Тема 8. ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ФУНКЦИЙ

Цель занятия: овладеть навыками алгоритмизации и программирования задач с использованием функций пользователя; получить практические навыки написания функций и обращения к ним, выбора параметров функций.

Теоретические сведения

С увеличением объема программы становится невозможно удерживать в памяти все детали. Чтобы уменьшить сложность программы, ее разбивают на части. В С++ задача может быть разделена на более простые подзадачи с помощью функций. Разделение задачи на функции также позволяет избежать избыточности кода, т.к. функцию записывают один раз, а вызывают многократно. Программу, которая содержит функции, легче отлаживать.

Часто используемые функции можно помещать в библиотеки. Таким образом, создаются более простые в отладке и сопровождении программы.

Функция – это именованная последовательность описаний и операторов, выполняющая законченное действие, например, формирование массива, печать массива и т.д. (рис. 1).

Функция, во-первых, является одним из производных типов С++, а, во-вторых, минимальным исполняемым модулем программы.

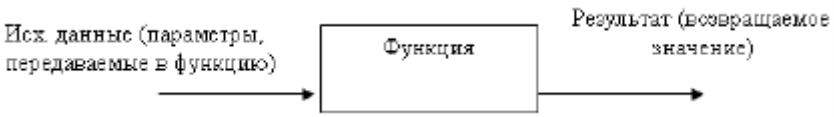

Рис. 1. Функция

Любая функция должна быть *объявлена и определена*.

Объявление функции (прототип, заголовок) задает имя функции, тип возвращаемого значения и список передаваемых параметров.

Определение функции содержит, кроме объявления, тело функции, которое представляет собой последовательность описаний и операторов. Общая форма определения функции:

тип имя_функции([список_формальных_параметров])

{ тело_функции}

Тело_функции – это блок или составной оператор. Внутри функции нельзя определить другую функцию.

В теле функции должен быть оператор, который возвращает полученное значение функции в точку вызова. Он может иметь формы:

1) *return выражение;*

2) *return*.

Форма 1) используется для возврата результата, поэтому выражение должно иметь тот же тип, что и тип функции в определении. Форма 2) используется, если функция не возвращает значения, т.е. имеет тип *void*. Программист может не использовать этот оператор в теле функции явно, компилятор добавит его автоматически в конец функции перед знаком «}».

Тип возвращаемого значения может быть любым, кроме ***массива и функции***, но может быть указателем на массив или функцию.

Список формальных параметров – это те величины, которые требуется передать в функцию. Элементы списка разделяются запятыми. Для каждого параметра указывается тип и имя. В объявлении имена можно не указывать.

Для того, чтобы выполнялись операторы, записанные в теле функции, функцию необходимо вызвать. При вызове указываются: *имя функции и фактические параметры*. Фактические параметры заменяют формальные параметры при выполнении операторов тела функции.

Фактические и формальные параметры должны совпадать по количеству и типу.

Объявление функции должно находиться в тексте раньше вызова функции, чтобы компилятор мог осуществить проверку правильности вызова. Если функция имеет тип не *void*, то ее вызов может быть операндом выражения.

Пример 1.1. Рассмотрим процедуру Cena, которая определяет стоимость телефонного разговора с поминутной оплатой 0.6 гр. + 20% НДС.

```
double  Cena (int k )
    {
      double c = k * 0.6;;
      return    c = c + 0.2 * c;
```

```
}
```

В приведенном примере k – формальный параметр-значение (это количество минут разговора), c – сумма, которую необходимо уплатить за k минут разговора, возвращаемое значение.

Задача 1.1. Составить программу для вычисления $z = \frac{sh^2 a + sh(a-b)}{sha + \sqrt{sh(a^2 - b^2)}}$, используя функцию $shx = \frac{e^x - e^{-x}}{2}$.

