

# Лабораторная работа №6

**ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ  
АВТОМОБИЛЬНО-ДОРОЖНЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И МЕХАТРОНИКИ**

**Кафедра информационных технологий и мехатроники**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**по проведению лабораторных работ по дисциплине «Алгоритмизация и  
программирование»  
для студентов специальности 6.050101 «Компьютерные науки»**

Разработчик - доцент кафедры информационных технологий и мехатроники  
кандидат технических наук, старший научный сотрудник  
Тимонин Владимир Алексеевич

Харків 2015

## Лабораторная работа №6

### Исследование возможностей интегрированной среды разработки Visual C# для создания приложений циклической структуры (операторы while, do ... while).

**Цель работы** – исследовать возможности интегрированной среды разработки Visual Studio 2010 и получить практические навыки по созданию приложений циклической структуры с использованием операторов while и do ... while.

#### 1. Теоретические сведения

Операторы цикла используются для организации многократно повторяющихся вычислений. Любой цикл состоит из тела цикла, то есть тех операторов, которые выполняются несколько раз, начальных установок, модификации параметра цикла и проверки условия продолжения выполнения цикла (рис. 1).

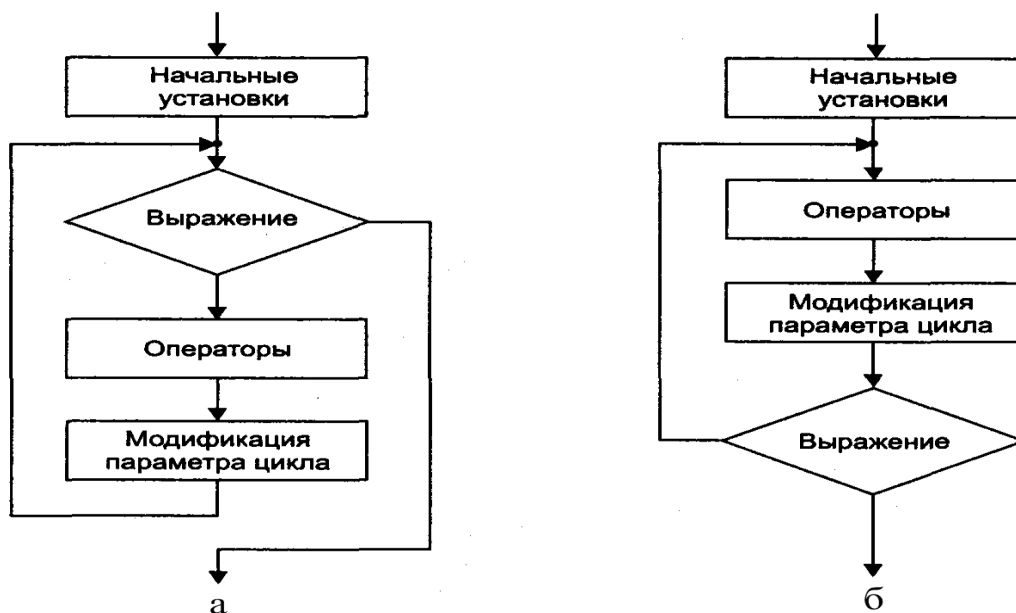


Рис. 1. Структурная схема операторов цикла с предусловием (а) и цикла с постусловием (б)

Один проход цикла называется итерацией. Проверка условия выполняется на каждой итерации либо до тела цикла (тогда говорят о цикле с предусловием), либо после тела цикла (цикл с постусловием). Разница между ними состоит в том, что тело цикла с постусловием всегда выполняется хотя бы один раз, после чего проверяется, надо ли его выполнять еще раз. Проверка необходимости выполнения цикла с предусловием делается до тела цикла, поэтому возможно, что он не выполнится ни разу.

Начальные установки могут явно не присутствовать в программе, их смысл состоит в том, чтобы до входа в цикл задать значения переменным, которые в нем используются.

