Лекция 6. ЦВП

Козьминых Н.М.

дисциплина «Программирование»

Циклические вычислительные процессы (ЦВП)

- Циклическими называются программы, содержащие циклы.
- Цикл — это многократно повторяемый участок программы.
- Итерация – одно выполнение операторов цикла, повторяемого участка программы.

Организация цикла

- В организации цикла можно выделить следующие этапы:
  - подготовка (инициализация) цикла (И);
  - выполнение вычислений цикла (тело цикла) (ТЦ);
  - модификация параметров (М);
  - проверка условия окончания цикла (ПУ).

Операторы цикла

- Существуют три типа операторов цикла:
  - цикл с предусловием;
  - цикл с постусловием;
  - цикл с параметром.

Операторы цикла
Цикл с предусловием

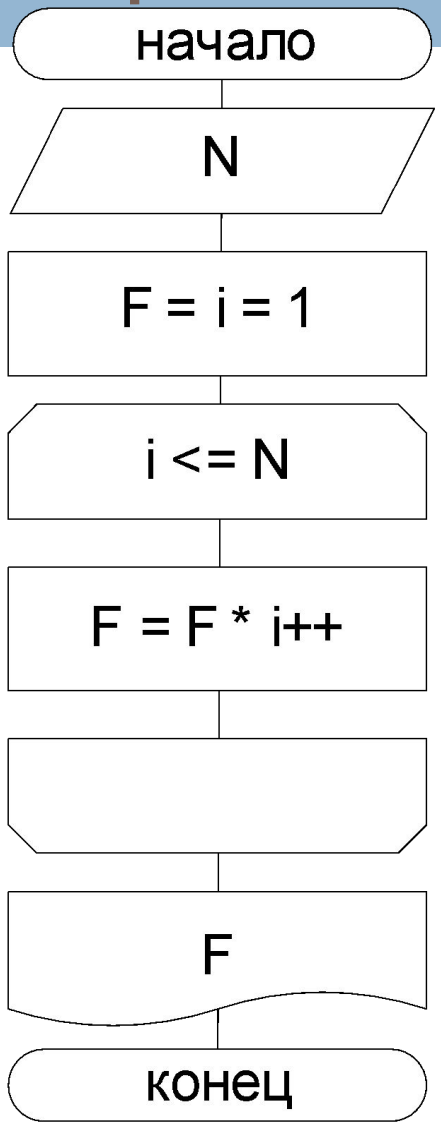

```
while (выражение)
    оператор;
```

- Цикл повторяет свое выполнение, пока значение выражения истинно, т. е. заключенное в нем условие цикла истинно.
- Выход из цикла происходит после того, как значение выражения станет ложным.
- Если выражение ложно, то операторы цикла могут ни разу не выполниться.

Операторы цикла
Цикл с предусловием. Пример 1

- Вычисление факториала целого положительное числа *N*!.

```
1.   private void btnDecide_Click(...){
2.      long  F;
3.      int  i, N;
4.      N = Convert.ToInt32(txtN.Text);
5.      F = i = 1;
6.      while (i <= N)
7.         F = F *i++;    // {F = F * i;
   i++;}
8.      txtResult.Text = N + "! =" + F;
9.   }
```