Программа имеет вид:

```
#include <iostream>
#include <math.h>
#include <conio.h>
using namespace std;
double sh(double x)
{
//функция возвращает sh(x)
return (exp(x)-exp(-x))/2.0;
}
void main()
{
        double a, b, z, t1, t2, t3;
        cout<<"\nEnter a, b:";
        cin>>a>>b;
        t1=sh(a);
        t2=sh(a-b);
        t3=sh(a*a-b*b);
        z = (t1*t1+t2) / (t1+sqrt(t3));
        cout<<"\nz = "<< z;
        getch();
}
```

Не вводя дополнительных переменных, вызов функции можно выполнить в операторе вывода. Тогда main() будет иметь вид:

```
void main()
{
        double a, b;
        cout<<"\nEnter a, b:";
        cin>>a>>b;
        cout<<"\nz = "<< (sh(a)*sh(a)+sh(a-b)) / (sh(a)+sqrt(sh(a*a-b*b)));
        getch();
}
```

Задание 1.1. Составить программу для вычисления значения функции $s = \sqrt{x^2 + y^2 + \sin^2 xy} + \sqrt{y^2 + z^2 + \sin^2 yz} + \sqrt{z^2 + x^2 + \sin^2 zx}$.

Основным способом обмена информацией между вызываемой и вызывающей функциями является механизм параметров. Существует два способа ***передачи параметров в функцию: по адресу и по значению***.

При использовании массива как параметра функции в функцию передается указатель на его первый элемент (*передача по адресу*), т.е. массив может быть изменен за счет операторов тела функции. Размерность массива следует передавать как отдельный параметр, т.к. информация о количестве элементов в массиве теряется.

Задача 1.2. Переписать в новый массив отрицательные элементы исходного массива.

```
#include <iostream>
#include <conio.h>
#include <stdlib.h>
using namespace std;
```

```
//функция формирования массива
int form(int a[100])
{
	int n;
	cout<<"\nEnter n";
	cin>>n;
	for(int i=0;i<n;i++)
	a[i]=rand()%100-50;
	return n;		//функция возвращает количество элементов в массиве
}
//печать массива
void print(int a[100],int n)
{
	for(int i=0;i<n;i++)
	cout<<a[i]<<" ";
	cout<<"\n";
}
//переписывание в новый массив отрицательные элементы исходного массива
void Neg(int a[100],int n,int b[100],int&k)
{
	k=0;	//количество элементов в новом массиве
	for(int i=0;i<n;i++)
	if(a[i]<0)  b[k++]=a[i];
}
void main()
{
	int mas[100],mas_new[100];
	int n,kol;
	n=form(mas);
	print(mas,n);
	Neg(mas,n,mas_new,kol);
	print(mas_new,kol);
	getch();
}
```

Задание 1.2. Написать функцию, которая проверяет упорядочен ли массив по возрастанию.

При передаче многомерных массивов в функцию размерности должны либо передаваться в качестве параметров, либо заранее определены как константы.

Задача 1.3. Элементы матрицы с одинаковыми индексами увеличить в два раза, у остальных элементов поменять знак на противоположный.

```
#include <iostream>
#include <conio.h>
#include <windows.h>
#include <iomanip>		// файл, где определена функция setw(4)
using namespace std;
const int N=4;	//глобальная переменная
//вывод матрицы
void print(int a[][N],int n)
{
	for(int i=0;i<n;i++)
	{
		for(int j=0;j<N;j++)
		cout<<setw(4)<<a[i][j];
		cout<<"\n";
	}
	cout<<"\n";
}
//ввод матрицы
void form(int a[][N],int n)
{
```

```
	cout<<"ВВОДИТЕ МАТРИЦУ\n";
	for(int i=0;i<n;i++)
	for(int j=0;j<N;j++)
	cin>>a[i][j];
	cout<<"\n";
}
//изменение матрицы
void transp(int a[][N],int n)
{
	for(int i=0;i<n;i++)
	for(int j=0;j<N;j++)
	if(i==j) a[i][j]=2*a[i][j];
	else a[i][j]=-a[i][j];
}
void main()
{
	SetConsoleOutputCP(1251);
	const int k=2;
	int mas[k][N];
	form(mas,k);
	cout<<"ИСХОДНАЯ МАТРИЦА\n";
	print(mas,k);
	cout<<"НОВАЯ МАТРИЦА\n";
	transp(mas,k);
	print(mas,k);
	getch();
}
```

Задание 1.3. Поменять местами максимальный и минимальный элементы матрицы.

Строки при передаче в функции могут передаваться как одномерные массивы типа *char* или как указатели типа *char**. В отличие от обычных массивов в функции не указывается длина строки, т.к. в конце строки есть признак конца строки /0.

