Синтаксис C#

(циклы)

Андрей Голяков

Циклы

Циклы являются такой же важной частью структурного программирования, как условные операторы.

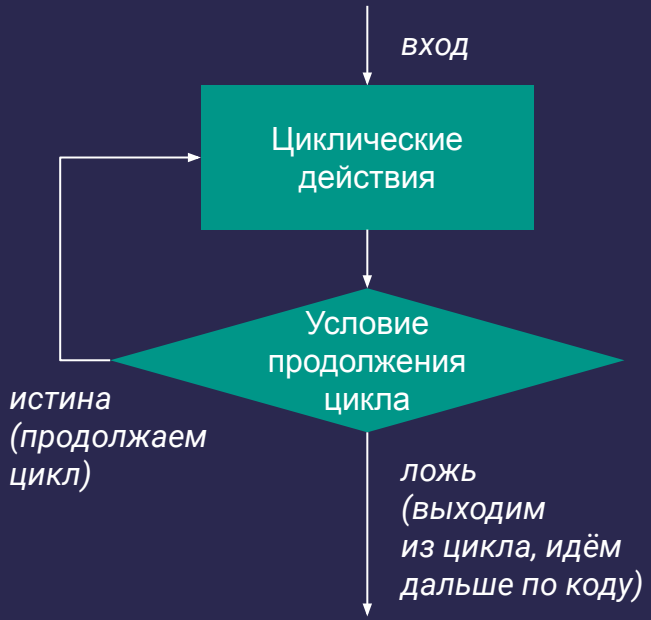
С помощью циклов можно организовать повторение выполнения участков кода.

Обычно разделяют следующие виды циклов:

- Циклы с постусловием
- Циклы с предусловием
- Циклы со счетчиком
- Совместные циклы (по множеству)

Цикл с постусловием: do...while

Оператор do выполняет определенный блок кода, пока условное логическое выражение истинно.



- Так условие этого выражения оценивается после каждого выполнения цикла, цикл do-while выполняется как минимум один раз.
- В любой точке блока операторов цикл do можно разорвать цикл с помощью оператора break.
- Также можно перейти непосредственно к оценке выражения while, воспользовавшись оператором continue.
  Если значение выражения истинно (true), выполнение продолжается с первого оператора тела цикла. В противном случае выполнение продолжается с первого оператора после цикла.

Цикл с постусловием: do...while

```
Console.WriteLine("Enter integer values.");
Console.WriteLine("When Sum exceeds 100 program will be finished:");

int sum = 0;
do
{   // begin of the cycling block
    int i = int.Parse(Console.ReadLine());
    sum = sum + i;
} while (sum < 100); // end of block, check the condition

Console.WriteLine($"The sum is {sum}. Press any key to exit...");
Console.ReadKey();
```

Самостоятельная работа: do...while

Написать программу, которая будет запрашивать строку у пользователя (никуда её не сохраняя) до тех, пока пользователь не введет exit.

Досрочный выход из цикла: break

Команда досрочного выхода break применяется, когда необходимо прервать выполнение цикла, в котором условие выхода ещё не достигнуто (в середине зацикленного кода).

Такое бывает, например, когда при выполнении тела цикла обнаруживается ошибка, после которой дальнейшая работа цикла не имеет смысла.

Другой пример: можно сделать вечный цикл, у которого единственной точкой выхода и будет break:

```
do
{
    // the exit point
    if (someReason) break;
} while (true); // infinite loop
```

Цикл с постусловием: do...while (break)

```
Console.WriteLine("Enter integer values.");
Console.WriteLine("When Sum exceeds 100 program will be finished:");

int sum = 0;
do
{
     try
     {
          var i = int.Parse(Console.ReadLine());
          sum = sum + i;
     }
     catch (FormatException)
     {
          Console.WriteLine("You entered wrong value! Calculation is finished.");
          break;  // stop cycling!
     }
} while (sum < 100);

Console.WriteLine($"The sum is {sum}. Press any key to exit...");
Console.ReadKey();
```

Самостоятельная работа: do...while (break)

Модифицировать программу таким образом, чтобы цикл был вечным, а выход осуществлялся бы с использованием оператора break.

Пропуск итерации: continue

Оператор пропуска итерации continue применяется, когда в текущей итерации цикла необходимо пропустить все команды до конца тела цикла.

При этом сам цикл прерываться не должен, условия продолжения или выхода должны вычисляться обычным образом.

