## Тема 3. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЦИКЛИЧЕСКОЙ СТРУКТУРЫ

**Цель занятия**: освоить разработку и программирование алгоритмов циклической структуры; получить практические навыки в выборе и использовании операторов цикла.

### Теоретические сведения

**Цикл –** это выполнение определенного набора операторов (*тела цикла*) некоторое количество раз. Использование циклов позволяет существенно сократить объем схемы алгоритма и длину соответствующей ей программы. Различают *циклы с известным (заданным) и неизвестным числом повторений*. К последним относятся *итерационные циклы*.

Для *организации цикла* необходимо выполнить следующие действия: 1) выполнить *подготовку цикла*, т.е. перед циклом задать *начальное значение* переменной, изменяющейся в цикле. Такая переменная называется *параметром цикла*; 2) *изменять параметр* перед каждым новым повторением цикла; 3) *проверять условие* окончания или повторения цикла; 4) *управлять циклом*, т.е. переходить к его началу, если он не закончен, или выходить из него по окончании. Последние три функции выполняются многократно.

Следует иметь в виду, что параметром цикла является при использовании простой переменной сама переменная, а при использовании переменной с индексом – её индекс [2].

Цикл реализуют при помощи операторов цикла. В языке С++ три вида операторов цикла: *с параметром, с предусловием и постусловием.*

**Оператор цикла с параметром (for).** Инструкция **for** - это наиболее часто используемое средство организации цикла. Инструкция цикла for предназначена для многократного (циклического) выполнения одной инструкции или блока инструкций в зависимости от того, истинно заданное условие или нет. Цикл for имеет следующий формат:

**for (инициализирующее выражение; условие; модифицирующее выражение)**

**инструкция**

При получении управления инструкцией ***for*** схема работы цикла следующая:

- вычисляется инициализирующее выражение (ИВ);
- вычисляется условие;
- если условие истинно (равно true), то выполняется простая инструкция или блок, составляющие тело цикла;
- вычисляется модифицирующее выражение (МВ) и всё повторяется, начиная со второго пункта (вычисления условия);
- если условие ложно, то выполнение цикла for заканчивается и начинает выполняться следующая за циклом инструкция.

***Пример.*** Вычислить квадраты чисел от 1 до 9.

```
int main()
        {
          int i, b;
          for (i =1; i<10; i++)
             b = i * i;
          return 0;
        }
```

Алгоритмическая структура, соответствующая этому оператору, показана на рис. 3.1.

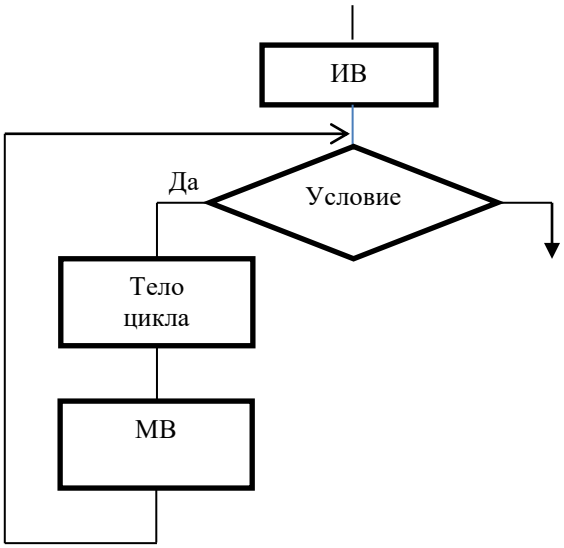


Рис. 3.1. Алгоритмическая структура выполнения оператора **for**

***Пример.*** Пусть s = 0. После выполнения оператора

```
for (int i = 4; i <=6; ++i)
  {
          int s = s + i;
          int z = 2 * i;
  }
```

переменная s получит значение 0+4+5+6=15, а переменная z = 12.

***Пример***. Заглавные латинские буквы в алфавитном порядке выведет цикл

```
for (char i = 'A'; i <='Z'; ++i)cout<<i;
```

а вложенные циклы

```
for (char i = 'A'; i <='Z'; ++i)
{
  for (char j = 'A'; j <=i; ++j)
  cout<<j;
  cout<<'\n';
}
```

Выведут на дисплей столбец последовательностей символов различной длины

A
AB
ABC
…
ABCD…WXYZ.

