БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ РАДИОФИЗИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ
КАФЕДРА ИНФОРМАТИКИ И КОМПЬЮТЕРНЫХ СИСТЕМ

Н. В. Серикова

ПРАКТИЧЕСКОЕ РУКОВОДСТВО
к лабораторному практикуму

«ЦИКЛЫ»

по дисциплине

«ПРОГРАММИРОВАНИЕ НА C++»

2024
МИНСК

Практическое руководство к лабораторному практикуму «ЦИКЛЫ» по дисциплине «ПРОГРАММИРОВАНИЕ НА С++» предназначено для студентов, изучающих базовый курс программирования на языке С++, специальностей «Компьютерная безопасность», «Прикладная информатика», «Радиофизика».

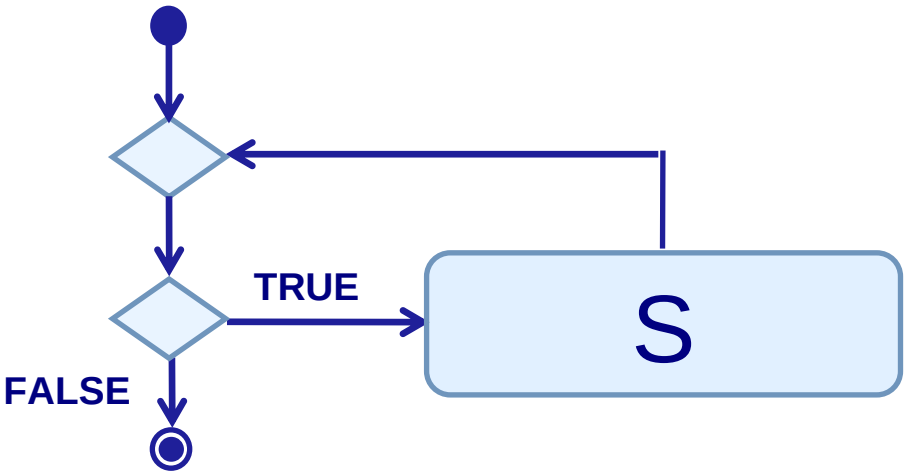
Руководство содержит некоторый справочный материал, примеры решения типовых задач с комментариями.

Автор будет признателен всем, кто поделится своими соображениями по совершенствованию данного пособия.

Возможные предложения и замечания можно присылать по адресу:
E-mail: Serikova@bsu.by

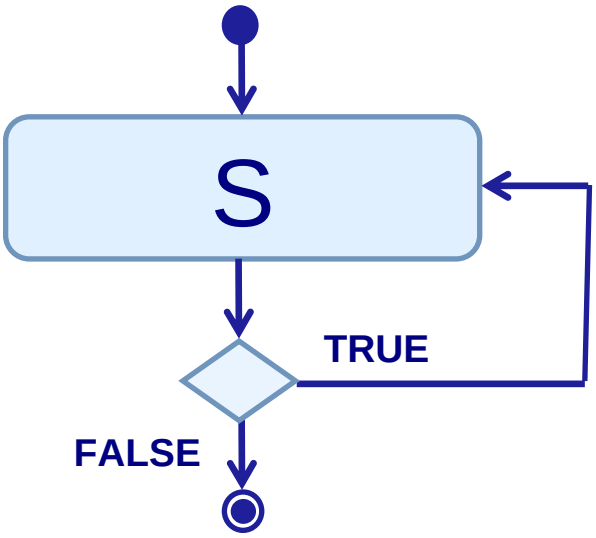
ОГЛАВЛЕНИЕ

ЦИКЛ С ПРЕДУСЛОВИЕМ «WHILE»



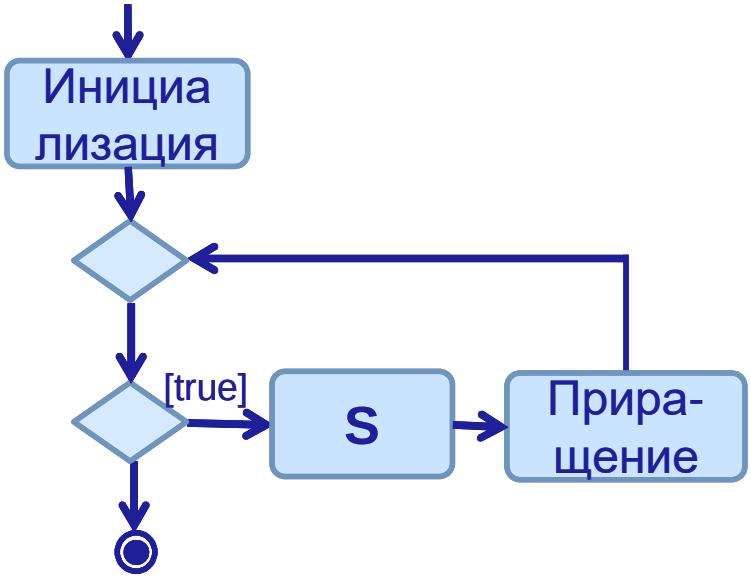
- **Тело цикла может не исполниться ни разу**
- **Требует инициализации переменных условия до входа в цикл**