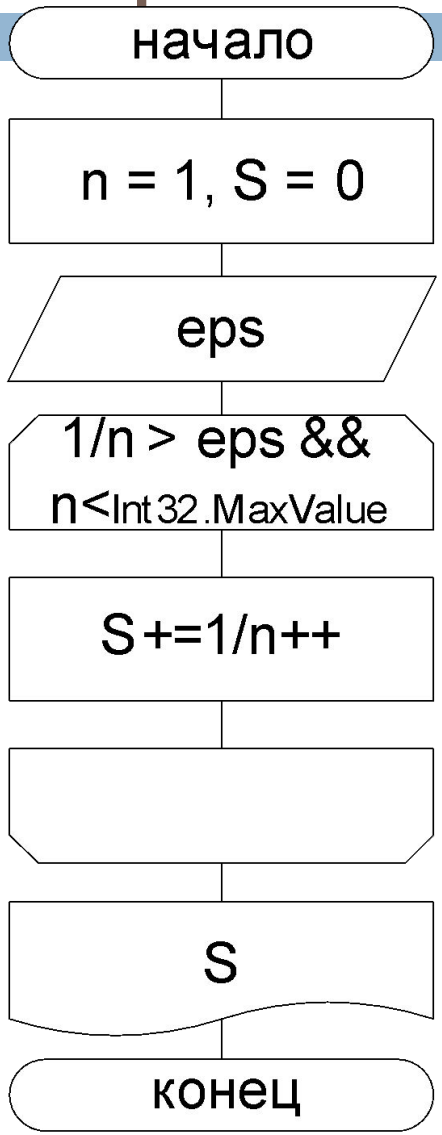


Операторы цикла
Цикл с предусловием. Пример 2

- Задача итерационного вычисления - вычисление суммы гармонического ряда: 1 + 1/2+1/3 + …

  с заданной точностью e

```
1.  private void btnDecide_Click(...){
2.    int  n = 1;
3.    double  S = 0, eps;
4.    eps = Convert.ToDouble(txtEps.Text);
5.    while(1.0/n > eps && n < Int32.MaxValue)
6.        S += 1.0/n++;
7.    txtResult.Text = "Сумма = " + S;
8.  }
```



Операторы цикла
Цикл с постусловием

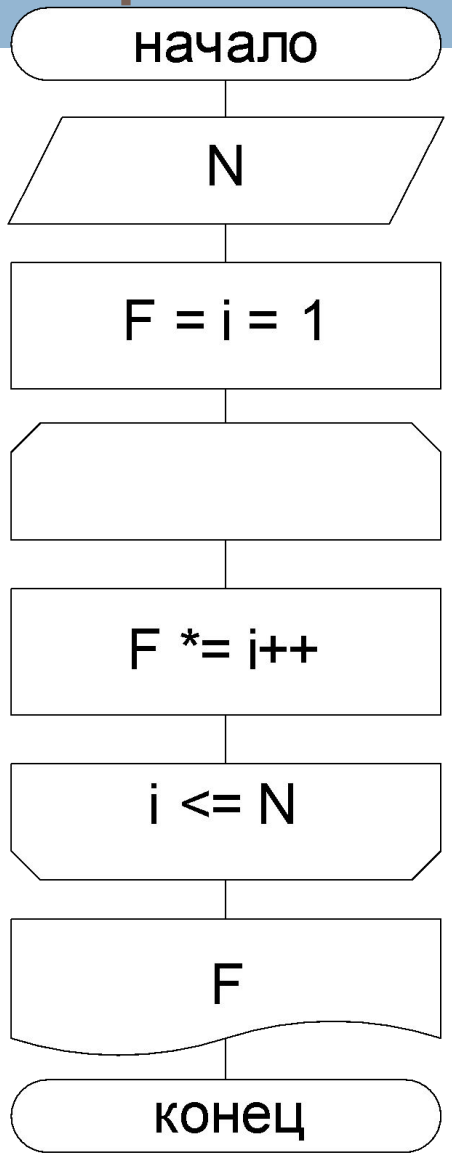

```
do
   оператор;
while (выражение);
```

- Цикл повторяет свое выполнение, пока значение выражения истинно, т. е. заключенное в нем условие цикла истинно.
- Выход из цикла происходит после того, как значение выражения станет ложным.
- Если выражение ложно, то операторы цикла выполняются минимум один раз.

