Лекции 4

ЦИКЛИЧЕСКИЕ ВЫЧИСЛИТЕЛЬНЫЕ ПРОЦЕССЫ В VISUAL C++ 2010

Цель работы: изучение циклических программ с использованием операторов цикла **while и do...while.**

Вопросы лекции:

1. Циклические вычислительные процессы
2. Оператор цикла **while** в среде Visual C++ 2010
3. Исследование оператора цикла **do...while** в среде Visual C++ 2010
4. Вложенные циклы в Visual C++ 2010

1. Циклические вычислительные процессы

Возможность повторно выполнять некоторые действия очень важна при разработке любых программ и программных приложений. Вычислительный процесс, содержащий многократные вычисления по одним и тем же математическим зависимостям, называется **циклическим.**

Цикл выполняет оператор или группу операторов до тех пор, пока истинно (или ложно) определенное условие относительно некоторой переменной, называемой **параметром цикла.**

Многократно повторяющиеся части такого процесса составляют **тело цикла.**

Алгоритм циклических структур должен содержать:

1. **Подготовку к циклу** – присваивание начального значения параметру цикла.
2. **Проверку условия** выполнения тела цикла.
3. **Тело цикла** – действия, которые выполняются в циклической программе для разных значений параметра цикла.
4. **Изменение (модификация)** значений параметра цикла.

На рис. 1 изображена блок-схема алгоритма циклического вычислительного процесса, где помимо характеристик операционных блоков в качестве примера приведены реальные операторы.

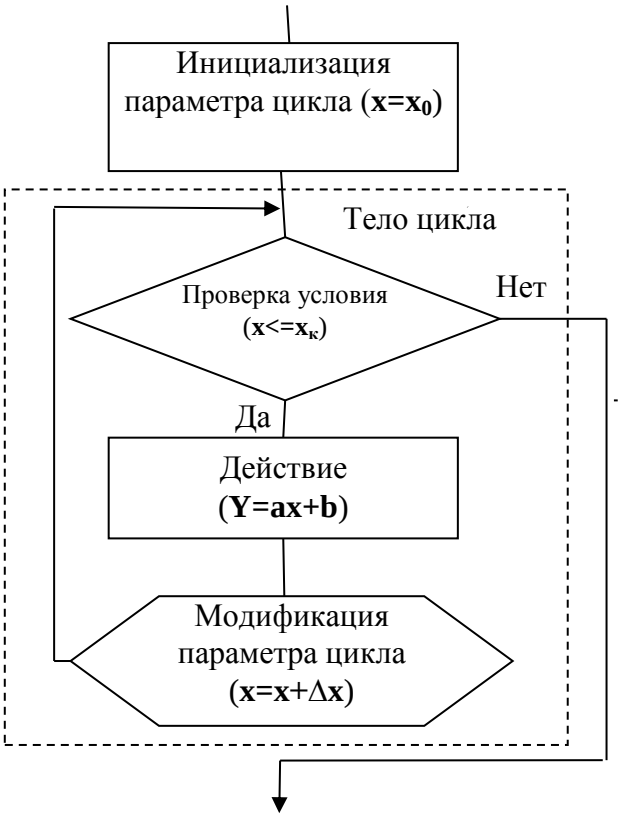

Рис. 1. Блок-схема алгоритма циклического вычислительного процесса

В среде Visual C++ 2010 циклические вычислительные процессы реализуются с помощью операторов **while**, **do...while** и **for**, анализ которых будет проведен ниже.

Для лучшего понимания действия операторов в настоящей работе следует знать перевод следующих английских слов:

do - делать, выполнять;
while - пока;
for - для.

2. Оператор цикла while в среде Visual C++ 2010

Оператор цикла **while** реализует вычислительную структуру **цикл с предусловием** и имеет следующий вид:

while (логическое выражение)
оператор;

В качестве **логического выражения** используется отношение или логическое выражение относительно параметра цикла в виде какого-то условия. Если оно истинно, то тело цикла (**оператор**) выполняется до тех пор, пока **выражение** не перестанет выполняться, т. е. не станет ложным. **Оператор** может быть простым или составным (из нескольких операторов, заключенных в фигурные скобки).