***Пример.*** Заглавные латинские буквы в обратном порядке выведет цикл:

```
for (char i = 'Z'; i >='A'; --i)     cout<<i;
```

***Задача 3.1.*** Вычислить значение выражения, состоящее из суммы и произведения $y = \sum_{i=1}^{25} \frac{5i^2}{i!} + \prod_{i=1}^{25} i^2$ [3].

1!=1
2!=1*2
3!=1*2*3
Схема алгоритма решения этой задачи приведена на рис. 3.2.

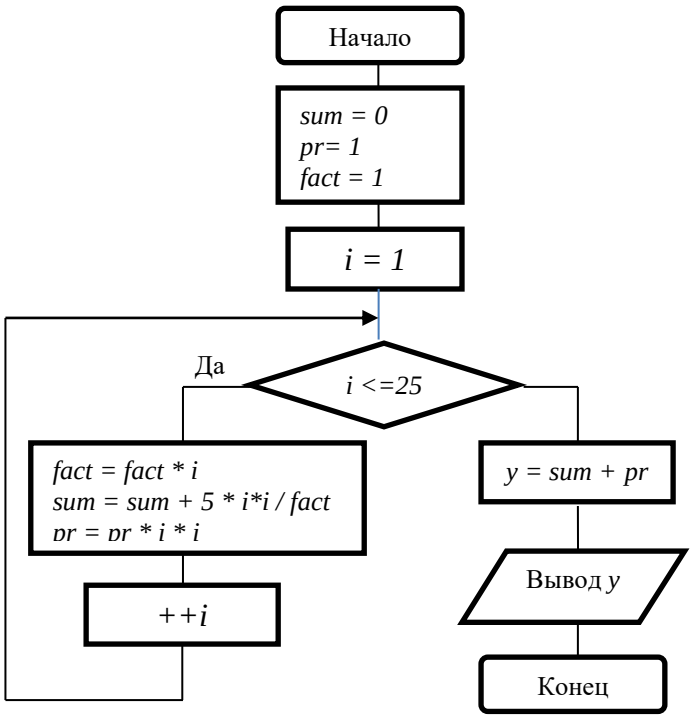


Рис. 3.2. Схема алгоритма решения задачи 3.1

Программа имеет вид:

```
#include <iostream>
#include <windows.h>
#include <math.h>
using namespace std;
int main()
{SetConsoleOutputCP(1251);
//Сначала сумма (sum) равна нулю, а произведение (pr) единице
double sum = 0, pr= 1;
long int  fact = 1;
for (int i = 1; i <=25; ++i)
{
  fact = fact * i;                  // Вычисляем i! = 1 * 2 * 3 * ... * i
  sum = sum + 5 * i*i / fact;     //Вычисляем сумму
  pr = pr * i * i;                  // Вычисляем произведение
}
double y = sum + pr;
cout<<"y="<<y;
}
```

***Задача 3.2***. Последовательность $x_1$, $x_2$, $x_3$… образована по закону: $x_1=1$; $x_2=0.3$; $x_i=(i+1)x_{i-2}$, i=3, 4, 5… . Получить первые 20 членов последовательности.

```
int main()
{
        SetConsoleOutputCP(1251);
        double x1=1.,x2=0.3,x3;
        cout<<"  x1="<<x1<<"  x2="<<x2<<'\n';
```

```
        for (int i = 3; i <=20; ++i)
        {
                x3=(i+1)*x1;
                if (i%2!=0)   cout<<"  x"<<i<<'='<<x3;    // вывод по два значения
                 else cout<<"  x"<<i<<'='<<x3<<'\n';      // в строке
                x1=x2;
                x2=x3;
        }
}
```

**Оператор цикла с предусловием (*while*).** Используется в тех случаях, когда число повторений цикла заранее не известно. Этот цикл имеет следующий формат:

**while (условие)**
**инструкция**

Схема выполнения цикла ***while*** следующая:

- вычисляется условие;
- если условие истинно (равно true), то выполняется одна инструкция или блок инструкций, составляющие тело цикла;
- предыдущие два пункта повторяются до тех пор, пока условие не станет ложным;
- если условие ложно, то выполнение цикла ***while*** заканчивается и начинает выполняться следующая за циклом инструкция.