Операторы цикла
Цикл с постусловием. Пример 3

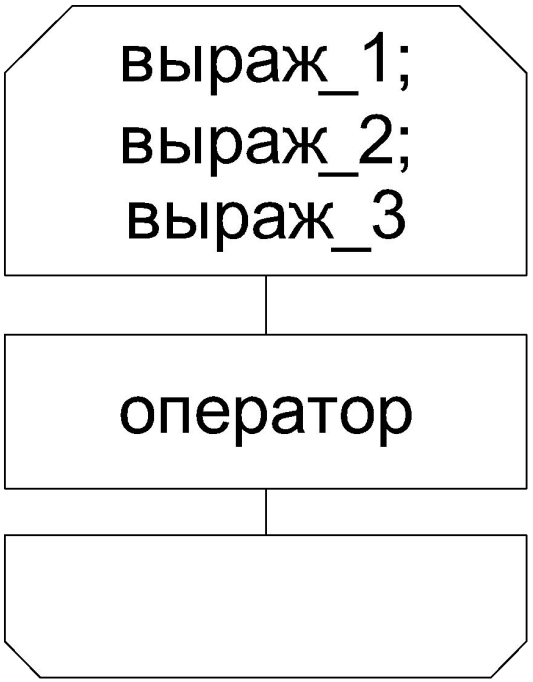
- Вычисление факториала целого положительное числа *N*!.

```
1.    private void btnDecide_Click(...){
2.       long F;
3.       int i, N;
4.       N = Convert.ToInt32(txtN.Text);
5.       F = i = 1;
6.       do
7.          F *= i++;
8.       while(i <= N);
9.       txtResult.Text = N + "! =" + F;
10.   }
```



Операторы цикла. Цикл с параметром


```
for (выраж_1; выраж_2; выраж_3)
    оператор;
```



- Выражение 1 выполняется только один раз в начале цикла, определяет начальное значение параметра цикла.
- Выражение 2 – это условие выполнения цикла.
- Выражение 3 обычно определяет изменение параметра цикла.
- Оператор — тело цикла, которое может быть простым и составным.

Операторы цикла. for и while

- Для сравнения операторs for и while

Операторы цикла. Цикл с параметром. Пример 4

- нахождение *N*!, различное написание:
  - обычное

```
F = 1; for(i = 1; i <= N; i++) F = F * i;
```

  - инициализация нескольких переменных через «,»

```
for(F = 1, i = 1; i <= N; i++) F = F * i;
```

  - с пустыми выражениями

```
F = 1; i = 1; for (; i <= N; i++) F = F * i;
```

  - оператор записан в выражении 2

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

  - оператор записан в выражении 3

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

Операторы цикла. Цикл с параметром. Пример 5

- Задача итерационного вычисления - вычисление суммы гармонического ряда: 1 + 1/2+1/3 + …

  с заданной точностью e

  - обычное

```
for (n = 1, S = 0; 1.0 / n > eps && n < Int32.MaxValue;
  n++)
    S += 1.0 / n;
```

  - с пустым телом цикла

```
for(n = 1, S = 0;
    1.0 / n > eps && n < Int32.MaxValue;
    S+=1.0/n++);
```

Пример 6

- Используя циклы while, do - while и for, написать три варианта программы получения на экране таблицы синусов для значений аргумента в диапазоне от 0 до π/2 с заданным значением шага.

Пример 6. for


```
1.    private void btnDecide_Click(...){
2.        double x, y, h;
3.        h = Convert.ToDouble(txtH.Text);
4.        h = h * Math.PI /180;
5.        lstBox.Items.Add("x \tsin(x)");
6.        for (x = 0; x <= Math.PI / 2; x += h){
7.            y = Math.Sin(x * Math.PI / 180);
8.            lstBox.Items.Add(String.Format("{0:F3}
    \t{1:F3}", x, y));
9.        }
10.   }
```