ЦИКЛ С ПОСТУСЛОВИЕМ «DO WHILE»



- **Тело цикла обязательно исполняется хотя бы один раз**
- **Не требует обязательной инициализации переменных условия до входа в цикл**

ЦИКЛ СО СЧЕТЧИКОМ «FOR»



- **Тело цикла может не исполняться ни разу**
- **Инициализация переменных условия обычно выполняется в заголовке цикла**
- **Не следует изменять переменные условия в теле цикла**

СРАВНЕНИЕ ЦИКЛОВ

Цикл «пока»	Цикл «до тех пор»	Цикл со счетчиком
Инициализация переменных, управляющих условием цикла, должна быть сделана до его начала	Переменные, управляющие условием цикла, могут быть проинициализированы как до начала цикла, так и в его теле	Начальная установка переменных (счетчиков) цикла до заголовка обычно не требуется и выполняется в первой части заголовка цикла (инициализация)
условие выполнения проверяется до входа в цикл	условие выполнения проверяется после первого выполнения тела цикла	условие выполнения проверяется до входа в цикл
В теле цикла должны присутствовать операторы, изменяющие переменные условия, чтобы цикл через некоторое число итераций завершился		Изменение в теле цикла значений переменных, стоящих в его заголовке нежелательно
Цикл работает, пока условие является истинным		
Цикл завершается, когда условие становится ложным		Количество итераций цикла обычно определяется в заголовке значениями нижней и верхней границ счетчика и шага цикла (приращения)
Цикл может не выполниться ни разу, если исходное значение условия при входе в цикл ложно	Цикл обязательно выполняется как минимум один раз	Цикл может не выполниться ни разу, если исходное значение условия при входе в цикл ложно
Если в теле цикла требуется более одного оператора, то необходимо использовать составной оператор (операторные скобки) { }		

Циклы применяются когда:

Цикл «пока»	Цикл «до тех пор»	Цикл со счетчиком
• количество итераций заранее неизвестно • значения переменных, определяющие условие выполнения известны до входа в цикл • тело цикла может не исполняться ни разу	• количество итераций заранее неизвестно • значения переменных, определяющие условие выполнения, вычисляются в теле цикла • тело цикла должно выполняться хотя бы один раз	• количество итераций известно заранее • количество итераций задается изменением переменной-счетчика от начального значения до значения, определяемого условием путем приращения • тело цикла может не исполняться ни разу

ПРИМЕР 1. Область видимости переменной

Пример вывода на экран значений переменной a, которая объявляется в разных областях видимости программы.

```
#include <iostream>  // for cin cout

using namespace std;

int a = 11;                 // глобальная переменная

void main ()
{
     cout<<" 1 a="<<a<<endl;          //          a =11

     int a = 22;
     cout<<" 2 a="<<a<<endl;          //          a=22

     {
          int a = 33;
          cout<<" 3 a="<<a<<endl;     //           a=33
     }

     cout<<" 4 a="<<a<<endl;         //  a=22
                // при возврате в предыдущий блок программы
}
```

ПРИМЕР 2. Оператор расширения области видимости

Пример тот же. Оператор расширения видимости (::)

```
#include <iostream>  // for cin cout

using namespace std;

int a = 11;                    // глобальная переменная

void main ()
{
     cout<<" 1 a="<<a<<endl;                    //     a =11

     int a = 22;
     cout<<" 2 a="<<a<<" "<< ::a <<endl;      //     a=22     11

     {
          int a = 33;
          cout<<" 3 a="<<a<<" "<< ::a <<endl;  //    a=33    11
     }

     cout<<" 4 a="<<a<<"  "<< ::a <<endl;    //      a=22  11
}
```

ПРИМЕР 3. Цикл со счетчиком

```
#include <iomanip>
#include <iostream>  // cin cout
using namespace std;

void main()
{    int  n = 5, sum = 0;
                                    // 1      i++      1..n
     for (int i = 1; i <= n; i++)
     {    cout << setw(5) << i;
          sum += i;
     }
     cout << endl << " s1= " << sum << endl << endl;

                                  // 2      ++i       1..n
     sum = 0;
     for (int i = 1; i <= n; ++i)
     {    cout << setw(5) << i;
          sum += i;
     }
     cout << endl << " s2= " << sum << endl << endl;

                                          // 3     0..n-1
     sum = 0;
     for (int i = 0; i < n; i++)
     {    cout << setw(5) << i;
          sum += i;
     }
     cout << endl << " s3= " << sum << endl << endl;
                                           // 4    10..n-1
     sum = 0;
     for (int i = 10; i < n; i++)
     {    cout << setw(5) << i;
          sum += i;
     }
     cout << endl << " s4= " << sum << endl << endl;
                                       // 5      нет тела цикла
     sum = 0;
     for (int i = 1; i <= n; i++);
     {    cout << setw(5) << i;
          sum += i;
     }
     cout << endl << " s5 = " << sum << endl << endl;
}
```

