ( ch <= 255))
        {
            // буква русского алфавита
            if ( ch < 240)
                st2[i] = st[i] - 64;
            else
                st2[i] = st[i] - 16;
        }
        else
                  // остальные символы
                  st2[i] = st[i];

        i++;
    }
    st2[i] = 0;
    return st2;
}

void main()
{
    printf(rus("Барт Симпсон\n"));
```

```
    printf( rus("Лиза Симпсон\n") );
    printf( rus("Мэгги Симпсон\n") );

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

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 идти пояснительный текст (Height =, Width =), а после — единица измерения (sm).

Задача 39

```
printf("Height = %3.2f sm\Width = %3.2f sm\n", h, w);
```

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

Задача 40

```
printf("m=%i  n=%i\n", m, n);
```

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 вы-

веден поясняющий текст, представляющий собой имя переменной, за которым следует знак «равно».

Факультатив

- Чтобы иметь возможность выводить на экран текст разным цветом, нужно использовать потоковый вывод на консоль.
- Цвет символов задается при помощи функции:

  `SetConsoleTextAttribute`

- Чтобы потоковый вывод на консоль стал доступным, необходимо подключить модуль `iostrem` — добавить в программу директиву:

  ```
  #include <iostream>
  ```

- Чтобы иметь возможность задать цвет текста при потоковом выводе на консоль, надо в программу включить директиву:

  ```
  #include <Windows.h>
  ```

Задачи

43. Написать программу, которая выводит палитру — фразу Hello, World! всеми возможными для консольного приложения цветами. Перед каждой фразой необходимо указать код цвета, которым она выведена.

Задача 43

```
// Палитра
#include <stdio.h>
#include <iostream>
#include <Windows.h>

int main()
{
    HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);

    for (int i = 0; i < 16; i++)
    {
        SetConsoleTextAttribute(hConsole, i);
```

```
        std::cout << i << " - Hello, World!\n";
    }
    return 0;
}
```

44. Написать программу, которая выводит на экран в трех строках слово RUSSIA. Первая строка должна быть выведена белым, вторая — синим, третья — красным цветом.

Задача 44

```
// Выводит разноцветный текст
#include <stdio.h>

#include <iostream>
#include <Windows.h>

#define WHITE       7
#define GRAY        8
#define BLUE        9
#define GREEN       10
#define BLUE_LIGHT  11
#define RED         12
#define MAGENTA     13
#define YELLOW      14
#define WHITE_LIGHT 15

int main()
{
    HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);

    SetConsoleTextAttribute(hConsole, WHITE_LIGHT);
    std::cout << "RUSSIA" << std::endl;

    SetConsoleTextAttribute(hConsole, BLUE);
    std::cout << "RUSSIA" << std::endl;

    SetConsoleTextAttribute(hConsole, RED);
    std::cout <<  "RUSSIA" << std::endl;
```

```
    std::cin.get();

    return 0;
}
```

ВВОД С КОНСОЛИ

Общие замечания

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

- для ввода исходных данных с консоли (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 `&` — т. е. в качестве параметров функции `scanf` указываются адреса переменных, значения которых надо получить;
- 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 символа `&` перед именем переменной, указанной в качестве параметра функции `scanf`, является типичной ошибкой начинающих программистов (следует учесть, 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 вещественных 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

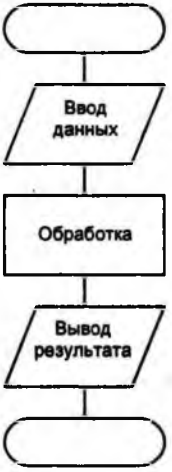
```
float sum;   // сумма
int period;  // срок (кол-во месяцев)
float rate;  // процентная ставка
float profit; // доход

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. Далее 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>

void main()
{
  float l,w; // длина и ширина прямоугольника
  float s;   // площадь прямоугольника

  printf("\nВычисление площади прямоугольника\n");
  printf("Введите исходные данные:\n");
  printf("Длина (см.)  -> ");
```

```
  scanf("%f", &l);
  printf("Ширина (см.) -> ");
  scanf("%f", &w);
  s = l * w;
  printf("Площадь прямоугольника: %10.2f кв. см\n", s);

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

52. Написать программу вычисления площади параллелограмма.

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

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

Объем: 337.50 куб. см.
```

54. Написать программу вычисления площади поверхности параллелепипеда. Далее 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>

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>");
  getchar();
}
```

55. Написать программу вычисления объема куба. Далее 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. Далее 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>

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>");
    getchar();
}
```

57. Написать программу вычисления стоимости покупки, состоящей из нескольких тетрадей и карандашей. Далее 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>

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>");
    getchar();
}
```

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

```
Вычисление площади треугольника
Введите в одной строке длины сторон
-> 25 17
Введите величину угла между сторонами
-> 30
Площадь треугольника: 106.25 кв. см.
```

Задача 61

```
// Вычисление площади треугольника по двум
// сторонам и величине угла между ними
#include <stdio.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>");
   getchar();
}
```

62. Написать программу вычисления сопротивления электрической цепи, состоящей из двух параллельно соединенных сопротивлений. Далее 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>

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>");
  getchar();
}
```

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

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

Поездка обойдется в 109.34 руб.
```

Задача 66

```
// Вычисление стоимости поездки на автомобиле
#include <stdio.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>");
    getchar();
}
```

67. Написать программу, которая вычисляет скорость, с которой спортсмен пробежал дистанцию. Далее приведен рекомендуемый вид экрана программы (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>

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>");
   getchar();
}
```

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

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

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

Задача 69

```
// Вычисление площади поверхности цилиндра
#include <stdio.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>");
    getchar();
}
```

70. Написать программу вычисления объема параллелепипеда. Далее 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). Далее 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>

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>");

   getchar();
}
```

72. Написать программу пересчета веса из фунтов в килограммы (1 российский фунт равен 405,9 г). Далее приведен рекомендуемый 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. Далее 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>

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 (месяцев) -> ");
```

```
  scanf("%i", &srok);
  printf("Процентная ставка (годовых) -> ");
  scanf("%f", &stavka);
  dohod = summ * stavka/12/100 * srok; 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>");
  getchar();
}
```

74. Написать программу пересчета величины временного интервала, заданного в минутах, в величину, выраженную в часах и минутах. Далее 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>

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>");
    getchar();
}
```

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. Далее 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>

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>");
    getchar();
}
```

76. Написать программу пересчета веса из фунтов в килограммы (1 фунт = 405,9 г). Далее 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. Далее 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.
```

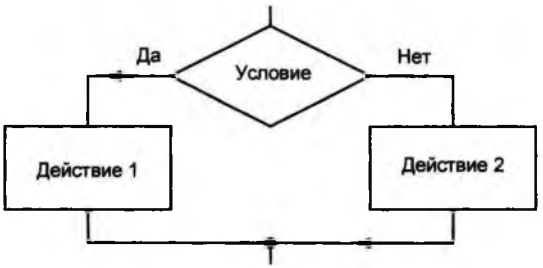
ВЫБОР

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. Далее 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>

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>");
  getchar();
}
```

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%. Далее 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>");
  getchar();
}
```

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%. Далее 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 вклада, 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>");
  getchar();
}
```

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

```
ФОТО
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>");
  getchar();
}
```

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. Далее 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>

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>");
    getchar();
}
```

83. Написать программу вычисления площади кольца. Программа должна проверять правильность исходных данных. Далее 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 не может быть больше радиуса кольца
```

Задача 83

```
// Вычисление площади кольца
#include <stdio.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>");
     getchar();
}
```

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. Далее приведен рекомендуемый вид экрана программы (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>

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>");
   getchar();
}
```

85. Написать программу, которая проверяет, является ли год високосным. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 85

```
// Проверяет, является ли год високосным
#include <stdio.h>

void main()
{
   int year;
   int r;    // остаток от деления year на 4

   printf("Введите год 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>");
   getchar();
}
```

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

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

Задача 86

```
// Решение квадратного уравнения
#include <stdio.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>");
    getchar();
}
```

87. Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 10% предоставляется, если сумма покупки больше 1000 руб. Далее 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. Далее приведен 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>

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>");
  getchar();
}
```

89. Написать программу проверки знания истории — например, даты основания Санкт-Петербурга. В случае неправильного ответа пользователя программа должна выводить правильную дату. Далее 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>

