## 19-20 ОАП Пр4 Составление программ циклической структуры

## Теоретические сведения

В С++ существуют три типа операторов цикла: цикл с предусловием, цикл с постусловием и цикл с параметром.

**Цикл с предусловием.** Формат оператора цикла с предусловием:

**while** (выражение) оператор;

Цикл повторяет свое выполнение, пока значение выражения истинно (отлично от нуля).

В качестве примера использования оператора цикла рассмотрим программу вычисления факториала целого положительного числа N!.

**Пример 1.**

*// Программа вычисления факториала*

#include <iostream>   
**using namespace** std; **int** main(**int** argc, **char**\*\* argv)   
{   
 **long int** F;   
 **int** i,N;  
 cout << ”N=";  
 cin >>N;  
 **while**(i<=N) F=F\*i++;   
 cout <<*"\n"*<< N << *"!="* << F;

}

Обратите внимание на операторы в теле цикла. Конечно, и в С++ программе можно было написать два оператора присваивания, объединив их фигурными скобками. Однако использованный способ записи более лаконичен и более характерен для С++. Этот же самый оператор можно было записать еще короче: F\*=i++

При практическом использовании этой программы не следует забывать, что факториал — очень быстро растущая функция, и поэтому при определенных значениях N выйдет из диапазона, соответствующего типу long int. Задав для переменной F тип unsigned long, можно сдвинуть эту границу, но этого может оказаться недостаточно.

Интересно свойство следующего оператора:

**while**(**true**);

Это бесконечный пустой цикл. Использование в качестве выражения константы **true** приводит к тому, что условие повторения цикла все время остается истинным и работа цикла никогда не заканчивается. Тело в этом цикле представляет собой пустой оператор. При исполнении такого оператора программа будет «топтаться на месте».

Рассмотрим еще один пример использования оператора цикла **while**. Рассмотрим задачу итерационного вычисления суммы:  с заданной точностью .

*// Сумма*#include <iostream>   
#include <climits>   
**using namespace** std;

**int** main (**int** argc, **char**\*\* argv)  
{  
 **int** n=l;  
 **double** S=0, eps;  
 cout << *"Точность"*;  
 cin >> eps;  
 **while**(1.0/n>eps && n<INT\_MAX)  
 S+=l./n++;   
 cout<<*"\nСумма=”* << S << endl;  
}

Файл climits, подключаемый препроцессором, содержит определения предельных констант для целых типов данных. В частности, константа с именем INT\_MAX равна максимальному значению типа int в данной реализации компилятора. Если для типа int используется двухбайтовое представление, то INT\_MAX=327 67. В этом же заголовочном файле определены и другие константы: INT\_MIN=-327 68; LONG\_MAX=2147483647 И Т.Д.

**Цикл с постусловием.** Формат оператора цикла с постусловием:

**do** оператор **while** (выражение);

Цикл выполняется до тех пор, пока выражение истинно (отлично от нуля). Выход из Цикла происходит после того, как значение выражения станет ложным. Таким образом, в отличие, например, от оператора repeat... until, используемого в Паскале, где в конце пишется условие выхода из цикла, в операторе **do** ... **while** в С++ в конце пишется условие повторения цикла. В качестве примера рассмотрим программу вычисления N!, в которой используется цикл с постусловием.