Блок-схема алгоритма циклического вычислительного процесса с предусловием, реализуемого оператором цикла **while**, приведена на рис. 1.

Исследуем простейшую программу с оператором цикла **while**, вычисляющую значения функции: $\mathbf{Y=ax + \sin(\pi x)}$, если значение x изменяется от $\mathbf{x_0}$ до $\mathbf{x_k}$ с шагом Δx.

Блок-схема алгоритма решения данного задания представлена на рис.6.2.

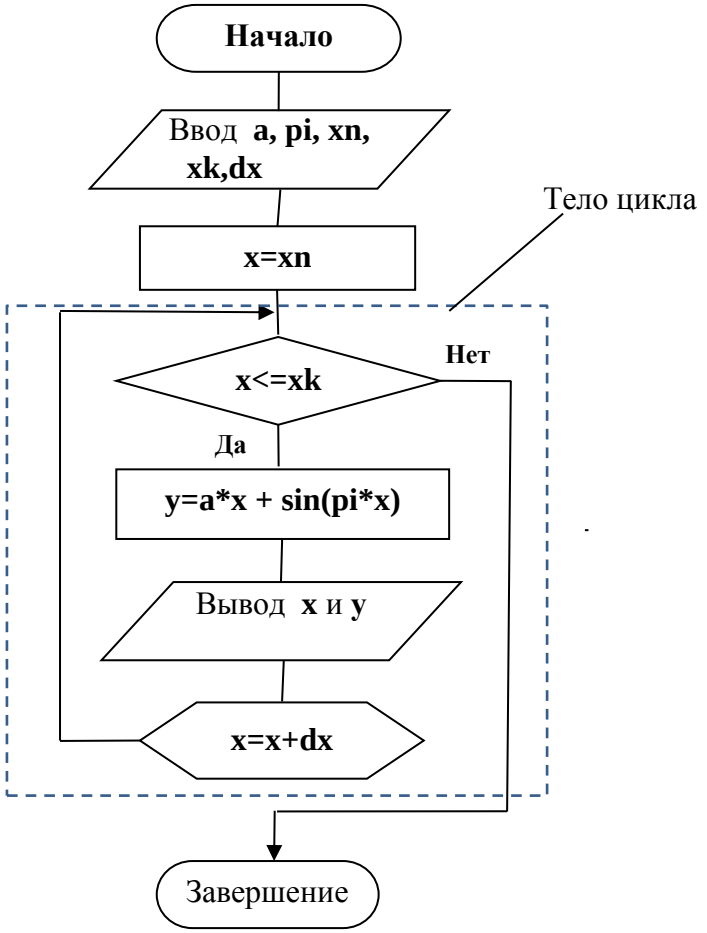

Рис. 2. Блок-схема алгоритма циклической программы с оператором **while**

Определим типы и значения исходных данных, которые понадобятся для решения задачи:

a – константа, инициализируем ее, как **a** = 2 и определим ее тип как **int** (целое);

π – константа, ее инициализируем как **pi** = 3.14, тип **double** (действительное двойной точности);
x_0 – переменная, обозначим ее как **xn**, тип **double**;
x_k – переменная, обозначим ее как **xk**, тип **double**;
Δx – переменная, обозначим ее как **dx**, тип **double**;
x – переменная, обозначим ее как **x**, тип **double**;
y – переменная, обозначим ее как **Y**, тип **double**.