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>");
  getchar();
}
```

90. Написать программу проверки знания даты начала Второй мировой войны. В случае неправильного ответа пользователя программа должна выводить правильную дату. Далее 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. Далее 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>
```

```
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>");
    getchar();
}
```

92. Написать программу проверки знания истории архитектуры. Программа должна вывести вопрос и три варианта ответа. Пользователь должен выбрать правильный ответ и ввести его номер. Далее 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 сооб-

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

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

Задача 93

```
// Сравнение двух целых чисел
#include <stdio.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>");
    getchar();
}
```

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. Далее 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 <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>");
  getchar();
}
```

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. Далее 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. Далее приведен рекомендуемый 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>

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>");
    getchar();
}
```

97. Написать программу, которая проверяет, делится ли на три введенное с клавиатуры целое число. Далее 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. Далее 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>

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>");
  getchar();
}
```

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. Далее приведен рекомендуемый вид экрана программы (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>

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>");
  getchar();
}
```

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

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

Задача 100

```
// Определение времени года по номеру месяца
#include <stdio.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>");
    getchar();
}
```

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>

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>");
    getchar();
}
```

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

```
Введите в одной строке 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>

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
```

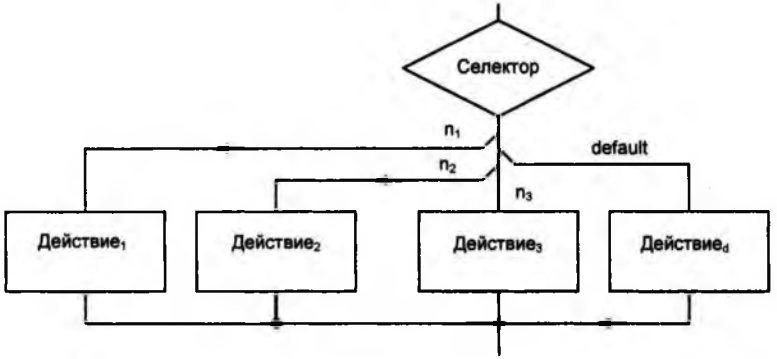
```
    printf("Введите в одной строке (цифрами) ");
    printf("сегодняшнюю дату\n");
    printf("(число месяц год) -> ");
    scanf("%i%i%i", &day, &month, &year);
    last = 0;
    if (month == 2) {
        if ((year % 4) != 0 && day == 28) last = 1;
        if ((year % 4) == 0 && day == 29) last = 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>");
     getchar();
}
```

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 вычислить цену жалюзи. Исходные данные: размер (ширина и высота, выраженные в сантиметрах) и тип материала (пластик, текстиль, алюминий). Далее 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>");
  getchar();
}
```

106. Написать программу, которая позволяет вычислить 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%. Далее 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>");
  getchar();
}
```

107. Написать программу, которая позволяет вычислить стоимость заправки автомобиля. Исходные данные: тип топлива (бензин 92, 95, 98 или дизельное топливо) и количество литров. Далее 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>");
  getchar();
}
```

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>

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");
    }
    getchar();
}
```

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>");
    getchar();
}
```

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. Далее 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 разговора (целое кол-во минут) -> 3
Город: Владивосток
Цена минуты: 2.20 руб.
Стоимость разговора: 6.60 руб.
```

Задача 110

```
// Определение стоимости междугороднего
// телефонного разговора
#include <stdio.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>");
   getchar();
}
```

111. Написать программу, которая по дате определяет соответствующий ей день недели. Для вычисления дня недели воспользуйтесь формулой:

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

Здесь d — число месяца, m — номер месяца, если начинать счет с марта, как это делали в Древнем Риме (март — 1, апрель — 2, ..., февраль — 12), Y — номер года в столетии, c — количество столетий. Квадратные скобки означают, что 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>

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");

    getchar();
}
```

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. Далее 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>
```

```
#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>");
    getchar();
}
```

114. Написать программу, которая выводит таблицу квадратов первых десяти целых положительных чисел. Далее 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>

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>");
    getchar();
}
```

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

```
------------------------
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. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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>");
    getchar();

}
```

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

```
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");

  getchar();
}
```

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. Далее 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>

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>");
    getchar();
}
```

121. Написать программу, которая вычисляет сумму первых *n* целых положительных четных чисел. Количество суммируемых чисел должно вводиться во время работы программы. Далее 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. Далее 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>

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 ряда -> ");
    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>");
    getchar();
}
```

123. Написать программу, которая вычисляет сумму первых n членов ряда: $1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\ldots$ Количество суммируемых членов ряда задается во время работы программы. Далее 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>

void main()
{
     int n;       // кол-во суммируемых членов ряда
     float i;     // номер элемента ряда; если обьявить 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>");
    getchar();
}
```

124. Написать программу, которая выводит таблицу степеней двойки: от нулевой до десятой. Далее 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>

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>");
    getchar();
}
```

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

```
Среднее 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. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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 последователь-

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

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

Для завершения нажмите <Enter>
```

Задача 128

```
// Среднее арифметическое дробных чисел, вводимых с клавиатуры
#include <stdio.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>");
    getchar();
}
```

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. Далее приведен рекомендуемый вид экрана программы (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>

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>");
     getchar();
}
```

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. Далее приведен рекомендуемый вид экрана программы.

```
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. Далее приведен рекомендуемый вид экрана программы.

```
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 <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>");
    getchar();
}
```

132. Написать программу, которая выводит на экран таблицу стоимости, например, яблок в диапазоне от 100 г до 1 кг с шагом 100 г. Далее 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 "math.h"

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

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

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

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

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. Далее приведен рекомендуемый вид экрана программы.

```
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>

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>");
    getchar();
}
```

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

```
   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

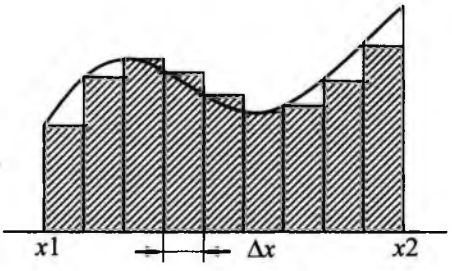
```
// Квадрат Пифагора - таблица умножения
#include <stdio.h>

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>");
    getchar();
}
```

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

Задача 137

```
// Вычисление числа "ПИ" с использованием
// свойства ряда 1 -1/3 + 1/5 - 1/7 + ...
#include <stdio.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>");
  getchar();
}
```

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.

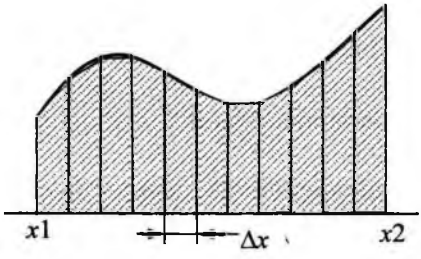


Задача 138

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

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>");
   getchar();
}
```

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>

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>");
    getchar();
}
```

140. Написать программу, которая выводит на экран изображение шахматной доски. Черные клетки отображать «звездочкой», белые — пробелом. Далее приведен рекомендуемый вид экрана программы.

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

141. Написать программу, которая преобразует введенное пользователем десятичное число в двоичное. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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>

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>");
   getchar();
}
```

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». Далее 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 <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>");
   getchar();
}
```

143. Написать программу проверки умения складывать и вычитать числа в пределах 100. Программа должна вывести 10 примеров, причем в каждом примере на вычитание 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». Далее приведен рекомендуемый вид экрана программы (ответы пользователя выделены полужирным).