*// Программа вычисления факториала*#include <iostream>

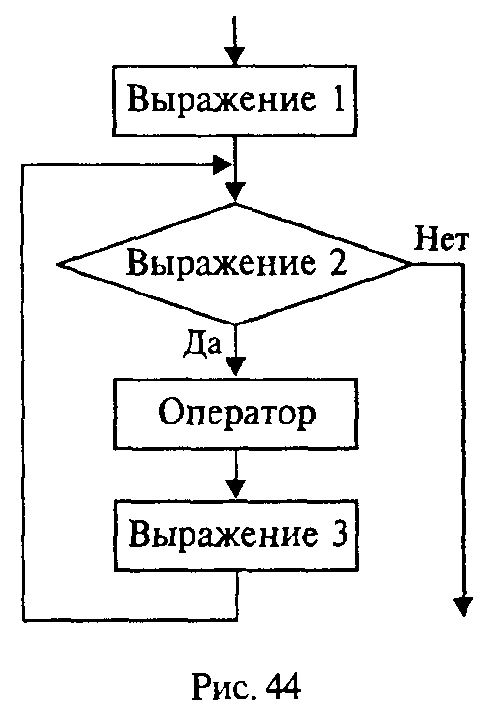
**using namespace** std;  
**int** main(**int** argc, **char**\*\* argv)   
{   
 **long int** F;   
 **int** i,N;  
 cout << *"N="*;   
 cin >> N;  
 F=i=l;   
 **do** F\*=i++;   
 **while**(i<=N);  
 cout <<*”\n”*<<N<<*"!="*<<F;  
}

**Цикл с параметром.** Формат оператора цикла с параметром:

**for**(выражение1; выражение2; выражение3) оператор;

***Выражение1*** выполняется только один раз в начале цикла. Обычно оно определяет начальное значение параметра цикла (инициализирует параметр цикла). ***Выражение2*** — это условие выполнения цикла. ***Выражение3*** обычно определяет изменение параметра цикла, ***оператор*** — тело цикла, которое может быть простым или составным. В последнем случае используются фигурные скобки.

Алгоритм выполнения цикла **for** представлен на блок-схеме на рисунке ниже.



Обратите внимание на то, что после вычисления ***выражения3*** происходит возврат к вычислению ***выражения 2*** — проверке условия повторения цикла.

С помощью цикла **for** нахождение N! можно организовать следующим образом:

**long** F = 1;  
**for**(**int** i=l; i<=N; ++i) F\*=i;

Используя операцию «запятая», можно в выражение 1 внести инициализацию значений сразу нескольких переменных:

**for**(**long** F=l,i=l; i<=N; ++i) F\*=i;

Некоторых элементов в оператора **for** может не быть, однако разделяющие их точки с запятой обязательно должны присутствовать. В следующем примере инициализирующая часть вынесена из оператора for:

**long** F=l; i=l;

**for**(;i<=N;++i) F=F\*i;

Ниже показан еще один вариант вычисления N!. В нем на месте тела цикла находится пустой оператор, а вычислительная часть внесена в выражение 3.

**for**(F=l,i=l;i<=N;F=F\*i,++i);

Этот же оператор можно записать в следующей форме:

**for**(F=l,i=l;i<=N;F\*=i++);

В языке С++ оператор **for** является достаточно универсальным средством для организации циклов.

Следующий фрагмент программы на С++ содержит два вложенных цикла **for**. В нем запрограммировано получение на экране таблицы умножения.

**for**(**short** х=2;х<=9;++х) **for**(**short** у=2;у<=9;++у)  
cout <<*”\n"*<< x<<*”\*"*<<y<<*” = "*<<x\*y<<endl;

На экране будет получен следующий результат:

2\*2=4

2\*3=6

…

9\*8=72

9\*9=81

**Оператор continue.** Если выполнение очередного шага цикла требуется завершить до того, как будет достигнут конец тела цикла, используется оператор **continue**. Следующий фрагмент программы обеспечивает вывод на экран всех четных чисел в диапазоне от 1 до 100.

**for**(**short** i=l;i<=100;++i)  
{  
 **if** (i%2) **continue**;  
 cout << *"\t"* << i;   
}

Для нечетных значений переменной i остаток от деления на 2 будет равен единице, этот результат воспринимается как значение «истина» в условии ветвления, и выполняется оператор **continue**. Он завершит очередной шаг цикла, выполнение цикла перейдет к следующему шагу.