Задача 1.4. Функция поиска заданного символа в строке

```
#include <iostream>
#include <conio.h>
#include <windows.h>
using namespace std;
int find (char *s,char c)
{
	for (int I=0;I<strlen(s);I++)
	if(s[I]==c) return (I);
	return (-1);
}
/*С помощью функции find подсчитаем количество различных гласных букв в строке.*/
void main()
{
	SetConsoleOutputCP(1251);
	char s[255];
	int k=0;
	cout<<"ВВЕДИТЕ СТРОКУ";
	cin>>s;
	char*gl="aouiey";
	for(int I=0;I<strlen(gl);I++)
	if(find(s,gl[I])>=0)k++;
	cout<<k;
	getch();
}
```

Задание 1.4. Подсчитать количество цифр в строке.

Задача 1.5. В заданном количестве строк во всех словах поменять местами первый и последний символы.

```
#include <iostream>
#include <string>
#include <clocale>
#include <conio.h>
using namespace std;
string  Neg(string st)
{
        int i = 0, begin=0,end;
        st=st+" ";
        while (i < st.size()-1)
        {
                while (st[i] == ' ' && i < st.size()-1)
                        ++i;                            //пропустить пробелы
                begin = i;                              // номер первого символа слова
                while ( st[i] != ' ' && i < st.size()-1)
                        ++i;                    // пропустить  символы слова
                end = i;     // номер символа,  следующего за последним символом слова
                //поменять местами первый и последний символы слова
                char c=st[begin];
                st[begin]=st[end-1];
                st[end-1] =c;
        }
         return st;
}
int main()
{
        setlocale (LC_CTYPE,"rus");

        int k;
        cout<<"Введите количество строк"<<"\n";
        cin>>k;
        string str;
        for (int j = 0;j<k; j++)
        {
                cout<<"Введите строку "<<j+1<<"\n";
                getline(cin,str,'*');           // ввод текста
                cout<<Neg(str)<<"\n";
        }
        getch();            // ждать нажатия любой клавиши
        return 0;
}
```

Переменные, которые используются внутри данной функции, называются ***локальными***.

Глобальные переменные – это переменные, описанные вне функций. Они видны во всех функциях, где нет локальных переменных с такими именами.

Пример 1.2.

```
int a,b;     //глобальные переменные
void change()
{
  int r;                //локальная переменная
  r=a; a=b; b=r;
}
void main()
{
  cin>>a,b;
  change();
  cout<<"a="<<a<<"b="<<b;
}
```

Методические указания

При подготовке к занятию необходимо изучить: правила записи функций и способы обращения к ней; способы передачи параметров в функцию; правила записи и порядок выполнения программ, использующих функции.

Аудиторные и домашние задания

1. Составить программу для вычисления значений функции $a=\frac{\sqrt{sz^3+qz^2+tz+t}}{1+e^{sz^3+z-1}};\ b=\frac{t^2\sin\alpha+t\cos\beta+3{,}5}{z^2+2r-t},$.

2. Составить программу для вычисления значения $c=\frac{n!}{m!(n-m)!}$, используя функцию.

3. Составить программу для вычисления среднего арифметического положительных элементов массивов **X**(60), **Y**(75), **Z**(80), используя функцию. В массивах имеются положительные элементы.

4. Составить программу для вычисления среднего геометрического положительных элементов каждого столбца матрицы **A**(8,10), используя функцию.

5. Составить программу для вычисления $z=\frac{e^{|x_{\max}|}-e^{|y_{\max}|}}{\sqrt{|x_{\min}y_{\min}|}}$, где $x_{\max}$ и $x_{\min}$ – наибольший и наименьший элементы массива X(100), $y_{\max}$ и $y_{\min}$ – наибольший и наименьший элементы массива **Y**(80).

6. Составить программу для вычисления значения функции $u=e^{x_1+y_1}-e^{x_2-y_2}$, где x_1, x_2 – корни уравнения $ax^2+bx-1{,}5=0$; y_1, y_2 – корни уравнения $2y^2-y+c=0$.
Корни находить в функции. Если корни мнимые, то считать их равными нулю.