```
Проверка умения складывать и вычитать числа.
После примера введите ответ и нажмите <Enter>
```

```
75+4=79
35-9=29
Вы ошиблись! 35-9=26
14-1=13
6+5=11
37-19=28
Вы ошиблись! 37-19=18
14+57=71
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 <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>");
    getchar();
}
```

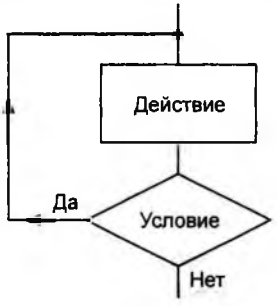
144. Написать программу, которая проверяет знание таблицы умножения. Программа должна вывести 10 примеров Оценка выставляется по следующему правилу: за 10 правильных ответов — «отлично», за 9 и 8 — «хорошо», за 7 и 6 — «удовлетворительно», за 6 и менее — «плохо». Программа должна вывести 10 примеров и после последнего ответа вывести оценку. Если в процессе тестирования количество ошибок достигнет 5 (т. е. получить положительную оценку испытуемый уже точно не сможет), то процесс тестирования должен быть прекращен.

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 присутствовать инструкции, выполнение которых влияет на условие завершения цикла;
- цикл `do ... while`, как правило, используется для организации приближенных вычислений, а также 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. Далее 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>

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>");
    getchar();
}
```

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

```
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>");

  getchar();
}
```

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

```
Вычисление среднего арифметического последовательности положительных чисел.
Вводите 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>

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>");
    getchar();
}
```

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

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

Задача 148

```
// Определение максимального числа
// в последовательности положительных чисел
#include <stdio.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>");
    getchar();
}
```

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

```
Определение минимального числа в последовательности положительных чисел.
Вводите 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>
```

```
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 ввода 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>");
    getchar();
}
```

150. Написать программу, которая проверяет, является ли введенное пользователем целое число простым (*простым* называется число, которое делится только на единицу и на само себя). Далее приведен рекомендуемый вид экрана программы (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>

void main()
{
     int n;  // число
```

```
    int d;  // делитель
    int r;  // остаток от деления n на d

    printf("Введите целое число и нажмите <Enter>\n-> ");
    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>");
   getchar();
}
```

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 <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>");
    getchar();
}
```

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

```
Компьютер "задумал" число от 1 до 10.
Вы должны его угадать за пять попыток.
Введите число и нажмите <Enter>
-> 5
Нет.
-> 3
ВЫ ВЫИГРАЛИ! Поздравляю!
```

Задача 152

```
// Игра "Угадай число"

#include <stdlib.h>  // для доступа к srand
#include <time.h>
#include <conio.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.");
   cprintf("Введите число и нажмите <Enter>");
   n = 0;
   do {
    cprintf("\n\r->");
```

```
    cscanf("%i",&igrok);
    n++;
  } while ((igrok != comp)&&(n < 3));

  if (igrok == comp)
  {
    cprintf("\n\rВы ВЫИГРАЛИ! Поздравляю!");
  }
  else
  {
    cprintf("\n\rВы проиграли.);
    cprintf("Компьютер \"задумал\" число %d",comp);
  }

  cprintf("\n\rДля завершения нажмите любую клавишу...");
  getchar();
  getchar();
}
```

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 `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 присутствовать инструкции, влияющие на значения переменных, входящих в условие выполнения цикла;
- цикл `while` обычно используется для организации приближенных вычислений, а также 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>

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>");
    getchar();
}
```

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 $\pi/4$. Далее приведен рекомендуемый вид экрана программы (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.
Для завершения нажмите <Enter>
```

Задача 155

```
// Вычисление числа "Пи"
#include <stdio.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);
        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>");
    getchar();
}
```

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>

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>");
  getchar();
}
```

МАССИВЫ

Общие замечания

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

- массив — это структура данных, представляющая собой совокупность элементов одного типа;
- в инструкции объявления массива указывается количество элементов массива;
- элементы массива нумеруются с нуля;
- доступ к элементу массива осуществляется путем указания индекса (номера) элемента. В качестве индекса можно 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. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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>

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>");
    getchar();
}
```

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 номе-

ром элемента. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 158

```
// Подсчет ненулевых элементов массива
// (доступ к элементам по номеру)
#include <stdio.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>");
    getchar();
}
```

159. Написать программу, которая выводит минимальный элемент введенного с клавиатуры массива целых чисел. Далее 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>

#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>");
    getchar();
}
```

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 элементам при помощи указателя)
#include <stdio.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>");
    getchar();
}
```

161. Написать программу, которая вычисляет среднее арифметическое ненулевых элементов введенного с клавиатуры массива целых чисел. Далее 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. Далее 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. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 163

```
// Вычисление среднедельной температуры воздуха
#include <stdio.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>");
    getchar();
}
```

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>

#define NB 5
void main()
```

```
{
    int m[НВ];  // 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", НВ);
    printf("чисел и нажмите <Enter>\n");
    printf("->");
    for (i = 0; i < НВ; 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 < НВ);
    if ( found )
         printf("Совпадение с элементом номер %i", i+1);
    else
         printf("Совпадений с образцом нет");

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

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

Задача 165

```
// Проверяет, отсортирован ли массив по возрастанию
#include <stdio.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>");
    getchar();
}
```

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>

#define НВ 5 // размер массива
```

```
void main()
{
    int a[HB]; // массив
    int obr;   // искомое число (образец)
    int n;     // кол-во элементов массива,
               // значение которых равно образцу
    int i;     // индекс

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

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>
```

```
#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>");
    getchar();
}
```

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

Задача 169

```
// Сортировка массива методом "пузырька"
#include <stdio.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>");
    getchar();
}
```

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

```
Объединение двух упорядоченных по возрастанию массивов
Введите первый массив (5 целых чисел) -> 1 3 5 7 9
Введите второй массив (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>
```

```
#define SZ 5  // размер исходных массивов
void main()
{
    int a[SZ], b[SZ];  // исходные массивы
    int c[SZ*2];       // массив - результат
    int k,i,m;         // индексы массивов a,b и c

    printf("Объединение двух упорядоченных ");
    printf("по возрастанию массивов\n");
    printf("Введите первый массив ");
    printf("(%i целых чисел) -> ", SZ);
    for (k = 0; k < SZ; k++)
        scanf("%i", &a[k]);

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

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

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

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

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

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

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>

#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>");
    getchar();
}
```

172. Написать программу, которая определяет количество учеников в классе, чей рост превышает средний. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, 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 учеников рост превышает средний
```

Задача 172

```
// Анализ роста учеников
#include <stdio.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>");
    getchar();
}
```

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>
```

```
#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>");
    getchar();
}
```

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 должна вычислить процент от общего количества оценок. Далее приведен рекомендуемый вид экрана программы (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>

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>");
    getchar();
}
```

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

```
Ввод по строкам
Введите элементы строки (5 чисел) и нажмите <Enter>
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>

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>");
    getchar();
}
```

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

Задача 177

```
// Строка с максимальной суммой элементов
#include <stdio.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");

    getchar();
}
```

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 далее 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>

#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>");
    getchar();
}
```

179. Написать программу, которая вычисляет доход по вкладу. Процентная ставка по вкладу определяется 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>");
  getchar();
}
```

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

```
Введите в одной строке количество золотых, серебряных и бронзовых медалей
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-я 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 соответствии 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>");
    getchar();
}
```

СИМВОЛЫ И СТРОКИ

Общие замечания

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

- каждому символу соответствует число — код этого символа;
- 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. При этом инициализировать массив и задать сообщения можно 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. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

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

Задача 181

```
// Приветствие
#include <stdio.h>

void main()
{
    char name[15]; // имя
    char fam[20];  // фамилия

    printf("Введите свое имя и фамилию, затем нажмите <Enter>");
    printf("-> ");
   // функция scanf читает из буфера клавиатуры символы
    // до разделителя - пробела

    scanf("%s", &name);
    scanf("%s", &fam);

    printf("Здравствуйте, %s %s!\n", name, fam);

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

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>

void main()
{
    char name[40]; // имя и отчество пользователя
    char ch;
    int i;

    printf("Ваше имя-> ");
    i = 0;
    while ((ch=getch()) != 13 && i < 40) // пока не <Enter>
    {
        putch(ch);
        name[i++] = ch;
    }
    name[i] = '\0';
    printf("\nПривет, %s!\n", name);

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

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

Задача 183

```
// Вычисляет длину строки
#include <stdio.h>

void main()
{
   char st[80]; // буфер для ввода строки
   int i = 0;   // длина строки
```

```
    puts("\nВведите строку и нажмите <Enter>");
    printf("->");
    gets(st);
    while( st[i++] != '\0')
        ;

    printf("Длина введенной строки: %i\n", i);
    printf("Для завершения работы нажмите <Enter>");
    getchar();
}
```

184. Написать программу, которая выводит на экран сообщение в «телеграфном» стиле: буквы слов сообщения должны быть разделены символами тире.

Задача 184

```
// Посимвольный вывод сообщения
#include <stdio.h>

void main()
{
    char msg[] = "\n\rПриветствую великого программиста!\0";
    int i;       // номер символа

    i = 0;
    while(msg[i])
    {
       putch(msg[i]);
       if ( msg[i+1] != ' '
           putch('-');
       i++;
    }

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

185. Написать программу, которая выводит код введенного пользователем символа. Программа должна завершать работу в ре-

зультате ввода, например, точки. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
Вводите символы.
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>