ПРИМЕР 4. Цикл while

Примеры аналогичные 1 только через while

```
#include <iomanip>
#include <iostream>  // cin cout
using namespace std;

void main ()
{
     int  n = 5, sum = 0, i = 1;
                                        // 1    i++         1..n
     while (i<=n)
     {
        cout << setw(5)<<i;
        sum += i++;
     }
     cout << endl <<" s1= " << sum << endl << endl;

                                       // 2   ++i         1..n
     i = 1; sum = 0;
     while (i <= n)
     {
         cout << setw(5) << i;
         sum += ++i;
     }
     cout << endl <<" s2= "<< sum <<endl <<endl;

                                        // 3      0..n-1
     i = 0; sum = 0;
     while (i < n)
     {
        cout << setw(5) << i;
        sum += i++;
     }
     cout << endl << " s3= " << sum << endl << endl;

                                    // 4 i++        10..n-1
     sum = 0;  i = 10;
     while (i < n)
     {
        cout << setw(5) << i;
        sum += i++;
     }
     cout << endl << " s4= " << sum << endl << endl;
```

```
                                        // 5 нет тела цикла
    sum = 0; i = 1;
    while (i <= n);
    {
      cout  << setw(5) << i;
      sum += i++;
    }
    cout << endl << " s5 = " << sum << endl << endl;
}
```

ПРИМЕР 5. Цикл do while

Примеры аналогичные 1 только через do while

```
#include <iomanip>
#include <iostream>  // cin cout
using namespace std;

void main ()
{    int  n = 5, sum = 0, i = 1;

                              // 1    i++          1..n
     do
     { cout << setw(5) << i;
       sum += i++;
     }
     while (i <= n);
     cout << endl << " s1= " << sum << endl << endl;

                               // 2     ++i          1..n
     i = 1; sum = 0;
     do
     { cout << setw(5)<<i;
        sum += ++i;
     }
     while (i <= n);
     cout << endl << " s2= " << sum << endl << endl;

                                 // 3               0..n-1
     i = 0; sum = 0;
     do
     { cout << setw(5)<<i;
        sum += i++;
     }
     while (i < n);
     cout << endl << " s3= " << sum << endl << endl;

                                    // 4             10..n-1
     sum = 0; i = 10;
     do
     {    cout << setw(5)<<i;
          sum += i++;
     }
     while (i<n);
     cout << endl << " s4= " << sum << endl << endl;
```

```
                                    // 5          нет тела цикла
     sum = 0;
     i = 1;
     do;
     {    cout << setw(5)<<i;
          sum += i;
     }
     while (i <= n);
     cout << endl << " s5 = " << sum << endl << endl;
}
```

ПРИМЕР 6. Область видимости переменной

Пример вывода на экран значения переменной a, которая объявляется в разных разных областях видимости программы. Пример аналогичен 1, добавлен цикл с параметром a.

```
#include <iostream>  // for cin cout

using namespace std;

int a = 11;                 // глобальная переменная

void main ()
{
     cout<<" 1 a= "<<a<<endl;              //    a=11

     int a = 22;
     cout<<" 2 a= "<<a<<endl;              //    a=22
     {
          int a = 33;
          cout<<" 3 a= "<<a<<endl;         //    a=33
          {
               for(int a = 0; a < 5; a++)
                  cout<<" 4 a= "<<a;        //    a=0 1 2 3 4
               cout<<endl;
          }
          cout<<" 5 a= "<<a<<endl;          //   a=33
     }
     cout<<" 6 a= "<<a<<endl;               //   a=22
}
```

ПРИМЕР 7. Натуральные делители

Дано натуральное число n. Получить все его натуральные делители

```
#include <iostream>  // cin cout

using namespace std;

void main ()
{
     int  n;

     cout << " n=";
     cin >> n;
                        // перебор значений для делителей
     for (int k = 2; k <= n / 2;  k++)
        if  (!(n  %  k))     //  условие делимости  n на k
              cout << k << "  ";

     cout << endl;
}
```

ПРИМЕР 8. Перебор значений

Дано натуральное число n. Получить все такие натуральные q, чтобы n делилось нацело на q и не делилось нацело на $2q$.

```
#include <iostream>  // cin cout

using namespace std;

void main ()
{
     int n;

     cout << " n= ";
     cin >> n;

                              // перебор значений  q
     for (int q = 1; q <= n / 2; q++)
     {     // условие делимости на q
           //  и неделимости на 2q
        if  (!(n  % q) && (n  % (2 * q)) )
           cout << q << endl;
     }
}
```