Пример 6. while


```
1.  private void btnDecide_Click(...){
2.      double x, y, h;
3.      h = Convert.ToDouble(txtH.Text);
4.      h = h * Math.PI /180;
5.      x = 0;
6.      lstBox.Items.Add("x \tsin(x)");
7.      while ( x <= Math.PI / 2){
8.          y = Math.Sin(x * Math.PI / 180);
9.          lstBox.Items.Add(String.Format("{0:F3}
    \t{1:F3}", x, y));
10.         x += h;
11.     }
12. }
```

Пример 6.
do while


```
1.  private void btnDecide_Click(...){
2.      double x, y, h;
3.      h = Convert.ToDouble(txtH.Text);
4.      h = h * Math.PI /180;
5.      x = 0;
6.      lstBox.Items.Add("x \tsin(x)");
7.      do
8.      {
9.          y = Math.Sin(x * Math.PI / 180);
10.         lstBox.Items.Add(String.Format("{0:F3}
    \t{1:F3}", x, y));
11.         x += h;
12.     } while (x <= Math.PI / 2);
13. }
```

Пример 7

- Дано натуральное число n. Вычислить

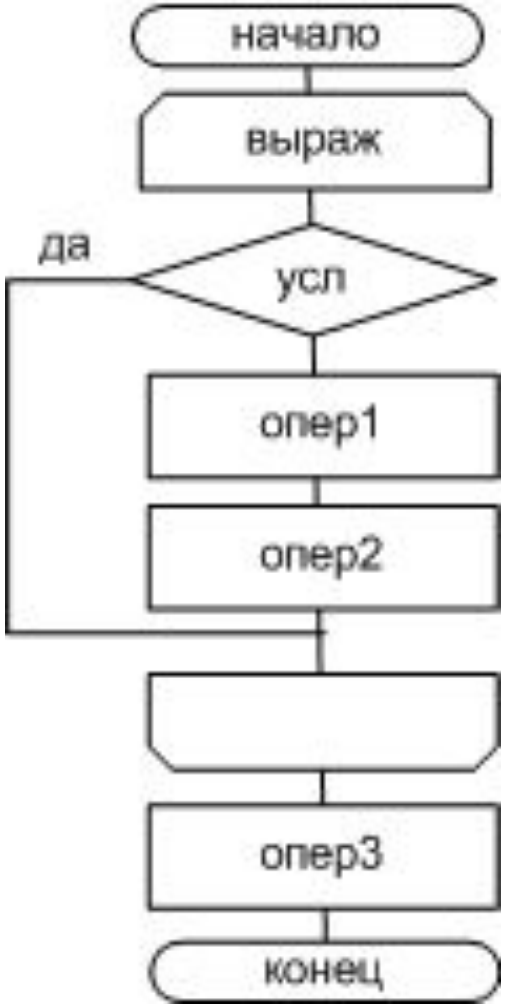
$$\frac{2}{1}+\frac{3}{2}+\frac{4}{3}+\ldots+\frac{n+1}{n}$$

```
1.  private void btnDecide_Click(...){
2.      int n, i;
3.      double sum;
4.      n = Convert.ToInt32(txtN.Text);
5.      sum = 0;
6.      for(i = 1; i <= n; i++)
7.          sum += (i+1.0)/i;
8.  }
```

Операторы перехода (прерывания)