Цикл с постусловием: do...while (continue)

```
Console.WriteLine("Enter integer values.");
Console.WriteLine("When Sum exceeds 100 program will be finished:");

int sum = 0;
do
{
     try
     {
          var i = int.Parse(Console.ReadLine());
          sum = sum + i;
     }
     catch (FormatException)
     {
          Console.WriteLine("You entered wrong value! Please try again:");
          continue;   // start the next cycle iteration!
     }
} while (sum < 100);

Console.WriteLine($"The sum is {sum}. Press any key to exit...");
Console.ReadKey();
```

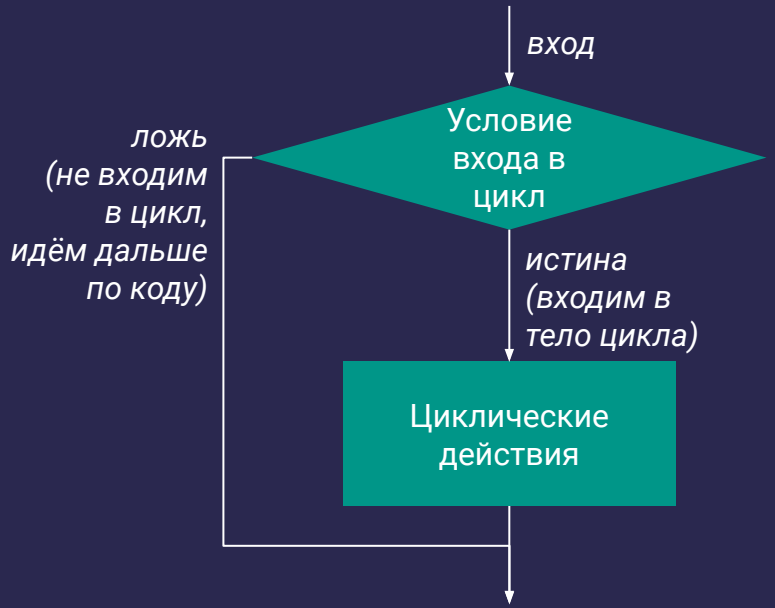
Самостоятельная работа: do...while (continue)

Модифицировать программу таким образом, чтобы на экран выводилась длина введенной строки в формате
“Entered string length is X”
для всех строк, короче или равным 15 символам.

Для строк больше 15 символов должна выводиться надпись “Too long string. Try another:”, и после вывода этой надписи цикл должен возвращался к началу.

Цикл с предусловием: while

Оператор while выполняет определенный блок кода, если условное логическое выражение равно значению true.



- Так как условное выражение оценивается перед каждым выполнением цикла, цикл while выполняется 0 или несколько раз.
- В любой точке блока операторов do можно разорвать цикл с помощью оператора break.
- Можно перейти непосредственно к оценке выражения while, воспользовавшись оператором continue.
  Если значение выражения оценивается как true, выполнение продолжается с первого оператора цикла. В противном случае выполнение продолжается с первого оператора после цикла.

Цикл с предусловием: while

```
Console.WriteLine("Enter integer values.");
Console.WriteLine("When Sum exceeds 100 program will be finished:");

int sum = 0;
while (sum < 100)   // now condition checks before the cycling
{
     try
     {
          var i = int.Parse(Console.ReadLine());
          sum = sum + i;
     }
     catch (FormatException)
     {
          Console.WriteLine("You entered wrong value! Please try again:");
          continue;   // return to the beginning of the cycle!
     }
}

Console.WriteLine($"The sum is {sum}. Press any key to exit...");
Console.ReadKey();
```

Цикл с предусловием: while (counter)

```
Console.WriteLine("Enter 5 integer values to summarize them:");

int sum = 0;
int num = 0;        // variable to calculate the number of entered values
while (num < 5)     // checking that number of entered values less than 5
{
     try
     {
          var i = int.Parse(Console.ReadLine());
          sum = sum + i;
     }
     catch (FormatException)
     {
          Console.WriteLine($"Aborting! only {num} values were summarized!");
          continue;
     }
     num++;
}
Console.WriteLine($"The sum is {sum}. Press any key to exit...");
Console.ReadKey();
```

Цикл с предусловием: while (infinite loop)

```
Console.WriteLine("Enter 5 integer values to summarize them:");
int sum = 0;
int num = 0;
while (true)              // infinite loop
{
     if (num > 4) break; // checking for exit condition in cycle body
     try
     {
          var i = int.Parse(Console.ReadLine());
          sum = sum + i;
     }
     catch (FormatException)
     {
          Console.WriteLine($"Aborting! only {num} values were summarized!");
          break;
     }
     num++;
}
Console.WriteLine($"The sum is {sum}. Press any key to exit...");
Console.ReadKey();
```