ПРИМЕР 9. Табулирование функции

Построить таблицу значений функции y=sin(x), аргумент x изменяется от *a* до *b* с шагом *h*.

```
#include <math.h>          //  sin
#include <iomanip>         //  setw
#include <iostream>        //  cin cout

using namespace std;


void main ()
{
     double  a, b, h, x;

                        // ввод  значений  a  b   h
     cout << " a=";
     cin >> a;
     cout << " b=";
     cin >> b;
     cout << " h=";
     cin >> h;

     cout << setw(15) << "  x  " << setw(15) << " y "<< endl;

     x = a;                    // начальное значение аргумента
     while (x < b + h / 2)     // условие для выхода из цикла
     {                      //  вывод значения  функции на экран
          cout << setw(15) << x << setw(15) << sin(x) << endl;
          x += h;     // изменение аргумента на величину шага h
     }
}
```

ПРИМЕР 10. Вычисление НОД двух чисел (по алгоритму Евклида)

```
#include <iostream>
using namespace std;
void main()
{
     int a,b,m,n;
     cin>>m;  cin>>n;

     a = m; b = n;
     if (a > 0 && b > 0) //Алгоритм Евклида с вычитанием
     {    while ( a != b)
          {
             if (a > b) a -= b;
             else       b -= a;
           }
          cout<<a<<endl;
     }
     else     cout<<"Error"<<endl;

     a = m;   b = n;
     if (a > 0 && b > 0)// Алгоритм Евклида с делением
     {    while ( a && b)
          {
            if (a > b) a %= b;
            else       b %= a;
           }
          cout<<a+b<<endl;
     }
     else     cout<<"Error"<<endl;

     a = m;   b = n;
     if (a > 0 && b > 0)
                    // или так Алгоритм Евклида с делением
     {  int c;
        while (b)
        {
          c = a % b;
          a = b;
          b = c;
        }
        cout<<a<<endl;
     }
     else     cout<<"Error"<<endl;
}
```

ПРИМЕР 11. Выделение цифр целого числа

Верно ли, что заданное натуральное число содержит ровно одну цифру 7?

```
#include <iostream>  // cin cout
using namespace std;

void main()
{    int   n, m, N, i ;


     cout<<" vvod n=";
     cin>>n;

     m = n;
     N = 0;
     i = 0;

     while(m > 0)
     {
       int c = m % 10;              // цифра
       m = m / 10;
       i++;                     // количество цифр числа
       cout << c << endl;
       N += int (c == 7);
      }

     if (N == 1 ) cout << "Yes" << endl;
     else         cout  << "No"<< endl;
}
```

ПРИМЕР 12. Пошаговый ввод данных

Вводится последовательность вещественных чисел. Количество чисел заранее неизвестно. Ввод прекращается, если вводится значение 100. Найти порядковый номер максимального из введенных чисел. Если максимальных значений несколько, то найти первое из них.

```
#include <iostream>  // cin cout

using namespace std;

void main ()
{
     double x, max;
     int i, j;

     cout << " x= ";
     cin >> x;              // ввод первого числа
     if ( x != 100)
     {
          max = x; i = 1;  j = 1;
          do
          {
               if (x > max)
               {
                    i = j;          // номер максимального числа
                    max = x;        //   максимальное число
               }
               j++;                 // счетчик введенных чисел
               cout << " x= ";
               cin >> x;      // ввод следующего значения
          }
          while (x != 100);       // условие выхода из цикла

          cout  << " N = " << i << " Max = " << max << endl;
     }
}
```

ПРИМЕР 13. Определение значения в заданном диапазоне

Определить наибольшую степень числа 10, которую можно вычислить, пользуясь типом long

```
#include <LIMITS.H>              //LONG_MAX
#include <iostream>             // cin cout

using namespace std;


void main ()
{
    float x = 1, p = 1;

    cout << "LONG_MAX=" << LONG_MAX << endl;

    while (x < LONG_MAX)        // x< 2147483648
    {
        x *= 10;
        cout << "10 v stepeni " << p << " =  " << x << endl;
        p++;
    }

    cout << "result " << x / 10 << " step " << p -1 <<endl;

}
```

ПРИМЕР 14. Вычисление значения многочлена

$$p(x) = 2^5 x^5 + 2^4 x^4 + 2^3 x^3 + 2^2 x^2 + 2^1 x^1 + 2^0$$

```
#include <iostream>  // cin cout

using namespace std;

void main ()
{
     double x, xPower, result, dwaPower;
     int n = 5;

     cout << " x = ";
     cin >> x;

               // начальное значение  суммы
     result = 1 + 2 * x;
              // начальное значение  переменной для степени x
     xPower = x;
              // начальное значение  переменной для степени 2
     dwaPower = 2;


     for (int i = 2; i <= n; i++)
     {
          xPower *=  x;                  // степени x
          dwaPower *= 2;                 //  степени 2
          result = result + dwaPower * xPower;  //   сумма
     }

     cout  <<  result << endl;
}
```