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>

#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>");
    getchar();
}
```

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

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

Задача 187

```
// Преобразование прописных букв в строчные
#include <stdio.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>");
    getchar();
}
```

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

Задача 188

```
// Удаление начальных пробелов из строки
#include <stdio.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>");
     getchar();
}
```

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

```
Полное имя -> Иван Иванович Иванов
Имя: Иван
Отчество: Иванович
Фамилия: Иванов
Для завершения нажмите <Enter>
```

Задача 189

```
// Разбивает строку на подстроки
#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 ->");
    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>");
    getchar();
}
```

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

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

Задача 190

```
// Проверяет, является ли строка целым числом
#include <stdio.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>");
    getchar();
}
```

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 "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>");
    getchar();
}
```

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

Задача 193

```
// Проверяет, является ли строка
// вещественным числом без знака

#include <stdio.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>");
    getchar();
}
```

194. Написать программу, которая преобразует введенное с клавиатуры восьмиразрядное двоичное число в десятичное. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, 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 <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>");
    getchar();
}
```

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

Задача 195

```
// Преобразует шестнадцатеричное число в десятичное
#include <stdio.h>

#include <string.h>
void main()
{
    char st[5];       // шестнадцатеричное число
    unsigned int dec;// десятичное число
    int v;            // вес разряда шестнадцатеричного числа
    int err = 0;      // err == 1 - в строке недопустимый символ
    int i;

    printf("Введите шестнадцатеричное ");
    printf("(не более 4-х знаков) число\n");
    printf("-> ");
    scanf("%s",&st);

    // преобразуем введенную строку к верхнему регистру
    strupr(st);

    dec = 0;
    v = 1;    // вес младшего разряда шестнадцатеричного числа
    for ( i = strlen(st) -1; i >= 0; i--)
    {
```

```
      //printf("\n%d\n",v);
      if (st[i] >= '0' && st[i] <= '9')
     dec += v * (st[i]- 48); // (int)'0'=48, (int)'1'=49 и т. д.
      else
     if (st[i] >= 'A' && st[i] <= 'F')
        // (int)'A'=65, (int)'B'=66 и т. д.
        // A обозначает 10, B - 11 и т. д.
        dec += v * (st[i]- 55);
     else // недопустимый символ
       { err = 1;
         break; }
      v *= 16;      // вес следующего разряда
   }
   if ( !err ) {
       printf("Шестнадцатеричному числу %s ", st);
       printf("соответствует десятичное %u\n", dec);
   }
   else {
       printf("Строка %s не является ", st);
       printf("шестнадцатеричным числом\n");
   }

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

196. Написать программу, которая преобразует введенное пользователем десятичное число в число в указанной системе счисления (от 2 до 10). Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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 до 10)
#include <stdio.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>");
    getchar();
}
```

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

Задача 197

```
// Преобразует десятичное число в шестнадцатеричное
#include <stdio.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>");
    getchar();
}
```

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

```
Введите арифметическое выражение,
например, 45+5-3-125+2 и нажмите <Enter> (пробелы и другие знаки недопустимы)
->9-5+4+2-6
Значение введенного выражения: 4
Для завершения нажмите <Enter>
```

Задача 198

```
// Вычисление значения арифметического выражения
#include <stdio.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>");
    getchar();
}
```

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 задуман-

ное число 523, то угаданы две цифры (5 и 2), но только одна из них (2) занимает верную позицию.

Далее приведен рекомендуемый вид экрана во время работы программы. Данные, введенные пользователем (игроком), 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 // включить 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("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>");
   getchar();
}
```

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 в следующей таблице.

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

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++]);
            }
      while ( sim[j] != ' ' && j <4 );
      }
     else
       if (mes[i] == ' ')  // пробел между словами
       {
          printf("  "); // пробел между словами сообщения
       }
  }
  puts("Для завершения работы нажмите <Enter>");
  getchar();
}
```

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 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 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 factorial 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 ноль в противном случае.

214. Написать функцию upcase, 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 len = 0;  // длина строки
    char* p = st;  // указатель на символ

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

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();
}
```

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 != 0)
  {
    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 знака минус не в первой позиции и более чем одной точки в составе числа. Функция должна позволять редактировать введенное число при помощи клавиши <Backspace>. При нажатии клавиши <Enter> функция должна завершать работу и возвращать введенное число.

Задача 220

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

// 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();
}
```

КЛАССЫ И ОБЪЕКТЫ

Общие замечания

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

- класс — это тип, определенный программистом;
- класс должен иметь, как минимум, один конструктор и один деструктор;
- хранение данных обеспечивают поля;
- доступ к данным объекта должны обеспечивать методы;
- объект — это экземпляр класса.

Задачи

221. Объявите структуру `empl` (от employee — сотрудник), состоящую из полей `name` и `salary`, предназначенную для хранения информации о зарплате сотрудников организации.

Задача 221

```
struct empl
{
    char name[15];
    double salary;
};
```

222. Напишите программу, которая формирует массив `staff`, состоящий из структур `empl` (см. *задачу 221*).

Задача 222

```
#include <string.h>

int main()
{
    struct empl
    {
        char name[15];
        double salary;
    };

    empl staf[5];  // сотрудники

    strcpy(staff[0].Name, "Homer Simpson");
    staff[0].salary = 5000;

    strcpy(staff[1].Name, "James Bond");
    staff[1].salary = 120000;

    printf("\nStaff\n");
    for (int i = 0; i < 5; i++)
    {
        if ( strlen( staf[i].name) != 0 )
          printf("%-15s %9.2f $\n",  staff[i].name,
                                staff[i].salary);
    }

    printf("\nTo exit press <Enter>");
    int ch = getchar();

    return 0;
}
```

223. Объявите класс `TPerson`, позволяющий создавать объекты, содержащие информацию о людях (имя, фамилия). Класс должен содержать конструктор, деструктор, а также свойство `FullName`, значением которого является полное имя (имя и фамилия как одна строка).

Задача 223

```
#include <string.h>
// объявление класса
class TPerson
{
    // поля
    char* fname;
    char* lname;

    char* fullname;

public:

    // конструктор
    TPerson(char* FirstName, char* LastName);

     // деструктор
    ~TPerson();

    // методы
    char* FullName()
    {
        if (fullname != NULL)  delete[] fullname;

        fullname = new char[strlen(fname)+strlen(lname)+2];
        sprintf(fullname, "%s %s", fname, lname);
        return fullname;
    };

};

// реализация конструктора
TPerson::TPerson(char* firstname, char* lastname)
{
    fname = new char[strlen(firstname)];
    lname = new char[strlen(lastname)];

    strcpy(fname, firstname);
    strcpy(lname, lastname);
```

```
    fullname = NULL;
}

// реализация деструктора
TPerson::~TPerson()
{
    delete[] fname;
    delete[] lname;
    delete[] fullname;
}
```

224. Измените объявление класса `TPerson` (см. *задачу 221*) так, чтобы он помимо имени и фамилии содержал адрес e-Mail.

Задача 224

```
#include <string.h>

class TPerson
{
    // поля
    char* fname;
    char* lname;
    char* email;

    char* fullname;

public:

    // конструктор
    TPerson(char* FirstName, char* LastName, char* EMail);

     // деструктор
    ~TPerson();

    // методы
    char* FullName()
    {
        if (fullname != NULL)  delete[] fullname;
```

```
        fullname = new char[strlen(fname)+strlen(lname)+2];
        sprintf(fullname, "%s %s", fname, lname);
        return fullname;
    };

    char* eMail()
    {
        return email;
    }

};

// реализация конструктора
TPerson::TPerson(char* firstname, char* lastname, char* mail)
{
    fname = new char[strlen(firstname)+1];
    lname = new char[strlen(lastname)+1];
    email = new char[strlen(mail)+1];

    strcpy(fname, firstname);
    strcpy(lname, lastname);
    strcpy(email, mail);

    fullname = NULL;
}

// деструктор
TPerson::~TPerson()
{
    delete[] fname;
    delete[] lname;
    delete[] email;
    delete[] fullname;
}
```

225. Напишите программу, которая для хранения информации о человеке использует класс TPerson.

Задача 225

```
#include <string.h>
#include <stdio.h>

//
//   Сюда поместить объявление класса TPerson
//

int main()
{
    TPerson *person;  // объект

    person = new TPerson("Bart", "Simpson");

    printf("%s\n", person->FullName());

    printf("\nTo exit press <Enter>");
    int ch = getchar();

        return 0;
}
```

226. Напишите программу, которая для хранения информации о людях использует массив объектов класса TPerson. Имена людей должны вводиться с клавиатуры во время работы программы.

Задача 226