7. Составить программу для вычисления значения $z=x_1+x_2+x_3$, где $x_1=\frac{\sum_{i=1}^{10}a_{2i+1}}{21!}, x_2=\frac{\sum_{i=1}^{20}b_{2i}}{40!}, x_3=\frac{\sum_{i=1}^{40}c_{2i+1}}{41!}$ a_i, b_i, c_i – элементы массивов. Для вычисления сумм и факториала использовать функцию.

8. Составить программу для вычисления математического ожидания m_x и дисперсии D_x для моментов появления случайных событий t_1, t_2, …, t_{200} и интервалов между ними. Для вычисления m_x и D_x, использовать функцию без параметров.

9. Даны действительные числа *a, b, c*. Получить
$$\frac{\max(a,a+b)+\max(a,b+c)}{1+\max(a+bc,1.15)}.$$

10. Даны три строки. В каждой строке выполнить замену символов: все единицы на нули, все нули на единицы. Замену выполнять, начиная с заданной позиции строки.

11. Даны натуральные числа n, m, целые числа $a_1,\dots,a_n$, $b_1,\dots,b_m$,$c_1,\dots,c_{30}$.
Получить $l=\begin{cases}\min(b_1,\dots,b_m)+\min(c_1,\dots,c_{30}), & \text{если } |\min(a_1,\dots,a_n)|>10,\\ 1+(\min(c_1,\dots,c_{30}))^2, & \text{иначе.}\end{cases}$

12. Даны символьные массивы C1(n) и C2(m). Определить общее число строчных гласных латинских букв (a,e,i,o,u) в этих массивах.

13. По заданным 20-элементным целым массивам x и y вычислить $u=\begin{cases}\sum_{i=1}^{20}x_i^2 & \text{при } \sum_{i=1}^{15}x_iy_i>0\\ \sum_{i=10}^{20}y_i^2 & \text{иначе}\end{cases}$.

14. Даны действительные числа *a, b*. Получить
$$u=\min(a,b),\quad v=\min(ab,a+b),\quad \min(u+v^2,\,3.14).$$

15. Даны три строки. Используя функцию, определить позицию самого первого вхождения заданного символа в каждую из строк. Если строка не содержит символ, результатом работы функции должна быть –1.

16. По заданным 10-элементным вещественным массивам a, b и c вычислить $t=\begin{cases}\frac{\min(b_i)}{\max(a_i)}+\frac{\max(c_i)}{\min(b_i)} & \text{при}\\ \max(b_i)+\min(c_i) & \text{иначе}\end{cases}$

$\min(a_i)<\max(b_i)$

17. Даны целые массивы A(7), B(6), C(8). Определить, в каком массиве число элементов, значения которых не превышают значение их последнего элемента, является наибольшим.

18. Даны два вещественных 20-элементных массива. Переписать в новые массивы элементы исходных, целая часть которых кратна 3.

19. Даны натуральные числа *n, m*, вещественные матрицы *a(n,m)* и *b(n,m)*. Поменять местами максимальные элементы матриц *a* и *b*. (Считать, что в каждой матрице только один максимальный элемент).
20. Даны две матрицы C(5,4) и D(6,5). Для каждой матрицы сформировать массивы из сумм положительных элементов каждой строки; если их в строке нет, результат должен быть равен 0.
21. Даны две матрицы A(6,4), B(3,5). Напечатать ту матрицу, у которой произведение максимального и минимального элементов наибольшее.
22. Даны три целые матрицы размером 9x4. Напечатать ту из них, где больше ненулевых элементов (если таких матриц несколько, напечатать их все).
23. Даны три вещественные матрицы A(5,6), B(7,4), C(4,5). Вычислить произведение минимальных элементов матриц.
24. Даны две матрицы C(5,7) и D(6,8). Для каждой матрицы сформировать массивы из произведений отрицательных элементов каждого столбца; если их в столбце нет, результат должен быть равен 0.
25. Даны две матрицы C(8,6) и D(7,5). Для каждой матрицы сформировать массивы из средних арифметических значений элементов нечетных строк.

Контрольные вопросы

1. Целесообразность использования функций.
2. Из чего состоят и где расположены описания функций?
3. Что такое локальные и глобальные переменные?
4. Какие способы передачи параметров реализованы в C++?
5. Какие параметры называются формальными, а какие фактическими? Как они согласуются?
6. Правила вызова функций.