ПРИМЕР 15. Вычисление значения многочлена по схеме Горнера

$$p(x) = 2^5 x^5 + 2^4 x^4 + 2^3 x^3 + 2^2 x^2 + 2^1 x^1 + 2^0$$

Схема Горнера для вычисления этого многочлена:

$$p(x) = (((((2^5)x + 2^4)x + 2^3)x + 2^2)x + 2^1)x + 2^0$$

```
#include <iostream>  // cin cout

using namespace std;

void main ()
{
     double x, result;
     int n = 5, dwaPower;

     cout << " x= ";
     cin >> x;

     result = 32.0;
     dwaPower = 32;
     for (int i = n - 1; i >= 0; i--)
     {            //  коэффициенты   многочлена (степени 2)
          dwaPower >>= 1;
          result = result * x + dwaPower;
     }
     cout << result << endl;


                                          //   или так
     result = 0;
     dwaPower =32;
     for (int i = n; i > 0;  i--)
     {
          result = (result + dwaPower) * x;
                  //  коэффициенты   многочлена (степени 2)
          dwaPower >>= 1;
     }
     cout << result +1 << endl;
}
```

ПРИМЕР 16. Цикл со счетчиком. Особенности использования

```
#include <iomanip>
#include <iostream>  // cin cout

using namespace std;

void main ()
{
     int  n = 5,sum = 0;
                                                  // 1
     for (int i = 1; i <= n; i = i + 2)// шаг изменения +2
     {    cout << setw(5) <<i;
          sum += i;
     }
     cout << endl << " s1 =  " << sum << endl << endl;

                                                   // 2
     sum = 0;
     for (int i = n; i > 0; i = i - 2) // шаг изменения -2
     {    cout << setw(5) << i;
          sum += i;
      }
     cout<< endl << " s2 =  " << sum << endl << endl;


          // 3 разные параметры  цикла: цикл не выполняется
      sum = 0;
     for (int i = 1, j = 7; j <= n; i++)
     {    cout << setw(5) << i;
          sum += i;
      }
     cout << endl << " s3 = " << sum << endl << endl;

               // 4 разные параметры  цикла: цикл бесконечный
    sum = 0;
    for (int i =1, j = 1; j <= n;  i++)
    {    cout << setw(5)<<i;
         sum += i;
     }
    cout << endl << " s4 =  " << sum << endl << endl;



                      // 5 символьный параметр цикла
```

```
    cout << " 5 " << endl;
    for (char c = 'a'; c <=  'z';   c++)
        cout << setw(2) << c;
    cout<<endl<<endl;

                  // 6  символьный параметр цикла

    cout << " 6 " << endl;
    for (char c = 'Z'; c >= 'A';  c--)
        cout << setw(2) << c;
    cout << endl << endl;

                     // 7 вещественный параметр цикла
    cout << " 7 " << endl;
    for (double f =1.0; f < 10.0; f += 0.2)
        cout << setw(10) << f;
    cout << endl << endl;

}
```

ПРИМЕР 17. Инструкции break, continue, exit, return

```
#include <iomanip>
#include <process.h>                        // for exit
#include <iostream>  // cin cout
using namespace std;

void main ()
{    int  n = 5;
                                                    // 1
     cout << " 1" << endl;
     for (int i = 0; i < n;  i++)
     {    cout << setw(5) << i << " 1 " << endl;
          cout << setw(5) << i << " 2 " <<endl;
     }
                                                    // 2
     cout << endl << " 2"<<endl;
     for (int i = 0; i < n;  i++)
     {    cout << setw(5)<< i << " 1 " << endl;
          if (i == 1) break;              // выход из цикла
          cout << setw(5) << i << " 2 " << endl;
     }
                                                       // 3
     cout << endl << " 3"<<endl;
     for (int i = 0; i < n;  i++)
     {    cout << setw(5) << i << " 1 " << endl;
          if (i == 1) continue; // прерывается итерация цикла
          cout << setw(5) << i << " 2 " << endl;
     }
                                                        //4
     cout << endl << " 4" << endl;
     for (int i = 0; i < n;  i++)
     {    cout << setw(5) << i << " 1 " << endl;
          if (i == 1) exit(0);           //  выход из программы
          cout << setw(5) << i << " 2 " << endl;
     }
                                                        //5
     cout<<endl<<" 5"<<endl;
     for (int i = 0; i < n; i++)
     {    cout << setw(5) << i << " 1 " << endl;
          if (i == 1) return;         //  выход из программы
          cout << setw(5)<< i<< " 2 "<<endl;
     }
     cout<<endl;
}
```

ПРИМЕР 18. Простое число

Проверить, является ли число *n* простым

```
#include <iostream>                 // cin cout
using namespace std;

void main ()
{   unsigned int  n, m;
    bool pr;

    cout << " n=";
    cin >> n;
                        // 1
    pr = true;          //    признак того, что делителей нет
                        //    перебор значений для делителей
    for (int k = 2; k <= n / 2;  k++)
    {
        if (!(n % k))
        {
            pr = false;
            break;
        }
    }
    cout << pr << endl;

    pr = true                       // 2 или так
    for (int k = 2; (k <= n / 2) && pr;  k++)
        if (!(n % k))
           pr = false;
    cout << pr << endl;

                                    //3  или так
    m = int(sqrt(n));
    if (n == 1) pr =  false;
    else if (n == 2) pr = true;
    else if (!(n % 2))  pr = false;
    else
    {
        pr = true;
        for (int k =3; k <= m && pr; k = k + 2)
            if (!(n % k))  pr = false;
    }
    cout << pr << endl;
}
```