```
#include <string.h>
#include <stdio.h>

//
//   Сюда поместить объявление класса TPerson
//

#define HB 10

int main()
{
    TPerson* persones[HB];
```

```
    // данные, введенные с клавиатуры
    char* fname = new char[20];
    char* lname = new char[20];
    char* email = new char[20];

    for (int i = 0; i < HB; i++)
    {
        printf("first name >>");
        scanf("%s", fname);

        printf("first name >>");
        scanf("%s", fname);

        printf("first name >>");
        scanf("%s", fname);

        persones[i] = new TPerson(fname, lname, email);
    }

    for (int i = 0; i < HB; i++)
    {
        printf("%s"\n", persones[i]->FullName());
    }

    // освободить память, занимаемую объектами
    for (int i = 0; i < HB; i++)
    {
        persones[i]->~TPerson();
    }

    printf("\nTo exit press <Enter>");
    int ch = getchar();

    return 0;
}
```

227. Объявите класс TPerson и производные от него классы TStud и TProf. Класс TPerson должен содержать общую информацию

о персоне. Классы TStud и TProf должны расширять информацию о персоне как о студенте и, соответственно, преподавателе.

Задача 227

```
class TPerson
{
    char *fname;
    char *lname;

public:
    TPerson() {} // чтобы иметь возможность создать массив объектов

    TPerson(char* FirstName, char* LastName)
    {
        fname = new char[strlen(FirstName) + 1];
        strcpy(fname, FirstName);
        lname = new char[strlen(LastName) + 1];
        strcpy(lname, LastName);
    }

    ~TPerson()
    {
        delete[] fname;
        delete[] lname;
    }

    char* FullName()
    {
        char* fullname =
             new char[strlen(fname) + strlen(lname) + 2];
        sprintf(fullname, "%s %s", fname, lname);
        return fullname;
    };

    virtual void print()  { }

};
```

```
// студент
class TStud : public TPerson
{
    int group;
public:
    TStud() {};

    TStud(char* FirstName, char* LastName, int Group):
                                TPerson(FirstName,LastName)
    {
        group = Group;
    }

    ~TStud()
    {

    }

    void print()     // выводит информацию о студенте
    {
        printf("%s, gr. %i\n", this->FullName(), group);
    }

};

// профессор
class TProf : public TPerson
{
     char* kafedra;

public:
    TProf() {};

    TProf(char* FirstName, char* LastName, char* Kafedra):
                                    TPerson(FirstName, LastName)
    {
        kafedra = new char[strlen(Kafedra) + 1];
        strcpy(kafedra, Kafedra);
    }
```

```
    ~TProf()
    {
        delete[] kafedra;
    }

    void  print()     // выводит информацию о профессоре
    {
        printf("%s, dep. %s\n", this->FullName(), kafedra);
    }

};
```

228. Напишите программу, которая хранит в одном массиве информацию о студентах и преподавателях (см. *задачу 227*).

Задача 228

```
#include <typeinfo>
#include <stdio.h>
#include <string.h>

//
// Сюда поместить объявление классов TPerson, TStud и TProf
//

int main()
{
    TPerson *pers[10]; //  студенты и преподаватели

    // Внимание!  Значение, возвращаемое sizeof,
    // зависит от разрядности платформы!

    int K = sizeof(pers) / sizeof(TPerson*);

    printf("Size of pers is %i\n", K);

    // инициализация массива ОБЯЗАТЕЛЬНА
    for (int i = 0; i < K; i++)
        pers[i] = NULL;
```

```
pers[0] = new TStud("Bart", "Simpson", 1358);
pers[1] = new TStud("Lisa", "Simpson", 1358);
pers[2] = new TProf("Homer", "Simpson",
                                "Nuclear Energy Systems");

//// Вывод списка. Вариант 1
//for (int i = 0; i < K; i++)
//{
//    if ( pers[i] != NULL)
//        pers[i]->print();
//}

// Вывод списка. Вариант 2
int i = 0;
while (pers[i] != NULL)
{
    pers[i]->print();
    i++;
}

// чтобы typeid стала доступной,
// надо добавить #include <typeinfo>

printf("\nStudents:\n");
i = 0;
while (pers[i] != NULL) {
    if (typeid(*pers[i]) == typeid(TStud)) {
        pers[i]->print();
    }
    i++;
}

//// освободить память
//i = 0;
//while (pers[i] != NULL) {
//    delete pers[i];
//    i++;
//}
```

```
    printf("\nTo exit press <Enter>");
    int ch = getchar();

    return 0;
}
```

229. Объявите структуру book для программы работы с информацией о книгах, например, в книжном магазине.

230. Напишите программу, формирующую массив структур book (см. *задачу 229*). Информация о книгах должна вводиться с клавиатуры во время работы программы.

231. Объявите класс TBook для программы работы с информацией о книгах, например, в книжном магазине.

232. Напишите программу, формирующую массив объектов TBook (см. *задачу 229*). Информация о книгах должна вводиться с клавиатуры во время работы программы

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 — функция fprintf;
- по завершении работы с файлом его нужно обязательно закрыть (функция fclose).

Задачи

233. Написать программу, которая создает на диске 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 233

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

#define FNAME "c:\\temp\\numbers.txt\0"
#define N 5                    // количество чисел

// Создает на диске файл и записывает в него
// целые числа, введенные пользователем
void main()
{
     char fname[40] = FNAME;
     FILE *f;     // файл чисел
     int n;       // число

     puts("\nСоздание файла");
     printf("Введенные 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();
}
```

234. 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 234

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

#define FNAME "c:\\temp\\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[40] = FNAME;
     FILE *f;      // файл чисел
     int n;        // число

     puts("\nДобавление в файл");
     printf("Введенные 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();
}
```

235. Написать программу, которая выводит на экран содержимое файла numbers.txt.

Задача 235

```
// Выводит на экран содержимое файла
#include <stdio.h>
#include <conio.h>
```

```
#define FNAME "c:\\temp\\numbers.txt\0" // имя файла

void main()
{
     char fname[40] = 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();
}
```

236. Написать программу, которая вычисляет среднее арифметическое чисел, находящихся в файле `numbers.txt`.

Задача 236

```
// Вычисляет среднее арифметическое чисел,
// находящихся в файлах
#include <stdio.h>
#include <conio.h>
```

```
#define FNAME "nc:\\temp\\numbers.txt\0" // имя файла

void main()
{
     char fname[40] = 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();
}
```

237. Написать программу, которая записывает в файл данные, находящиеся в двумерном массиве дробного типа.

Задача 237

```
// 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("c:\\temp\\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();
}
```

238. Написать программу, которая загружает из файла данные в двумерный массив дробного типа.

Задача 238

```
#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 элементов

FILE *f; // файл

// предполагается, что строки текстового файла
// содержат значения строк элементов массива

 // Открыть для чтения (r) текстовый файл (t)
 if ((f = fopen("c:\\temp\\a.dat", "rt")) == NULL)
 {
    printf("Нет файла c:\\temp\\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();
}
```

239. Написать программу, которая позволяет просматривать текстовые файлы (выводит на экран их содержимое) — например, файлы исходных программ C/C++. Имя просматриваемого файла должно вводиться пользователем во время работы программы.

Задача 239

```
// Выводит на экран содержимое файла,
// имя которого указано пользователем
#include <stdio.h>
#include <conio.h>
#include <string.h>

#define  MAXLEN 80    // максимальная длина строки в файле
void main()
{
    char fname[40];   // имя файла
    FILE *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;
    }
```

```
    while ( !feof(f) )
    {
        fgets(st, MAXLEN, f);
        printf("%s", st);
        if (++n > 21)
        {
          printf("\nДля продолжения нажмите <Enter>");
          key = getch();
          n = 0;
        }
    }
    fclose(f);      // закрыть файл

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

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

```
Добавление информации в телефонный справочник
Фамилия -> Сидоров
Имя -> Вася
Телефон -> 234-84-37

Информация добавлена
Для завершения нажмите <Enter>
```

Задача 240

```
// Дописывает в файл contacts.txt фамилию, имя и номер телефона
#include <stdio.h>
#include <conio.h>

#define FNAME "c:\\temp\\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);

    // запишем данные 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();
}
```

241. Усовершенствуйте программу работы с телефонным справочником (см. *задачу 240*) так, чтобы за один сеанс работы в файл

contacts.txt можно было добавить информацию о нескольких людях. Далее приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным).

```
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>
```

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

```
Поиск в телефонном справочнике

Фамилия -> Петров
Данных об абоненте Петров в БД нет.
Фамилия -> Иванов
Иванов Вася 578-12-45
Иванов Сергей 244-34-02
Фамилия ->
```

Задача 242

```
// Поиск в телефонном справочнике
#include <stdio.h>
#include <conio.h>
```

```
#define FNAME "c:\\temp\\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();
}
```

243. Написать программу, которая объединяет возможности программ «Добавление информации в телефонный справочник» (см. *задачу 241*) и «Поиск в телефонном справочнике» (см. *задачу 242*). При запуске программы на экране должно отображаться меню следующего вида:

```
*** Телефонный справочник ***
1 - Добавление
2 - Поиск
0 - Завершение работы

Ваш выбор ->
```

244. Написать программу, при помощи которой можно автоматизировать процесс проверки знаний. Сам тест, содержащий последовательность вопросов и вариантов ответа, должен находиться в текстовом файле. Имя файла теста программа должна получать из командной 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.