Ниже приведен программный код, реализующий данный алгоритм. Обратите внимание на сдвиг фигурных скобок, ограничивающих сложные операторы, относительно скобок, охватывающих тело главной функции.

```
int _tmain(int argc, _TCHAR* argv[])    //Объявление главной функции _tmain
{
const int a=3;                          //Инициализация целой константы a
const  double  pi=3.14;                 //Инициализация действительной константы π
double x, y, xn, xk, dx;                //переменных и параметра цикла: Y, x, x0, xk и Δx
cout<<" Vvedite xn, xk, dx: "<<endl;    //Вопрос для ввода параметров цикла
cin>>xn>>xk>>dx;                        //Ввод исходных данных
cout<<"xn= "<<xn<<"  xk= "<<xk<<"  dx= "<<dx<<endl;//Вывод исходных данных
x=xn;                                   //Подготовка к циклу – предоставление начального
                                        //значения параметру цикла - переменной x
cout<<" X   "<<"   Y  "<<endl;          //Вывод на экран заголовков "X" и "Y"
      while (x<=xk)                     //Оператор while (условие цикла x ≤ xk)
      {                                 //Начало составного оператора
      y=a*x + sin(pi*x);                //Вычисление действ. переменной y на данном шаге
      cout<<x<<"   "<<y<<endl;          //Вывод на экран действительной переменной y
      x=x+dx;                           //Вычисление следующего значения параметра x
      }                                 //Конец составного оператора
getch();                                //Функция задержки окна DOS на экране
return 0;
}                                       //Конец главной функции
```

3. Оператор цикла do...while в среде Visual C++ 2010

Оператор цикла **do...while** реализует вычислительную структуру **цикл с послеусловием** и имеет следующий вид:

do
оператор;
while (логическое выражение);

В качестве **логического выражения** используется отношение или логическое выражение относительно параметра цикла в виде какого-то условия. Если оно истинно, то тело цикла (**оператор**) выполняется до тех пор, пока **выражение** не перестанет выполняться, т. е. не станет ложным. **Оператор** может быть простым или составным (из нескольких операторов, заключенных в фигурные скобки).

Основное отличие оператора цикла **do…while** от оператора **while** состоит в том, что здесь условие относительно параметра цикла проверяется после выполнения тела цикла. Поэтому оператор **do…while** выполняется как минимум один раз.

Блок-схему вычислительного процесса, реализующего оператор цикла **do…while**, проанализируем на примере решения программы для вычисления значения функции $Y = \sin^3(x-a) - \cos^2(x+a)^3$ для значений аргумента **x** от x_0=0 до x_k=1 с шагом **Δx**=0,2. Используем оператор **do…while**. Блок-схема алгоритма решения данной задачи приведена на рис. 3.

Определим типы и значения исходных данных, которые понадобятся для решения задачи:

a – константа, инициализируем ее, как **a** = 0,3 и определим ее тип как **double** (действительное двойной точности);
x_0 – переменная, обозначим ее как **xn**, тип **double**;
x_k – переменная, обозначим ее как **xk**, тип **double**;
Δx – переменная, обозначим ее как **dx**, тип **double**;

x – переменная, обозначим ее как **x**, тип **double**;
y – переменная, обозначим ее как **Y**, тип **double**.