Чтобы прервать выполнение цикла до того, как условие станет ложным, в теле цикла можно использовать инструкцию ***break***.

***Пример.*** Цикл с предусловием. Вычисление суммы чисел пока не будет введено нулевое значение

```
while (a!=0)
{
cin>>a;
s+=a;
}
```

Алгоритмическая структура, соответствующая оператору **while**, показана на рис. 3.3.

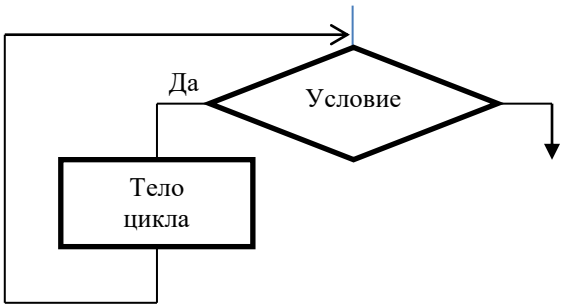


Рис. 3.3. Алгоритмическая структура выполнения оператора **while**

***Пример.*** Пусть переменные x, s равны x = 4, s = 0. После выполнения оператора

```
while (x <= 8)
{ s = s + x; x = x +1; }
```

они получат значения $s = 4 + 5 + 6 + 7 + 8 = 30$, $x = 9$.

***Пример.*** Вывести на экран таблицу чисел от 20 до 30, их квадраты и кубы, используя команду **while**, можно так:

```
int i = 20;
while (i <= 30)
{
  cout<< i<< "   "<<i * i<<"   "<< i * i * i<<'\n';
  i++;
}
```

*Замечание.* В теле цикла необходимо изменять параметр, иначе произойдет так называемое зацикливание (бесконечное повторение тела цикла).

***Задача 3.3.*** Составить программу табулирования функции

$$y=\begin{cases} a\lg x+\sqrt{|x|}, & \text{если } x>1; \\ 2a\cos x+3x^2, & \text{если } x\le 1, \end{cases}$$

для $a$=0,9 при изменении аргумента $x$ в диапазоне $x\in[0{,}8;2]$ с шагом 0,1. Вывод значений $x$ и $y$ выполнить в виде таблицы.

```
#include <iostream>
#include <windows.h>
#include <math.h>
using namespace std;
int main()
{
  SetConsoleOutputCP(1251);
  double a, x, y,x0, xk, step;
  cout<<"Введите a, x0, xk, step\n";            //Ввести с клавиатуры
  cin>>a>> x0>>xk>> step;                       // a=0.9, x0=0.8, xk=2 и step=0.1
  cout<<"\n Таблица функции Y(X)\n";            //Заголовок таблицы
  cout<<"\n X       Y\n";
  x=x0;
  while (x<=xk + step / 2)
//Слагаемое step / 2 обеспечивает включение последней точки xk   в таблицу
  {
        if (x > 1)   y = a * log10(x) + sqrt(abs(x));
        else   y = 2 * a * cos(x) + 3 * x * x;
        cout<< x<<"   "<<y<<'\n';
        x += step;
  }
}
```

***Задача 3.4.*** Составить таблицу значений аргумента $x$ и функции $y = \sin(x)$ на отрезке [0; 3,1] с шагом $h$ = 0,1. Вычислить среднее арифметическое (s1) значений функции больших, чем 0,1, и меньших, чем 0,6.

```
int main()
{
  SetConsoleOutputCP(1251);
  double x = 0, xk = 3.1, h = 0.1, s = 0, n = 0;
  while (x <= xk + h / 2.)
  {
        double y=sin(x);
        cout<<"x = "<< x<<"     y = "<< y<<'\n';
```

```
            if (y>0.1 && y<0.6)
              {
                  s+= y;
                  n++;
              }
            x+= h;
    }
     if (n > 0)
    {
          double s1 = s / n;
          cout<<"Среднее ="<< s1<<'\n';
    }
     else cout<<"Таких значений нет n=0";
}
```

**Оператор цикла с постусловием (*do / while* ).**

Используется в тех случаях, когда необходимо выполнить тело цикла хотя бы один раз. Этот цикл имеет следующий формат:

**do**
**инструкция**
**while (условие)**

Схема выполнения цикла ***do / while*** следующая:

- выполняется тело цикла (которое может быть одиночной инструкцией или блоком);
- вычисляется условие;
- если условие истинно, предыдущие два пункта повторяется до тех пор, пока условие не станет ложным;
- если условие ложно, то выполнение цикла ***do / while*** заканчивается и начинает выполняться следующая за циклом инструкция.