ПРИМЕР 19. Цикл в цикле

```
#include <iomanip>
#include <iostream>  // cin cout

using namespace std;

void main ()
{
     int  n, m ;
     n = 3; m = 5;

                                                          //1
     for (int i =0; i < n;  i++)
          for (int j = 0; j < m; j++)
               cout<< setw(3) << i * m + j <<setw(5)
                    << i <<setw(5)<< j <<endl;
     cout<<endl;

                                                          //2
     for (int i =0; i < n;  i++)
     {
          cout << " i=" << i;
          for (int j = 0; j < m; j++)
               cout << "   j= " << setw(2) << j;
          cout<<endl;
     }
     cout<<endl;

                                                         //3
     for (int i =0; i < n;  i++)
          cout << " i=" << i;
     for (int j = 0; j < m; j++)
          cout << "   j= " << setw(2) << j;
     cout << endl;
     cout << endl;
}
```

ПРИМЕР 20. Цикл в цикле: break, continue, exit, return

```
#include <iomanip>
#include <process.h>                    // exit
#include <iostream>                     // cin cout
using namespace std;

void main ()
{   int  n = 3,  m = 5;
                                                          //1
    for (int i = 0; i < n; i++)
    {
        cout << " i=" << i;
        for (int j = 0; j < m;  j++)
        {          // выход из внутреннего цикла
            if (i == 1)
                 break;
            cout << "   j= " <<setw(2) <<j;
        }
        cout << endl;
    }
    cout<<endl;
                                                          //2
    for (int i = 0; i < n;  i++)
    {
        cout << " i=" << i;
        for (int j = 0; j < m; j++)
        {       //  прерывается итерация внутреннего цикла
            if (i == 1) continue;
            cout <<"   j= " << setw(2) <<j;
        }
        cout << endl;
    }
    cout<<endl;
                                                          //3
    for (int i =0; i < n;  i++)
    {
        cout << " i=" << i;
        for (int j = 0; j < m; j++)
        {            // выход из внутреннего цикла
            if (j == 1) break;
            cout <<"   j= " << setw(2) << j;
        }
        cout<<endl;
    }
    cout<<endl;
```

```
                                                        //4
    for (int i = 0; i < n;  i++)
    {
        cout << " i=" << i;
        for (int j = 0; j < m;  j++)
        {       //  прерывается итерация внутреннего цикла
            if (j == 1) continue;
            cout << "   j= " << setw(2) << j;
        }
        cout<<endl;
    }
    cout<<endl;

                                                          //5
    for (int i = 0; i < n;  i++)
    {
        cout << " i=" << i;
        for (int j =0; j < m;  j++)
        {
            if (j == 1) exit(0);      // выход из программы
            cout << "   j= " << setw(2) << j;
        }
        cout << endl;
    }
    cout<<endl;


                                                    //6
    for (int i =0; i < n; i++)
    {
        cout << " i=" << i;
        for (int j = 0; j < m; j++)
        {
            if (j == 1) return;    // выход из программы
            cout << "   j= " << setw(2) <<j;
        }
        cout<<endl;
    }
    cout<<endl;
}
```

ПРИМЕР 21. Таблица умножения

Вывести на экран таблицу умножения.

```
#include <iomanip>
#include <iostream>  // cin cout

using namespace std;


void main ()
{
     cout << setw(3) << "   |  ";

     for (int i = 1; i <= 10;  i++)
          cout << setw(3) << i;
     cout<<endl;

     cout <<"-----------------------------------"<<endl;

     for (int i = 2; i <= 9;  i++)
     {
          cout << setw(3) << i << "|  ";

          for (int j = 1; j <= 10;  j++)
               cout << setw(3) << i * j;
          cout<<endl;
     }
}
```

ПРИМЕР 22. Натуральные делители чисел от 2 до *n*.

Вывести на экран натуральные делители всех чисел до заданного *n*.

```
#include <iomanip>
#include <iostream>  // cin cout

using namespace std;


void main ()
{
     int  n;
     cout<<" n=";
     cin >> n;

                        // перебор значений от 2 до n
     for (int m = 2;  m <= n;  m++)
     {
          cout<< " deliteli " << setw(5) << m <<  "  = ";
          for (int k = 2; k <= m / 2;  k++)
               if ( !(m % k) )
                 cout << k << "  ";    // делители числа m
          cout<<endl;
     }
}
```