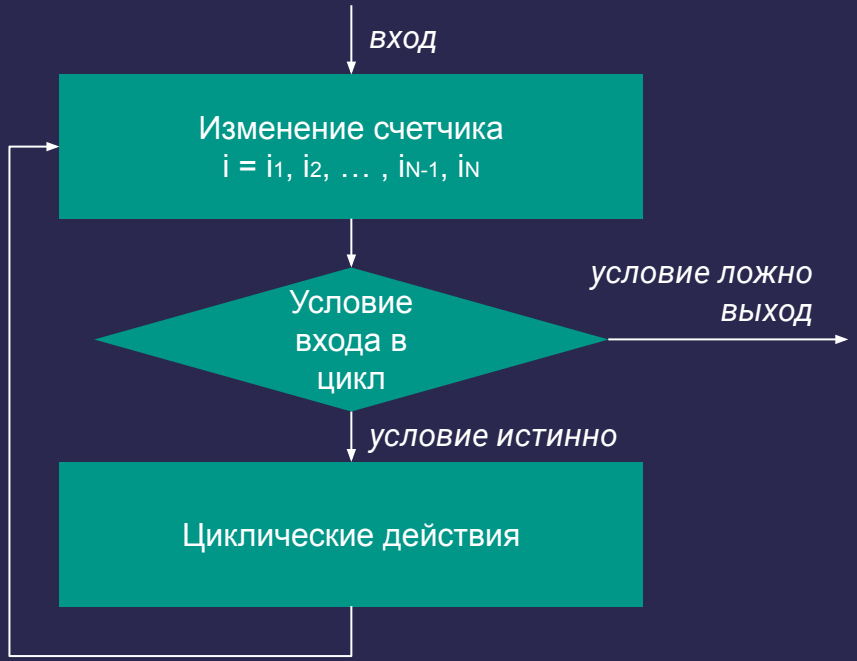
Самостоятельная работа: while

Написать программу, которая бы считала сумму чисел, расположенную в заданном массиве используя цикл while.

В каждой итерации цикла необходимо выводить промежуточный результат подсчета суммы.

Цикл со счетчиком: for

Оператор for выполняет определенный блок кода, каждую итерацию изменяя значение счётчика, пока условное логическое выражение истинно



- Так как условное выражение оценивается перед каждым выполнением цикла, цикл for выполняется 0 или несколько раз.
- В любой момент в блоке операторов for вы можете прервать цикл с помощью оператора break или перейти к следующей итерации в цикле с помощью оператора continue.

Цикл со счетчиком: for

```
// Common structure
// for (initializer; condition; iterator)
//    cycle_body

for (int i = 0; i < 5; i++)
{
    Console.WriteLine(i);
}

Console.WriteLine("Press any key to exit...");
Console.ReadKey();

// 0
// 1
// 2
// 3
// 4
```

Цикл со счетчиком: for (with arrays)

```
Console.WriteLine("Enter 5 integer values:");
int[] numbers = new int[5];

for (int i = 0; i < 5; i++)
{
    var num = int.Parse(Console.ReadLine());
    numbers[i] = num; // access to the element of array by its number
}

Console.WriteLine("The entered values are: " + string.Join(", ", numbers));

Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```

Цикл со счетчиком: for (internal loops)

```
// Internal loops demo

for (int x = 1; x <= 3; x++)
{
    for (int y = 1; y <= 10; y++)
    {
        Console.WriteLine($"{x} * {y} = {x * y}");
    }
}

Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```

Самостоятельная работа: for

- Дан двумерный массив целых чисел от 1 до 5. Представим, что это оценки, полученные за неделю неким учеником, разбитые по дням недели: 0 - Пн, 1 - Вт, и т.д.:

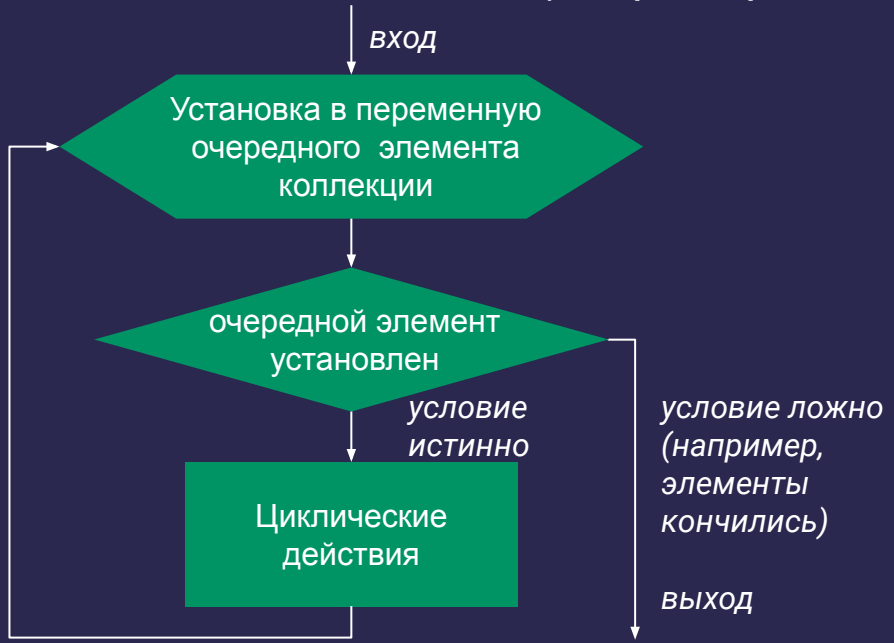
```
var marks = new[]
{
      new [] { 2, 3, 3, 2, 3}, // Monday (it was a good weekend :)
      new [] { 2, 4, 5, 3},    // Tuesday (anyway better than Monday)
      null,                    // Wednesday (felt sick, stayed at home :( )
      new [] { 5, 5, 5, 5},    // Thursday (God mode :)
      new [] { 4 }             // Friday (a very short day)
};
```

- Необходимо написать программу, которая бы посчитала и вывела на экран средний балл по каждому дню и суммарный средний бал за неделю.
- Средние баллы должны быть выведены с точностью до десятых долей.
- Если данных недостаточно, вывести для этого дня “N/A” (not applicable/не применимо).
- Ожидаемый вывод:

```
The average mark for day #0 is 2.6
The average mark for day #1 is 3.5
The average mark for day #2 is N/A
The average mark for day #3 is 5.0
The average mark for day #4 is 4.0
The average mark for all the week is 3.6
```

Итератор: foreach...in

Оператор foreach выполняет определенный блок кода для каждого элемента в коллекции (например, массиве)



- Так как наличие элементов для перебора оценивается перед каждым выполнением цикла, цикл foreach выполняется 0 или несколько раз.

Совместный цикл: foreach...in

```
Console.WriteLine("Enter 5 integer values:");
int[] numbers = new int[5];

for (int i = 0; i < 5; i++)
{
    var num = int.Parse(Console.ReadLine());
    numbers[i] = num;
}

foreach (int number in numbers)
{   // each iteration “number” will be equals to the next item in the array
    Console.WriteLine($"{number}^2 = {number * number}");
}

Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```

Совместный цикл: foreach...in

```
Console.WriteLine("Enter the string for encryption: ");
int encrypctionKey = 1;
string line = Console.ReadLine();

foreach (char letter in line)
{
    Console.Write((char)(letter + encrypctionKey));
}

Console.WriteLine();

Console.WriteLine("Press any key to exit...");
Console.ReadKey();
```

Домашнее задание 1

Написать консольное приложение, которое запрашивает натуральное число и выводит количество четных цифр в нем.

Пример работы программы:

```
> Введите положительное натуральное число не более 2 миллиардов:
> -5 /это ввод пользователя/
> Введено неверное значение! Попробуйте ещё раз:
> 300000000000 /это ввод пользователя/
> Ошибка System.OverflowException! Попробуйте ещё раз:
> ABCD /это ввод пользователя/
> Ошибка System.FormatException! Попробуйте ещё раз:
> 1234567 /это ввод пользователя/
> В числе 1234567 содержится следующее количество четных цифр: 3.
> Нажмите любую клавишу для выхода...
```

Не забывать обрабатывать все предсказуемые исключения.

Домашнее задание 2

Написать консольное приложение, которое запрашивает 1) сумму первоначального взноса, 2) ежедневный процент дохода и 3) желаемую сумму накопления. Программа должна вывести номер дня, когда накопление впервые превысит желаемое.

Пример работы программы (при корректном вводе):

```
> Введите сумму первоначального взноса в рублях:
> 100 /это ввод пользователя/
> Введите ежедневный процент дохода в виде десятичной дроби (1% = 0.01):
> 0.0003 /это ввод пользователя/
> Введите желаемую сумму накопления в рублях:
> 200 /это ввод пользователя/
> Необходимое количество дней для накопления желаемой суммы: 2311.
> Нажмите любую клавишу для выхода…
```

Не забывать обрабатывать все предсказуемые исключения.

Спасибо за внимание.