Чтобы прервать выполнение цикла до того, как условие станет ложным, можно использовать инструкцию ***break***.

Алгоритмическая структура, соответствующая оператору **do / while**, показана на рис. 3.4.

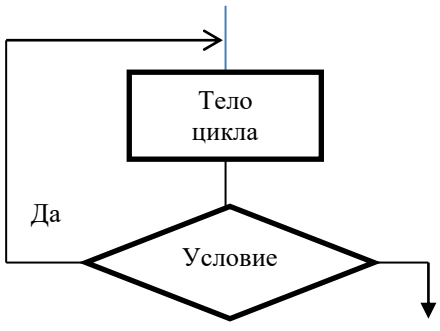


Рис. 3.4. Алгоритмическая структура выполнения оператора **do / while**

***Пример***. Цикл с постусловием. Вычисление суммы чисел пока не будет введено нулевое значение

```
do
{
cin>>a;
```

```
s+=a;
}
while(a!=0);
```

Циклы *while* и *do / while* могут быть вложенными:

***Пример***. Использование вложенных циклов

```
int i, j, k;
    i = 0;  j = 0;  k = 0;
    do
    {
       i++;
       j--;
       while (a[k] < i)  k++;
    }
    while (i < 30 && j > -30);
```

***Задача 3.5***. Протабулировать функцию $y=\sin(x)$ на отрезке $[-\pi,\pi]$ с шагом $h=\pi/5$. Результаты вычислений вывести на экран в виде таблицы. Найти максимальное (max) и минимальное (min) значения функции на этом отрезке.

Максимальное или минимальное значения функции или значение, удовлетворяющее некоторому условию поиска, определяют методом сканирования (перебора, просмотра) всех подряд значений функции и их сравнения с некоторым эталоном.

```
#include <iostream>
#include <windows.h>
#include <math.h>
using namespace std;
int main()
{
  SetConsoleOutputCP(1251);
  double  h, x, y, max, min,pi=3.14;
  h=pi/5;
  x=-pi;
  max=sin(x);                    //Предполагаем, что max и min
  min= sin(x);             // достигаются в первой точке
  cout<<"  x        y "<< '\n';
  do
     {
          y = sin(x);
          cout<<x<<"   "<< y<< '\n';
          if  (y > max)  max=y;          //Определяем максимум функции
          if  (y<min)  min=y;            //Определяем минимум функции
          x += h;                                //Увеличиваем x на h
     }
  while (x < pi + h / 2);
  cout<<"max="<<max<<"   min="<<min;
}
```

*Замечание.* Чтобы кроме этого определить значение аргумента (xmax), при котором функция достигает, например, максимального значения, условный оператор следует использовать так:

```
xmax =-pi;
```

```
…
 do
    {
          …
         if (y>max) {    max=y; xmax=x; }
          …
    }
while (x < pi + h / 2);
```

**Методические указания**

При подготовке к занятию необходимо изучить: возможности языка С++ для построения циклов с известным числом повторений; организацию циклов; приемы программирования – накопления суммы, произведения; табулирования функции.