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

Далее приведены структура файла теста (N_i — количество вариантов ответа к i-му вопросу, K_i — номер правильного ответа) и пример файла теста.

Структура файла теста

```
Вопрос1
N1 K1
Ответ1
…
ОтветN1
Вопрос2
N2 K2
Ответ1
…
ОтветK2

ВопросM
NM KM
```

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);
```

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

```
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 далее инструкций содержит ошибку
1 - scanf ("%f", &kurs);
2 - scanf("%f", kurs);
Ваш выбор-> 2

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

Задача 244

```
// 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 процент правильных ответов
  p = 100 * PravOtv / VsegoVopr;
  printf("\nВаша оценка - ");
  if (p == 100) puts("ОТЛИЧНО!");
  if (p >= 99 && p <= 80) puts("ХОРОШО.");
  if (p >= 60 && p <= 79) puts("УДОВЛЕТВОРИТЕЛЬНО.");
  if (p < 60) puts("ПЛОХО!\n");

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

РЕКУРСИЯ

Общие замечания

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

- рекурсивной называется такая функция, которая вызывает сама себя;
- для окончания процесса рекурсии в алгоритме рекурсивной функции обязательно должна быть ветвь, обеспечивающая завершение функции (выход из функции).

Задачи

245. Написать рекурсивную функцию вычисления факториала и программу, проверяющую ее работу.

Задача 245

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

246. Написать рекурсивную функцию вычисления числа Фибоначчи и программу, проверяющую ее работу.

Задача 246

```
#include <stdio.h>
#include <conio.h>

int Fibonacci(int n)
{
```

```
    if ((n == 0) || (n == 1))
        return 1;
    else
        return Fibonacci(n - 2) + Fibonacci(n - 1);
}

int main()
{
    printf("\nNumeri di Fibonacci\n");
    for (int n = 0; n < 21; n++)
    {
        printf("%i ", Fibonacci(n));
    }

    printf("\n\nPress <Enter>");
    int ch = getchar();
        return 0;
}
```

247. Программа демонстрирует применение рекурсии для решения задачи поиска — в частности, поиска пути на графе. Для представления графа используется массив.

Задача 247

```
#include <stdio.h>

#define N 7  // количество узлов графа

// карта
int map[N][N] = { {0,1,1,1,0,0,0},
                  {1,0,0,0,0,0,0},
                  {1,0,0,1,0,0,1},
                  {1,0,1,0,0,1,0},
                  {0,0,0,0,0,1,1},
                  {0,0,0,1,1,0,1},
                  {0,0,1,0,1,1,0} };
// путь;
int road[N] = {-1,-1,-1,-1,-1,-1,-1};
```

```
int inRoad[N] = {0,0,0,0,0,0,0,0}; // inRoad[k] == 1, если
                                   // точка k включена в маршрут
// выбор очередной точки
void step(int s, int f, int p)
{
    if (s == f) {
        printf("\nPath:");
        for (int i = 0; i < N; i++)
            if ( road[i] != -1)
                 printf("%i ", road[i]);
        printf("\n");
    }

    else
        for (int c = 0; c < N ; c++)
        {
            if ((map[s][c] != 0) && (inRoad[c] == 0))
            {
                // включить в путь и сделать шаг
                road[p] = c;
                inRoad[c] = 1;
                step(c, f, p + 1);

                // чтобы найти все возможные варианты, надо
                // исключить из пути и проверить следующий узел
                road[p] = -1;
                inRoad[c] = 0;
            }
        }
}

int main()
{
    int start = 2;
    int finish = 6;

    // внести стартовую точку в маршрут
    road[0] = start;
    inRoad[start] = 1;
```

```
    printf("From %i to %i\nSearch...\n", start, finish);

    // поиск маршрута
    step(start, finish, 1);   // шаг из стартовой точки

    printf("\nTo exit press <Enter>");
    int ch = getchar();

    return 0;
}
```

Часть 2

Справочник

СТРУКТУРА ПРОГРАММЫ

Программа на языке C/C++ представляет собой набор функций, одна из которых имеет имя `main`.

В простейшем случае программа представляет собой одну 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

Вещественные 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 присваивания в общем виде записывается так:

```
переменная = выражение
```

Некоторые действия, выполняемые с использованием инструкции присваивания, можно записать при помощи операторов инкремента, декремента, увеличения, уменьшения, операторов «умножить на» и «разделить на».

«Обычная» инструкция присваивания	Альтернативная инструкция
x = x + 1	x++
x = x - 1	x--
x = x + y	x += y
x = x - y	x -= y
x = x*y	x *= y
x = x % y	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

Цикл *for*

Синтаксис:

```
for ( Инициализация; Условие_Выполнения; Изменение )
{
    // Здесь инструкции (тело цикла)
}
```

- *Инициализация* — инструкция инициализации счетчика циклов.
- *Условие_Выполнения* — выражение, значение которого определяет условие выполнения инструкций цикла. Инструкции цикла выполняются до тех пор, пока *Условие_Выполнения* истинно (не равно нулю).

- *Изменение* — инструкция изменения параметра цикла. Как правило, эта инструкция изменяет значение переменной, которая входит в *Условие_Выполнения*.

Цикл *do ... while*

Синтаксис:

```
do
{
    // Инструкции (тело цикла)
}
while ( Условие_Повторения );
```

Сначала выполняются инструкции цикла (тело цикла), затем проверяется значение выражения *Условие_Повторения*, и если условие истинно (не равно нулю), то инструкции цикла выполняются еще раз. И так до тех пор, пока *Условие_Повторения* не станет ложным, т. е. равным нулю.

Цикл *while*

Синтаксис:

```
while ( Условие_Выполнения )
{
    // Инструкции (тело цикла)
}
```

Сначала проверяется значение выражения *Условие_Выполнения*. Если оно истинно (не равно нулю), то выполняются инструкции цикла (тело цикла). Затем снова проверяется значение выражения *Условие_Выполнения*, и если оно истинно (не равно нулю), инструкции цикла выполняются еще раз. И так до тех пор, пока значение выражения *Условие_Выполнения* 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 *Consolas*;
- 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
%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.

> **Примечание**
>
> В качестве параметра функции scanf должны передаваться *адреса* переменных, а не их имена.

Спецификаторы формата

Спецификатор	Вводит
%i %d	Десятичное число со знаком
%u	Беззнаковое целое десятичное число
%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, чтобы определить, какая служебная клавиша нажата, нужно обратиться к функции `getch` еще раз.

> ***Примечание***
>
> 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`.

Примечание

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. Список задает переменные, значения которых должны быть выведены.
- Параметр *Формат* задает способ отображения значений переменных.

Действие функции sprintf аналогично действию функции printf, но вывод выполняется в строку-буфер, а не на экран.

Заголовочный файл: <stdio.h>

ФУНКЦИИ РАБОТЫ С ФАЙЛАМИ

fopen

Синтаксис:

```
FILE* fopen(const char * Имя, const char* Режим)
```

Открывает файл с указанным именем для действия, которое задается параметром *Режим*.

Режимы

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 режим, при котором 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, СписокПеременных);
```

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);
```

Возвращают 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);
```

Возвращают 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);
```

Возвращают 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 expl(long double (x));
```

Возвращают 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));
```

Возвращают 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`.

Заголовочный файл: `<stdlib.h>`

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);
```

Возвращают 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:

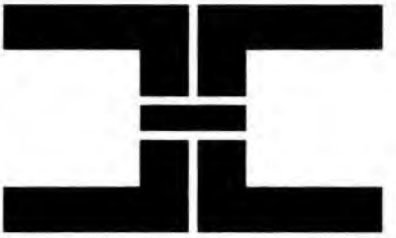
```
char* itoa (int Значение, char* Строка, int Основание);
char* ltoa (long Значение, char* Строка, int Основание);
char* ultoa(unsigned long Значение, char* Строка, int
Основание);
```

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 символов, функциями `ltoa` и `ultoa` — 33 символа.