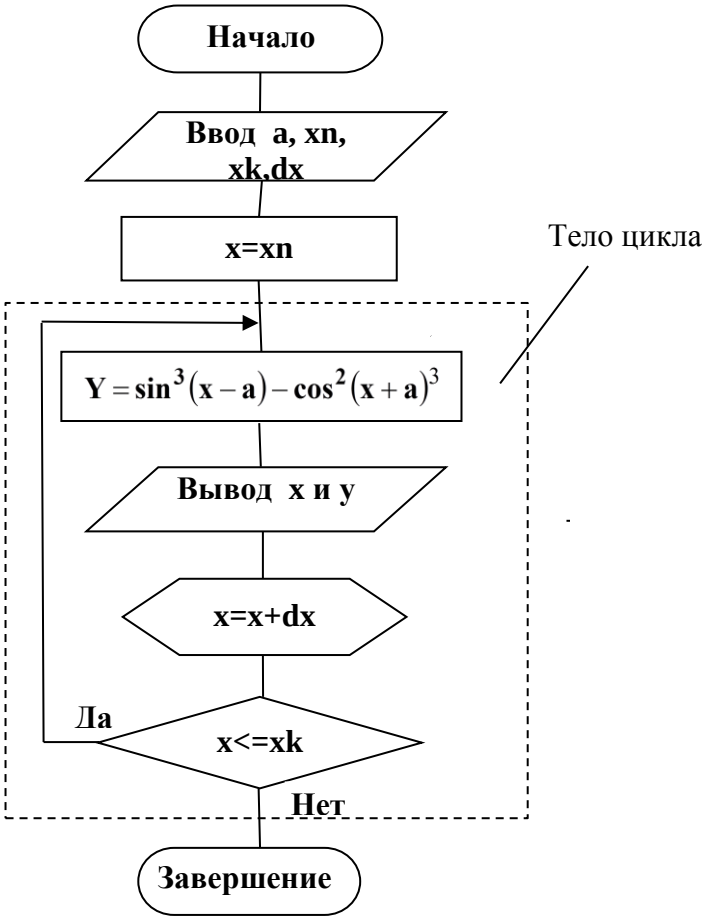

Рис. 3. Блок-схема алгоритма циклической программы с оператором **do...while**

Ниже приведен программный код, реализующий данный алгоритм. Обратите внимание на сдвиг фигурных скобок, ограничивающих сложные операторы, относительно скобок, охватывающих тело главной функции.

```
int _tmain(int argc, _TCHAR* argv[])    //Объявление главной функции _tmain
{
double a=0.3;                           //Инициализация константы a
double x, y, xn, xk, dx;                //переменные и параметр цикла: Y, x, x0, xk и Δx
cout<<" Vvedite xn, xk, dx: "<<endl;    //Вопрос для ввода параметров цикла
cin>>xn>>xk>>dx;                        //Ввод исходных данных
cout<<"xn= "<<xn<<"  xk= "<<xk<<"  dx= "<<dx<<endl;     //Вывод исходных данных
x=xn;                                   //Подготовка к циклу – предоставление начального
                                        //значения параметру цикла - переменной x
cout<<" X   "<<"   Y  "<<endl;          //Вывод на экран заголовков "X" и "Y"
      do                                //Начало оператора do…while
      {                                 //Начало составного оператора
      y=pow(sin(x-a),3)-pow(pow(cos(x+a),3),2);        //Вычисление y на данном шаге
      cout<<x<<"   "<<y<<endl;          //Вывод на экран действительной переменной y
      x=x+dx;                           //Вычисление следующего значения параметра x
      }                                 //Конец составного оператора
      while (x<=xk);                    //Оператор while (условие цикла x ≤ xk)
getch();                                //Функция задержки окна DOS на экране
return 0;
}                                       //Конец главной функции
```

4. Вложенные циклы в Visual C++ 2010

Циклы можно вкладывать друг в друга. Это следует из записи в общем виде, например, оператора цикла **while**:

while (логическое выражение)
оператор;

где **оператор** – любой оператор, в том числе и сложный, состоящий из нескольких операторов. Т. к. исключений из этого правила в языке C++ нет, то в качестве оператора может быть использован любой из операторов цикла.

Тогда вложенный цикл с оператором **while** будет иметь следующий вид:

```
while (логическое выражение)
{
        while (логическое выражение)
        оператор;
},
```

где **оператор** – любой оператор, в том числе и сложный.

Проанализируем работу программы в среде Visual C++ 2010, использующей такую вычислительную конструкцию, как вложенный цикл, или цикл в цикле на примере программы для вычисления значения функции $s=\sum_{n=1}^{4}\sum_{k=1}^{5}\sin^3(knx-a)$ для некоторой переменной **x**. Используем вложенный цикл с оператором **while**. Блок-схема алгоритма решения данной задачи приведена на рис. 4. Необходимо ввести код программы, построить решение и произвести вычисления.