**Аудиторные и домашние задания**

1. Составить таблицу значений аргумента $x$ и функции $y$ = tg$x$ на отрезке [2; 6,1] с шагом h = 0,2. Вычислить среднее арифметическое значений функции больших, чем 0,2, и меньших, чем 0,8.
2. Вывести таблицу значений функции $y(x)=\begin{cases}\dfrac{\ln(x^2+0{,}75)}{4x^2+x}, & \text{если } |x|<5\\ x^4+2x^2\sin x, & \text{если } |x|>5\end{cases}$, где $x\in[-10;10]$, $h_x=2$.
3. Последовательность чисел Фибоначчи $u_0, u_1, \ldots$ образуется по закону $u_0=0$, $u_1=1$, $u_i=u_{i-1}+u_{i-2}$ $(i=2,3,..)$. Дано натуральное число $n>1$. Получить $u_n$.
4. Дано целое число $n$, вещественное число $b$. Вычислить $b(b+n)(b+2n)\ldots(b+n^2)$.
5. Даны вещественное число $a$, целое число $n$. Вычислить $\frac{1}{a}+\frac{1}{a(a+1)}+\ldots+\frac{1}{a(a+1)\ldots(a+n)}$.
6. Дано натуральное число $n$. Вычислить произведение первых n сомножителей : $\frac{1}{2}\cdot\frac{3}{4}\cdot\frac{5}{6}\ldots$
7. Дано действительное $x$ , найти : $y=\dfrac{(x-2)(x-4)(x-8)\ldots(x-64)}{(x-1)(x-3)(x-7)\ldots(x-63)}$.
8. Дано действительное $x$ , найти : $y=-\dfrac{x^3}{3!}+\dfrac{x^5}{5!}-\dfrac{x^7}{7!}+\dfrac{x^9}{9!}-\dfrac{x^{11}}{11!}+\dfrac{x^{13}}{13!}$.
9. Пусть $x_1=x_3=x_2=1$; $x_i=x_{i-1}+x_{i-3}$ ; $i=4,5,6\ldots$ Найти $s=\sum_{i=1}^{100}\frac{x_i}{2^i}$.
10. Дано целое число $n$ ($n>1$). Вычислить $y=1!+2!+3!+\ldots+n!$.
11. Даны целое число $n$, вещественное $x$. Вычислить $\sum_{i=1}^{n}\dfrac{x+\cos(ix)}{2^i}$.
12. Пусть $x_1=y_1=1$, $x_i=0.3x_{i-1}$, $y_i=x_{i-1}+y_{i-1}$, $i=2, 3, \ldots$ . Дано целое число $n$. Найти $\sum_{i=1}^{n}\frac{x_i}{1+|y_i|}$.
13. Вычислить $\prod_{i=1}^{52}\dfrac{i^2}{i^2+2i+3}$.
14. Дано натуральное число $n$, вещественное число $x$. Вычислить $\prod_{k=1}^{n}\left(1+\dfrac{\sin(kx)}{k!}\right)$.

15. Дано целое число *n*. Вычислить значения функции $y = \frac{x^2 - 3x + 2}{\sqrt{2x^3 - 1}}$ для *x=1, 1.1, 1.2, ..., 1+0.1n.*
16. Вычислить последовательности значений функций $p_1(x) = x$, $p_2 = \frac{3x^2 - 1}{2}$, $p_3(x) = \frac{5x^2 - 3x}{2}$ для значений аргумента *x=0, 0.05, 0.1, ..., 20.*
17. Подсчитать количество цифр в десятичной записи целого неотрицательного числа *n*.
18. Даны натуральное число *n* и вещественные числа *t*, $a_0$ , $a_1$ , … , $a_n$ . Вычислить значение многочлена $a_0x^n + a_1x^{n-1} + ... + a_{n-1}x + a_n$ .
19. Вычислить: $Y = \sqrt{3 + \sqrt{6 + ... + \sqrt{96 + \sqrt{99.}}}}$
20. Дано 100 вещественных чисел. Определить, образуют ли они возрастающую последовательность.
21. Даны натуральное *n* и вещественные числа $x_1$, $y_1$, $x_2$, $y_2$,…,$x_n$, $y_n$. Рассматривая пары $x_i$, $y_i$ как координаты точек на плоскости, определить радиус наименьшего круга (с центром в начале координат), внутрь которого попадают все эти точки.

**Контрольные вопросы**

1. Назначение и правила организации цикла.
2. Опишите схему работы цикла **for.**
3. Опишите схему работы цикла **while.**
4. Опишите схему работы цикла **do / while**.
5. В чем заключается различие между операторами **do / while** и **while**?
6. Какие операторы составляют тело цикла?
7. Какую инструкцию можно использовать в теле цикла, чтобы прервать выполнение цикла до того, как условие станет ложным?