ПРИМЕР 23. Таблица символов

Вывести на экран таблицу символов.

```
#include <iomanip>
#include <conio.h>      // getche
#include <iostream>     // cin cout

using namespace std;


void main ()
{
    int n = 10,  m = 20;

    cout << setw(4) << "    |  ";

    for (int i = 0; i < n; i++)
        cout << setw(4) << i;
    cout<<endl;

    cout <<"---------------------------------------"<<endl;

    for (int i = 0; i <= m - 1;  i++)
    {   unsigned char c=30;
        cout << setw(4) << c + i * n << "|  ";
        for (int j = 0; j <= n - 1;  j++)
            cout << setw(4) << char(c + i * n + j);
        cout<<endl;
    }
    cout<<endl<<endl;
}
```

ПРИМЕР 24. Оптимизация вычислений

Найти сумму ряда для заданного *n*.

$$S = \sum_{i=1}^{n} \frac{1}{i!}$$

```
#include <iostream>  //  cin cout

using namespace std;

int main()
{
     unsigned short n;
     double sum = 0, mult;

     cout << "n=";
     cin >> n;

     for (int i = 1; i <= n; i++)
     {
          mult = 1.0; //  плохо  !!!
          for (int j = 2; j <= i; j++)
               mult *= j;
          sum = sum + 1.0 / p;
     }
     cout << "sum= " << sum<<endl;


                                   //  лучше так
     sum = 0;    mult = 1.0;
     for (int i = 1; i <= n; i++)
     {
          mult *= i;              // вычисление факториала
          sum = sum + 1.0 / mult;
     }
     cout << "sum= " << sum<<endl;
     return 0;
}
```

ПРИМЕР 25. Задача нахождения суммы конечного числа членов ряда

Вычислить:

$$S = \sum_{i=0}^{n} (-1)^i \frac{1}{2^i}$$

где

$$(-1)^i = \begin{cases} 1 \ при \ k, \ кратном \ 2; \\ -1 \ при \ k, \ не \ кратном \ 2; \end{cases}$$

$$2^i = 2^{i-1} \cdot 2$$

```
#include <iostream>  // cin cout
using namespace std;

void main()
{    unsigned short n,  i;
     double sum, p;

     cout << "n=";
     cin >> n;

     i = 1; sum = 1; p = 1;
     while (i <= n)
     {    p = -p * 2;
          cout<<p<<endl;
          sum += 1 / p;
          i++;
     }
     cout << "sum = " << sum <<endl;

                                  // или так
     i = 1; sum = 1; p = 1;
     while (i <= n)
     {
          p = -p / 2;
          cout<<p<<endl;
          sum += p;
          i++;
     }
     cout << "sum = " << sum <<endl;

}
```

ПРИМЕР 26. Задача нахождения суммы бесконечного числа членов ряда

Вычислить: $$S = \sum_{i=0}^{\infty} \frac{1}{2^i}$$

```
#include <iostream>      // cin cout

using namespace std;

void main()
{
     unsigned short i = 1;
     double sum = 1.0,  p = 1.0,    eps =  1e-4;

     while (p > eps)
     {
          p = p / 2.0;
          sum = sum + p;
               //      cout<<p<<"    "<< sum <<endl;
          i++;
     }
     cout<< " sum =  " << sum << "  i = " << i <<endl;
}
```

ПРИМЕР 27. Цикл со счетчиком. Параметры

```
#include <iomanip>
#include <iostream>  // cin cout

using namespace std;

void main ()
{
     int  n = 5, x = 0, sum = 0, i = 0;

                          // 1      опущен 1-ый параметр
    for (;   i < n;  i++)
    {  cout << setw(5) << i;
       sum += i;
     }
    cout << endl << " s1= " << sum << endl <<endl;


                         //   2     опущен 1-ый параметр
    i =1;
    for (;  i<n;  i++)
    {  cout << setw(5) << i;
       sum += i;
    }
    cout<< endl << " s2= " << sum << endl << endl;


                        //  3       опущен 3-ий параметр
    for (i = 0, sum = 0; i < n; )
    { cout << setw(5)<<i;
      sum += i;
     }
    cout << endl << " s3 = " << sum << endl << endl;


                         //  4    3-ий параметр не изменяется
    for (i = 0, sum = 0; i < n; i = 3)
    { cout << setw(5)<<i;
      sum += i;
     }
    cout<< endl << " s4 = " << sum << endl << endl;
```

```
                              // 5   3-ий параметр не изменяется
    for (i = 0, sum = 0; i < n; i = 5)
    { cout << setw(5) << i;
      sum += i;
     }
    cout << endl << " s5= " << sum << endl << endl;
                            //  6  опущены все параметры
    sum = 0;
    for (; ;)
    {
       cout << setw(5)<< i;
       sum += i;
    }
    cout << endl << " s6= " << sum << endl <<endl;


                             // 7     опущены все параметры
    sum = 0;
    for (; ; )
    {  cout << setw(5)<<i;
        sum += i;
        if (sum > 20)   break;
     }
    cout << endl << " s7= " << sum << endl << endl;

                            //    8 опущены 1-ый и 3-ий параметры
    sum = 0;
    for (;  sum <= 20; )
    { cout << setw(5)<<i;
       sum += i;
     }
    cout << endl << " s8= " << sum << endl << endl;
}
```