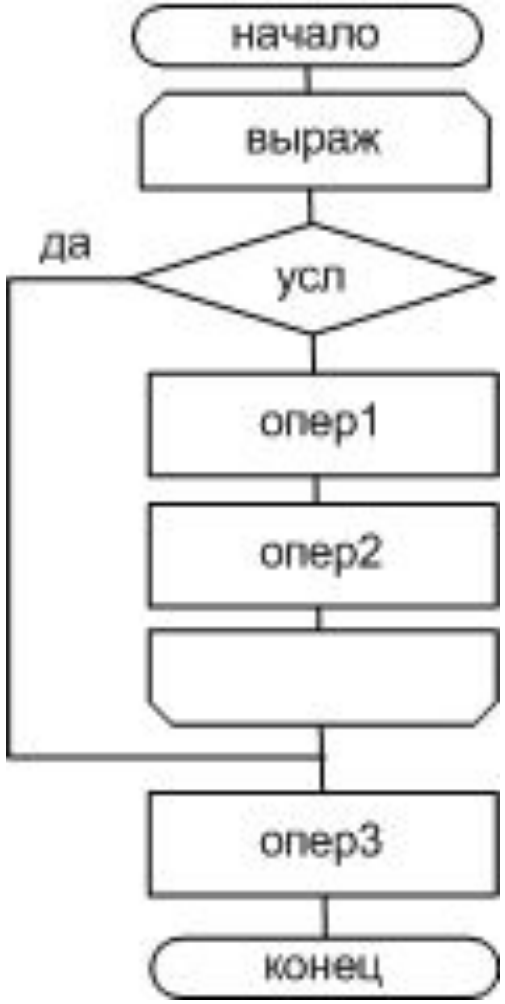
continue

- ***continue*** – прерывание текущей итерации.
- Используется, если выполнение очередного шага цикла требуется завершить до того, как будет достигнут конец тела цикла



Операторы перехода (прерывания) break

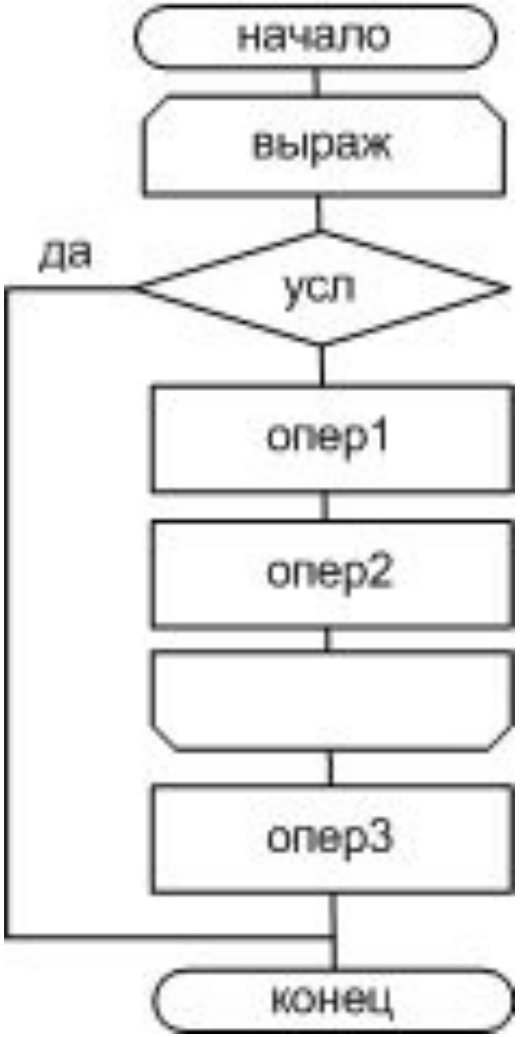
- ***break*** – прерывание цикла.
- ***break*** – прерывание case в операторе switch.
- Используется для альтернативного выхода из самого внутреннего цикла, то есть переходит к первому оператору, следующему за текущим оператором цикла.



Операторы перехода (прерывания)

return

- ***return*** - производит досрочный выход из текущей функции.
- ***return*** - возвращает значение результата функции.



Примеры

№	Задача	Ответ

1	`int k = -3, i;`
	`for (i = 0; i < 5; i++) {`
	`  k++;`
	`}`
		i= k=

2	`int k = 3, i;`
	`for (i = 0; i < 5; i++) {`
	`  if (i == 3)`
	`  continue;`
	`  k++;`
	`}`
		i= k=

3	`int k = 3, i;`
	`for (i = 0; i < 5; i++) {`
	`  if (i == 3)`
	`  break;`
	`  k++;`
	`}`
		i= k=

Спасибо за внимание

Вопросы…