Определим типы и значения исходных данных, которые понадобятся для решения задачи:

a – константа, инициализируем ее, как **a** = 1,4 и определим ее тип как **double** (действительное двойной точности);

x – переменная, обозначим ее как **x**, тип **double**;

y – переменная, обозначим ее как **Y**, тип **double**;

s – переменная, обозначим ее как **S**, тип **double**.

Ниже приведен программный код, реализующий данный алгоритм. Обратите внимание на сдвиг фигурных скобок, ограничивающих сложные операторы, относительно скобок, охватывающих тело главной функции.

```
int _tmain(int argc, _TCHAR* argv[])    //Объявление главной функции _tmain
{
const double a=1.4;                     //Инициализация константы a
double x, y, n, k, s;                   //Инициализация переменных и параметров цикла:
                                        //y,x, s, n  и k

cout<<" Vvedite x "<<endl;              //Вопрос для ввода  x
cin>>x;                                 //Ввод  переменной x
cout<<"x= "<<x<<"  a= "<<a<<endl;       //Вывод исходных данных
n=1;                                    //Подготовка к циклу – предоставление начального
                                        //значения параметру цикла - переменной n
s=0;                                    //Начальное значение переменной s
   while (n<=4)                         //Оператор while (условие цикла n ≤ 4)
   {                                    //Начало составного оператора – вложенного цикла
   y=0;                                 //Начальное значение переменной y
   k=1;                                 //Подготовка к циклу – предоставление начального
                                        //значения параметру цикла - переменной k
       while (k<=5)                     //Вложенный оператор while (условие цикла k ≤ 5)
       {
       y=y+ pow(sin(k*n*x-a),3);        //Суммирование y на k-ом шаге
       cout<<k<<"   y =  "<<y<<endl;    //Вывод на экран переменной y на k-ом шаге
       k=k+1;                           //Вычисление  следующего значения  параметра k
       }
```

```
    s=s+y;                          //Суммирование s на n-ом шаге
    cout<<n<<"   s =  "<<s<<endl;   //Вывод на экран переменной s на k-ом шаге
    n=n+1;                          //Вычисление  следующего значения  переменной n
    }                               //Конец составного оператора с вложенным циклом
getch();                            //Функция задержки окна DOS на экране
return 0;
}                                   //Конец главной функции
```