ПРИМЕР 28. Цикл со счетчиком. Сложные параметры

```
#include <iomanip>
#include <iostream>  // cin cout

using namespace std;


void main ()
{
    int  n = 3;

                                                    // 1
    cout << " 1 " << endl;
    for (int i = 0,  j = 10;   i <= n;  i++,  j++)
        cout << setw(5)<<i<<setw(5)<<j<<endl;
    cout<<endl;

                                                    // 2
    cout<<" 2 "<<endl;
    for (int i = 0,  j =10; i <= n;  i++,  j--)
        cout << setw(5)<<i<<setw(5)<<j<<endl;
    cout<<endl;

                                                     // 3
    cout<<" 3 "<<endl;
    for (int i = 0,  j =10;  i <= j;  i++,  j--)
        cout << setw(5)<<i<<setw(5)<<j<<endl;
    cout<<endl;

                                                      // 4
    cout<<" 4 "<<endl;
for (int i = 0, j =10;  (i <= n) || (j < 20);  i--,  j++)
        cout << setw(5)<<i<<setw(5)<<j<<endl;
    cout<<endl;

                                                      // 5
    cout<<" 5 "<<endl;
    for (int i = 0, j =10; (i <= n) && (j < 20);  i--, j++)
        cout << setw(5)<<i<<setw(5)<<j<<endl;
    cout<<endl;
}
```

ПРИМЕР 29. Цикл while. Особенности

```
#include <iomanip>
#include <conio.h>        // getche
#include <stdlib.h>       // rand
#include <iostream>       // cin cout

using namespace std;


void main ()
{
     unsigned char c;

                                                  //1
     c = 30;
     while (c)
     {
          cout << setw(4)<<char(c);
          c++;
     }
     cout<<endl;
     getche();


                                                   //2
     while(rand() % 100 != 25)
          cout<<'.';
     cout<<endl;
     getche();


                                                     //3
     while((c = getche()) != ' ' && c != EOF);
     cout<<endl;
     getche();
}
```

СЛОВАРЬ ПОНЯТИЙ, ИСПОЛЬЗУЕМЫХ В ЗАДАНИЯХ

Натуральное число b является **делителем** натурального числа a, если существует натуральное число c, такое что $a = bc$.

Любое натуральное число a имеет хотя бы 2 делителя: 1 и a. Если число a не имеет никаких других делителей, кроме этих, то будем говорить, что a – **простое**.

Простое число называется **числом Мерсена**, если оно может быть представлено в виде $2^p - 1$, где p – тоже простое число.

НОД – наибольший общий делитель чисел

Взаимно простые числа – числа, которые не имеют общих делителей

НОК – наименьшее общее кратное чисел

Числа Фибоначчи – последовательность чисел, определяемая соотношениями:

$$F(0) = 1; F(1) = 1; F(n) = F(n-1) + F(n-2), n>2.$$

Нулевое и первое значения должны быть заданы, их значения равны 1.

Палиндром - число, десятичная запись которого читается одинаково слева направо и справа налево.

Алгоритмы нахождения делителей числа

Найдем количество делителей и сами делители числа N.

1. Для этого будем перебирать все числа i, подходящие на роль делителя. Очевидно, что 1 <= i <= N. Это самый медленный способ нахождения делителей.

2. Очевидно, что любое число делится на 1 и на себя. Поэтому будем перебирать все числа i, подходящие на роль делителя 2 <= i <= N / 2. Этот способ быстрее.

3. Чтобы ускорить работу алгоритма заметим, что если i – делитель a, то N / i – тоже делитель N, к тому же одно из чисел i или N / i не превышает Sqrt(N) (если предположить противное, то получим N = i * (N / i) > Sqrt(N) * Sqrt(N) = N – противоречие).

Поэтому достаточно перебирать числа i в пределах от 1 до (Sqrt(N)) и при нахождении, что некоторое i – делитель выдавать, что N / i – тоже делитель и увеличивать количество делителей на 2. (Этот алгоритм не будет работать, когда a – точный квадрат, что легко исправляется).

Алгоритмы нахождения простых чисел

Для того, чтобы проверить, является ли число простым, нужно посчитать количество его делителей.

1. Перебираем каждое число от 3 и определяем количество делителей: делим его на все числа от 2 до N-1. Это самый медленный, но самый понятный способ нахождения простого числа.

2. Перебираем каждое число от 3 и делим его на все числа от 2 до SQRT(N) включительно. Этот алгоритм быстрее.

3. Перебираем все нечетные числа, (исключаем числа кратные 5), и делим их на все числа от 3 до SQRT(N) включительно. Числа 2, 3 и 5, - простые числа, определяются до цикла поиска. Этот алгоритм еще быстрее.

4. Перебираем все нечетные числа, (исключаем числа кратные 5), и делим их на все найденные до этого времени простые числа, от 3 до SQRT(N) включительно. Числа 2, 3 и 5, - простые числа, определяются до цикла поиска. Этот алгоритм еще быстрее.