Заголовочный файл: `<stdlib.h>`



ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ 1

Таблица ASCII кодировки символов

Символы с кодами 0—127

0-	16- ►	32-	48- 0	64- @	80- P	96- `	112- p
1- ☺	17- ◄	33- !	49- 1	65- A	81- Q	97- a	113- q
2- ☻	18- ↕	34- "	50- 2	66- B	82- R	98- b	114- r
3- ♥	19- ‼	35- #	51- 3	67- C	83- S	99- c	115- s
4- ♦	20- ¶	36- $	52- 4	68- D	84- T	100- d	116- t
5- ♣	21- §	37- %	53- 5	69- E	85- U	101- e	117- u
6- ♠	22- ▬	38- &	54- 6	70- F	86- V	102- f	118- v
7-	23- ↨	39- '	55- 7	71- G	87- W	103- g	119- w
8-	24- ↑	40- (	56- 8	72- H	88- X	104- h	120- x
9-	25- ↓	41-)	57- 9	73- I	89- Y	105- i	121- y
10-	26- →	42- *	58- :	74- J	90- Z	106- j	122- z
11-	27- ←	43- +	59- ;	75- K	91- [	107- k	123- {
12-	28- ∟	44- ,	60- <	76- L	92- \	108- l	124- |
13-	29- ↔	45- -	61- =	77- M	93-]	109- m	125- }
14- ♫	30- ▲	46- .	62- >	78- N	94- ^	110- n	126- ~
15- ☼	31- ▼	47- /	63- ?	79- O	95- _	111- o	127- ⌂
16- ►	32-	48- 0	64- @	80- P	96- `	112- p	

Символы с кодами 128—255

128- А	144- Р	160- а	176- ▒	192- └	208- ╨	224- р	240- Ё
129- Б	145- С	161- б	177- ▓	193- ┴	209- ╤	225- с	241- ё
130- В	146- Т	162- в	178- █	194- ┬	210- ╥	226- т	242- Є
131- Г	147- У	163- г	179- │	195- ├	211- ╙	227- у	243- є
132- Д	148- Ф	164- д	180- ┤	196- ─	212- ╘	228- ф	244- Ї
133- Е	149- Х	165- е	181- ╡	197- ┼	213- ╒	229- х	245- ї
134- Ж	150- Ц	166- ж	182- ╢	198- ╞	214- ╓	230- ц	246- Ў
135- З	151- Ч	167- з	183- ╖	199- ╟	215- ╫	231- ч	247- ў
136- И	152- Ш	168- и	184- ╕	200- ╚	216- ╪	232- ш	248- °
137- Й	153- Щ	169- й	185- ╣	201- ╔	217- ┘	233- щ	249- ∙
138- К	154- Ъ	170- к	186- ║	202- ╩	218- ┌	234- ъ	250- ·
139- Л	155- Ы	171- л	187- ╗	203- ╦	219- ▒	235- ы	251- √
140- М	156- Ь	172- м	188- ╝	204- ╠	220- ▄	236- ь	252- №
141- Н	157- Э	173- н	189- ╜	205- ═	221- ▌	237- э	253- ¤
142- О	158- Ю	174- о	190- ╛	206- ╬	222- ▐	238- ю	254- ■
143- П	159- Я	175- п	191- ┐	207- ╧	223- ▀	239- я	255-
144- Р	160- а	176- ▒	192- └	208- ╨	224- р	240- Ё	256-

ПРИЛОЖЕНИЕ 2

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. П2.1).

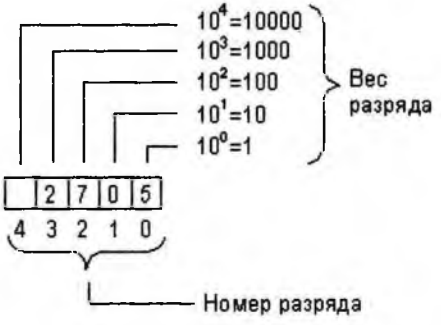
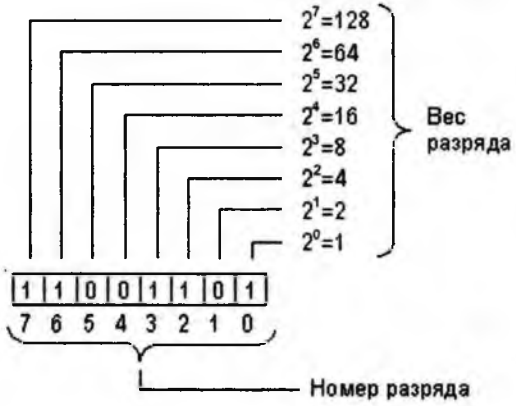


Рис. П2.1. Вес разрядов в десятичной системе счисления

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. П2.2).



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. П2.2. 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. П2.3).

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. П2.1 приведены десятичные числа от нуля до 15 и соответствующие им шестнадцатеричные цифры.

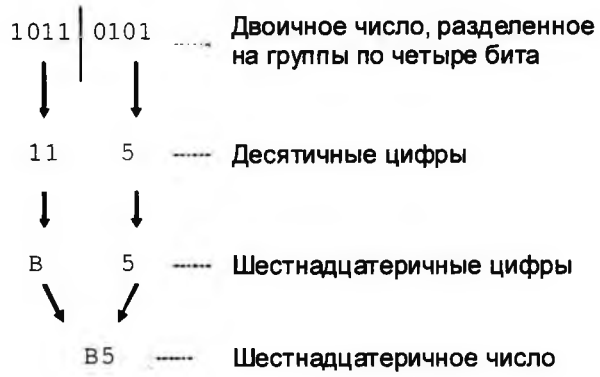


Рис. П2.3. Перевод двоичного числа в шестнадцатеричное

Таблица П2.1. *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

Таблица П2.1 *(окончание)*

Десятичное число	Шестнадцатеричная цифра
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`.

Предметный указатель

К

М

Н

П

Р

С

Т

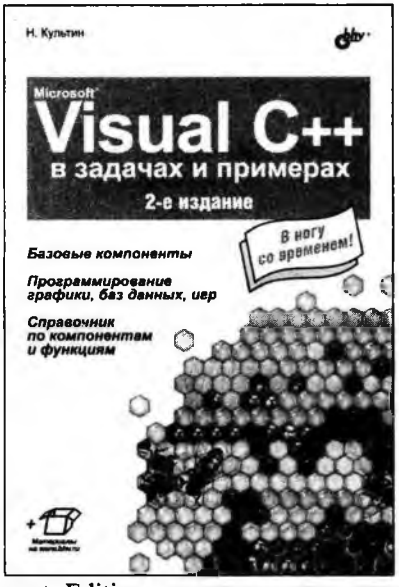
Ф

Ц

www.bhv.ru

Отдел оптовых поставок
E-mail: opt@bhv.spb.su

Культин Н.
Microsoft Visual C++ в задачах и примерах, 2-е изд.



- Базовые компоненты
- Программирование графики, баз данных, игр
- Справочник по компонентам и функциям

Если вы хотите научиться программировать в Microsoft Visual C++, то эта книга для вас.

В ней вы найдете хорошо документированные примеры программ — от простейших, демонстрирующих назначение базовых компонентов, до приложений работы с файлами, графикой и базами данных Microsoft Access и Microsoft SQL Server Compact Edition, раскрывающих тонкости программирования в Microsoft Visual C++. Предлагаемые для самостоятельного решения задачи позволят закрепить полученные знания. Справочник, входящий в книгу, содержит описание базовых компонентов, событий, исключений и наиболее часто используемых функций. На FTP-сервере издательства находятся коды примеров из книги.