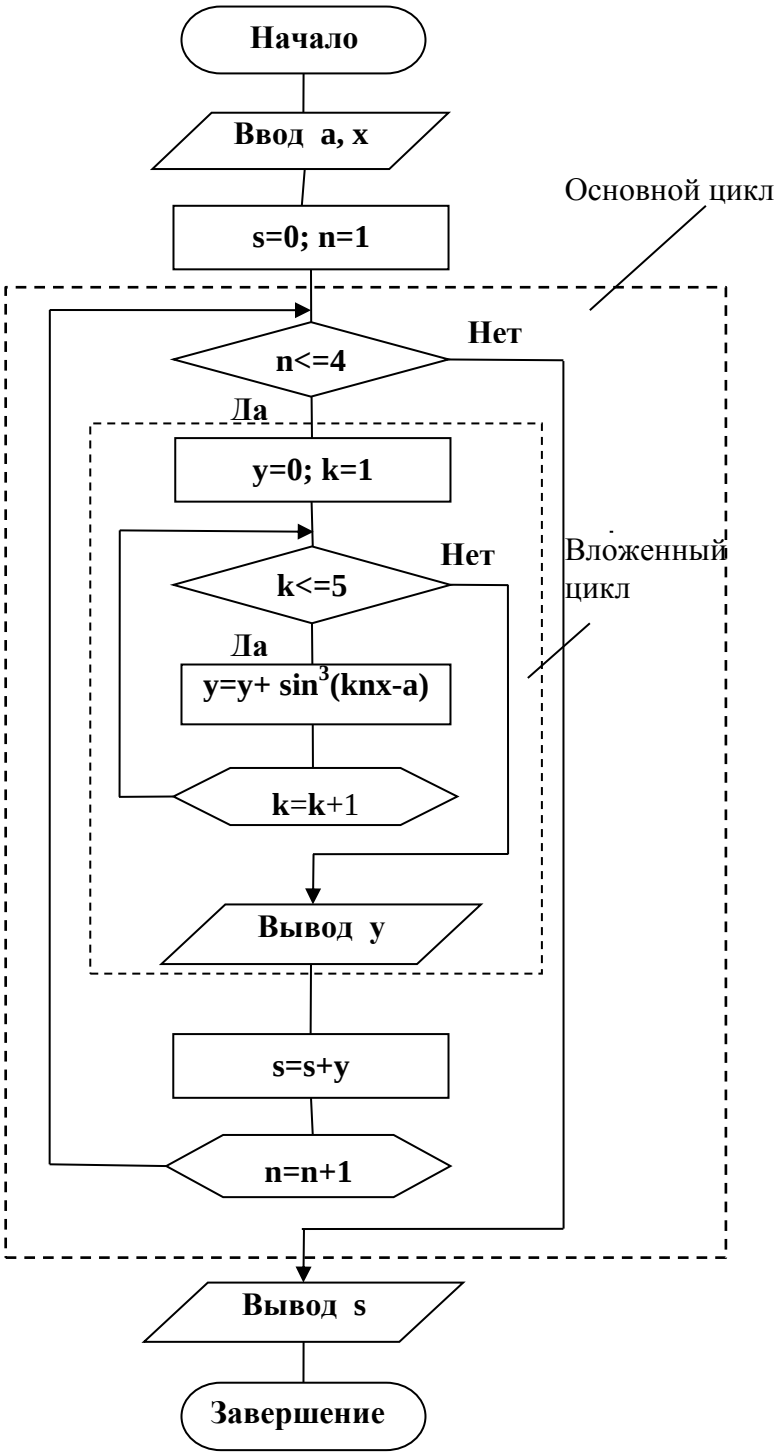

Рис. 4. Блок-схема алгоритма программы с использованием вложенного цикла

Вопросы для самоконтроля.

1. С помощью каких действий реализуются циклические процессы в языке С++?
2. Запишите в общем виде оператор цикла **while** и опишите, как он выполняется.
3. Запишите в общем виде оператор цикла **do... while** и опишите, как он выполняется.
4. Перечислите, что должно быть в конструкции цикла.
5. В языке С++ оператор **do… while** – это…
    1. Оператор цикла с предпосылкой
    2. Оператор цикла с послеусловием
    3. Оператор цикла с параметром
6. В языке С++ в операторе **while** последовательность операторов (блок)
    1. Нужно заключать в фигурные скобки
    2. Не нужно заключать в фигурные скобки
    3. Заключать в фигурные скобки на усмотрение автора программы
7. Сколько раз выполнится оператор цикла **int i=1; while(i>3) i=i+1;** ?
    1. Один раз
    2. Ни одного
    3. Некоторое число раз
8. Сколько раз выполнится оператор цикла **int x=1; do x=x+1; while(x>3);** ?
    1. Один раз
    2. Ни одного
    3. Некоторое число раз
9. Сколько раз выполнится оператор цикла **int i=1; while(i<3) i=i+1;** ?
    1. 1
    2. 2
    3. 3
10. Сколько раз выполнится оператор цикла **int x=3; while(x>1) x=x+1;** ?
    1. Один раз
    2. Ни одного
    3. Бесчисленное число раз
11. Сколько раз выполнится оператор цикла **int x=1; while(x<=3) x=x+1;** ?
    1. 1
    2. 2
    3. 3
12. Чему будет равняться **x** после выхода из цикла **int x=1; do x=x+1; while(x<3);** ?
    1. x=1
    2. x=2
    3. x=3