## Контрольные вопросы

1. Как функционирует оператор цикла *for* и какая типовая схема алгоритма ему соответствует?
2. Как функционирует оператор цикла *while* и какая типовая схема алгоритма ему соответствует?
3. Как функционирует оператор цикла *do…wile* и какая типовая схема алгоритма ему соответствует?
4. Определить значение переменной S после выполнения операторов:

s=0; i=1;  
**while** (i>5) { ++i; s+=1/i;}

1. Определить значение переменной S после выполнения операторов:

s=0; i=1;  
**do** {s+=s+1/i; ++i;} **wile** (i<1)

1. Определить значение переменной S после выполнения операторов:

s=0; i=1;  
**while** (i>1){s+=1/i; --i;}

1. Определить значение переменной S после выполнения операторов:

s=1; n=1;  
**for** (**int** i=2; i<= n; ++i) s += 1/i;

1. Сколько раз будет выполнятся тело цикла?

k:=0;  
**for** (**int** i=1;i <= k+1 ;++k)++k;

## Задания к практической работе № 4

1. Вычисление f=10! описать каждым из трех вариантов оператора цикла.
2. Подсчитать k – количество цифр в десятичной записи целого неотрицательного числа n.
3. Задача Л. Эйлера. Некий чиновник купил лошадей и быков на 1770 талеров. За каждую лошадь он уплатил по 31 талеру, а за каждого быка – по 21 талеру. Сколько лошадей и быков купил чиновник? Выяснить, если решения в целых числах имеются, то сколько их – одно или несколько?
4. Вычислить y = cos x+cos2x + cos3 x +...+cos30x.
5. Вычислить y = sin 1+ sin 1.1 + ... +sin 2.
6. Логической переменной p присвоить значение *true*, если целое n (n>1) – простое число, и значение *false* иначе.
7. Вычислить:

20

p=П xi2, где xi=xi-1+dx; dx=0.5.

i=1

1. Определить k – количество трехзначных натуральных чисел, сумма цифр которых равна n (1<n<27).
2. Напечатать в возрастающем порядке все трехзначные числа, в десятичной записи которых нет одинаковых цифр.
3. Определить количество натуральных чисел, не превышающих n , которые не делятся на 11.
4. Дано натуральное число n.

а) Выяснить, входит ли цифра 3 в запись числа n.

б) Поменять порядок цифр числа n на обратный.

1. Дано натуральное число n.

а) Переставить первую и последнюю цифры числа n.

б) Приписать по единице в начало и в конец записи числа n.

1. Вычислить:

10

S=∑(xi3+8)0.5 , где xi=xi-1+dx2; x1=2; dx=0.2.

i=1

1. Вычислить:

8

D= ∑ (xj2+1.5)/(|xj|+10) , где xj=xj-1+dx; x1=2; dx=1.

j=1

1. Вычислить:

15

R= Π sin xi2, где xi=xi-1+dx; dx=0.1; x1=1.

i=1

1. Математик Христиан Гольдбах (1690 –1764 гг.) выдвинул гипотезу, что любое четное число, большее 2, представимо в виде суммы двух простых чисел. Проверьте эту гипотезу для всех четных чисел, не превышающих число 30.
2. Вычислить:

10

B= ∑(xi+yi)/(|xi|+|yi|), где xi=xi-1+dx; yi=yi-1+dy; dx=0.5;

i=1 dy=0.2; x1=1; y1=0.5.

1. Вычислить:

10

P= Π xi2/yi2+ |xi|, где xi=xi-1+dx; yi=yi-1+dy; dx=0.1;

i=1 dy=0.2; x1=3; y1=2.

1. Вычислить:

10

F= ∑ xi2 + xi +7, где xi=xi-1 + dx; x1=2; dx=0.1 .

i=1

1. Вычислить:

10 ¬

S= ∑√xi  + 5.3, где xi=xi-1+dx; x1=3; dx=0.2 .

i=1

1. Вычислить:

20

R= ∑ cos xi2, где xi=xi-1+dx; dx=0.3; x1=2.

i=1

1. Вычислить:

12

A= ∑ ctg xi3, где xi=xi-1+dx; dx=0.1; x1=1.

i=1