Никита Борисович Культин, кандидат технических наук, доцент Санкт-Петербургского государственного политехнического университета, где читает курс «Теория и технология программирования». Автор серии книг по программированию в Microsoft Visual C++, Microsoft Visual C#, Microsoft Visual Basic, Delphi, Pascal, которые вышли общим тиражом более 350 тыс. экземпляров.

Полубенцева М.

С/С++. Процедурное программирование

www.bhv.ru
Отдел оптовых поставок
E-mail: opt@bhv.spb.su

«Программировать — значит понимать»

Кристиан Нюгард (Kristen Nygaard), норвежский математик, один из пионеров в разработке языков программирования

- Пространства имен
- Препроцессор
- Указатели и массивы
- Динамическое выделение памяти
- Ссылки
- Функции
- Структуры и объединения

«…В современных книгах незаслуженно мало внимания уделяется процедурным возможностям языков С/С++. Книга призвана заполнить этот пробел. В ней рассмотрены как основные понятия процедурного программирования на С/С++, так и взаимосвязи между ними. В результате по мере прочтения в голове у читателя формируется модель внутренней структуры изучаемого предмета, и после этого существенно облегчается восприятие последующего материала, поскольку начинает активно работать ассоциативная память.

Начинающий программист найдет в книге структурированный материал «для старта», а программист, уже имеющий опыт программирования на С/С++, откроет для себя возможности языка, которыми он до сих пор не пользовался (или пользовался неосознанно). Книга будет полезна разработчикам программного обеспечения для встраиваемых систем, поскольку в настоящее время большинство программ для микроконтроллеров пишут на языке С.

Для понимания правил, по которым действует компилятор, приводятся пояснения на уровне языка Ассемблера…»

М. И. Полубенцева, старший преподаватель компьютерных дисциплин СПбГПУ

Полубенцева Марина Игоревна, старший преподаватель компьютерных дисциплин в Санкт-Петербургском государственном политехническом университете (СПбГПУ). Имеет многолетний преподавательский и практический опыт в области разработки приложений на С++. Читает курсы по процедурному программированию на С/С++, объектно-ориентированному программированию на С++, Windows-программированию, системному программированию в ОС Windows.

www.bhv.ru

Шлее М.

Qt 5.10. Профессиональное программирование на C++

Отдел оптовых поставок
E-mail: opt@bhv.spb.su

Платформно-независимая реализация приложений — это уже сегодняшний и завтрашний день программной индустрии. И эта книга станет вашим путеводителем в будущее.

- Кроссплатформенная реализация приложений для Windows, Mac OS X и Linux
- Разработка мобильных приложений для Android и iOS
- Программирование 2D- и 3D-графики, мультимедиа, веб-приложений, баз данных, сети, таймера, многопоточности, XML, QML и JavaScript
- 240 завершенных программ

Книга подробно знакомит с библиотекой Qt 5.10, являющейся не только средством для создания пользовательских интерфейсов, но и позволяющей разрабатывать приложения практически любой сложности. Недаром Qt широко используется многими организациями и компаниями, такими как ADOBE, AMAZON, AMD, BOSCH, BLACKBERRY, BMW, CANON, CISCO SYSTEMS, DISNEY, INTEL, IBM, PANASONIC, PARALLELS, PIONEER, PHILIPS, ORACLE, HP, GOOBER, GOOGLE, MERCEDES, NASA, NEC, NEONWAY, RAKUTEN, SAMSUNG, SIEMENS, SONY, SUN, TESLA, XEROX, XILINX, YAMAHA и др.

Если вы хотите идти в ногу со временем, то вам без этой книги просто не обойтись, поскольку она является исчерпывающим пособием по написанию Qt 5-программ на C++ и QML.

Макс Шлее (Max Schlee) — закончил Университет прикладных наук в городе Кайзерлаутерн (Германия). Сооснователь компании NEONWAY по разработке программного обеспечения на Qt. Работал разработчиком программного обеспечения в фирмах THOMSON, Grass Valley, DigitalFilmTechnology Weiterstadt, Goober Ltd. и Advancis. Эксперт в области объектно-ориентированного проектирования, специализирующийся на C++ и Qt. Создатель более 50 программ и приложений для Windows, Mac OS X, iOS и Android. Увлеченно занимается проектами в области программирования графики, звука и анализа финансовых рынков. Является автором ряда статей, научных докладов на международных конференциях по генеративному программированию пользовательского интерфейса. Автор книг «Qt 3/4/4.5/4.8/5.3. Профессиональное программирование на C++» и др.

www.bhv.ru

Культин Н.

Microsoft Visual C# в задачах и примерах, 2-е изд.

Отдел оптовых поставок
E-mail: opt@bhv.spb.su

В ногу со временем!

- Базовые компоненты
- Программирование графики и баз данных
- Работа с LINQ
- Справочник по компонентам и функциям

Если вы хотите научиться программировать в Microsoft Visual C#, то эта книга для вас. В ней вы найдете хорошо документированные примеры программ — от простейших, демонстрирующих назначение базовых компонентов, до приложений работы с файлами, XML-документами, графикой и базами данных Microsoft Access и Microsoft SQL Server Compact, раскрывающих тонкости программирования в Microsoft Visual C#. Справочник, входящий в книгу, содержит описание базовых компонентов, событий, исключений и наиболее часто используемых функций. На FTP-сервере издательства находятся коды примеров из книги. Приведены примеры использования технологии LINQ. Предлагаемые для самостоятельного решения задачи позволят закрепить полученные знания.

Никита Борисович Культин, кандидат технических наук, доцент Санкт-Петербургского государственного политехнического университета, где читает курс «Теория и технология программирования». Автор серии книг по программированию в Microsoft Visual C++, Microsoft Visual C#, Microsoft Visual Basic, Delphi, Pascal, которые вышли общим тиражом более 350 тыс. экземпляров.

www.bhv.ru

Фленов М.

Библия C#, 3-е изд.

Отдел оптовых поставок
E-mail: opt@bhv.spb.su

Программирование может быть доступно каждому!

- Программирование для .NET на C#
- Базы данных
- Графика и мультимедиа
- Сетевое программирование
- Повторное использование кода
- Изучение языка на полезных примерах

Книга будет полезна всем, кто хочет научиться писать программы для платформы Microsoft .NET на современном и удобном языке программирования C#. Большое количество практических примеров, легкость изложения материала и интересные комментарии призваны сделать обучение занимательным и нескучным, а подробное описание логики выполнения каждого участка кода поможет читателю использовать полученные знания при разработке собственных приложений.

Дополнительным источником знаний в процессе дальнейшего обучения служит электронный архив к книге, содержащий справочную информацию и статьи, а также готовые компоненты и изображения.

Флёнов Михаил, профессиональный программист. Работал в журнале «Хакер», в котором несколько лет вел рубрики «Hack-FAQ» и «Кодинг» для программистов, печатался в журналах «Игромания» и «Chip-Россия». Автор бестселлеров «Библия Delphi», «Программирование в Delphi глазами хакера», «Программирование на C++ глазами хакера», «Web-сервер глазами хакера», «Компьютер глазами хакера» и др. Некоторые книги переведены на иностранные языки и популярны в США, Канаде, Польше и других странах.

www.bhv.ru

Отдел оптовых поставок
E-mail: opt@bhv.spb.su

Прохоренок Н.

Основы Java

Просто о сложном

- Базовый синтаксис языка Java
- Объектно-ориентированное программирование
- Работа с файлами и каталогами
- Stream API
- Функциональные интерфейсы
- Лямбда-выражения
- Работа с базой данных MySQL
- Получение данных из Интернета
- Примеры и советы из практики

Если вы хотите научиться программировать на языке Java, то эта книга для вас. В книге описан базовый синтаксис языка Java: типы данных, операторы, условия, циклы, регулярные выражения, объектно-ориентированное программирование. Рассмотрены основные классы стандартной библиотеки, получение данных из сети Интернет, работа с базой данных MySQL. Приводится описание большинства нововведений: Date API, Stream API, лямбда-выражения, ссылки на методы, функциональные интерфейсы и др.

Книга содержит большое количество практических примеров, помогающих начать программировать на языке Java самостоятельно. Весь материал тщательно подобран, хорошо структурирован и компактно изложен, что позволяет использовать книгу как удобный справочник.

Прохоренок Николай Анатольевич, профессиональный программист, автор книг «HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера», «Python 3. Самое необходимое», «Python 3 и PyQt 5. Разработка приложений», «Разработка Web-сайтов с помощью Perl и MySQL» и др.