Переменные, изменяющиеся в теле цикла и используемые при проверке условия продолжения, называются параметрами цикла. Целочисленные параметры цикла, изменяющиеся с постоянным шагом на каждой итерации, называются счетчиками цикла. Цикл завершается, если условие его продолжения не выполняется.

### 1.1. Цикл с предусловием

Цикл с предусловием реализует структурную схему, приведенную на рис. 1.а, и имеет вид:

**while ( выражение ) оператор;**

Выражение определяет условие повторения тела цикла, представленного простым или составным оператором. Выполнение оператора начинается с вычисления выражения. Если оно истинно (равно **true**), выполняется оператор цикла. Если при первой проверке выражение равно **false**, цикл не выполнится ни разу. Тип выражения должен быть арифметическим или приводимым к нему. Выражение вычисляется перед каждой итерацией цикла.



**Пример 1.** В качестве примера создадим консольное приложение, которое вычисляет значение функции  $y = \sin^2(x) + 1$  с заданным шагом  $\Delta x$  в заданном диапазоне аргумента  $x$ .

**1. Анализ задачи.** Исходные данные:  $x_n, x_k$  – вещественные значения угла в градусах, определяющие начальное и конечное значение диапазона,  $\Delta x$  – вещественное значение приращения угла, результат  $y$  – вещественное значение функции. Ввод значений  $x_n, x_k, \Delta x$  осуществляется с клавиатуры. Так как ввод и вывод осуществляется в виде строк, необходима переменная для хранения вводимых и выводимых значений – введем промежуточную переменную **Str**.

Вычислительный процесс – циклический.

**2. Разработка алгоритма.** В алгоритме необходимо выполнить следующие действия:

1. ввод значений диапазона изменения угла и шаг приращения угла  $\Delta x$ ;
2. установка начального значения угла  $x$ ;
3. проверка, находится выбранное значение угла  $x$  в заданном интервале  $x$ ;
4. вычисление значения функции  $y$ ;
5. вывод результата на печать;
6. модификация значения угла.

Алгоритм представляет собой последовательность функциональных блоков: формирования заголовка приложения; ввода значений диапазона ( $x_n, x_k$ ), шага приращения  $\Delta x$  аргумента  $x$  с выдачей соответствующего сообщения и преобразованием введенного значения в число; присвоения начального значения угла  $x$ ; проверки попадания значения  $x$  в соответствующий интервал; вычисления значения функции  $y$ , преобразования результата в строковую переменную и вывода результата; ожидания нажатия клавиши <Enter>. Схема алгоритма приведена на рис. 3.

**3. Разработка программного кода.** Программный код решения задачи представлен на рис. 4. В теле главной функции **Main()** объявляются переменные, используемые в программе:  $x_n, x_k, \Delta x, x, y$  – переменные вещественного типа, переменная **Str** – переменная строкового типа.

Оператор 17 определяет заголовок приложения. Оператор 18 выводит сообщение о необходимом действии пользователя. Оператор 19 записывает значение аргумента  $x_n$  в виде строки в переменную **Str**. Оператор 20 преобразует строковую переменную в вещественное число.

Операторы 21 – 23, 24 – 26 обеспечивают ввод значений  $x_k$  и  $\Delta x$ .

Оператор 27 осуществляет начальную установку параметра цикла  $x$ .

Оператор 28 осуществляют проверку, принадлежит значение аргумента  $x$  заданному диапазону. Если значение  $x$  принадлежит соответствующему интервалу, то выполняется тело цикла (операторы 29 – 34). Оператор 30 вычисляет значение функции  $y$ . Оператор 31 преобразует результат вычисления (вещественное число) в строковую переменную **Str**. Оператор 32 выдаёт результат на экран монитора. Оператор 33 модифицирует значение параметра цикла (текущее значение угла увеличивается на шаг приращения).

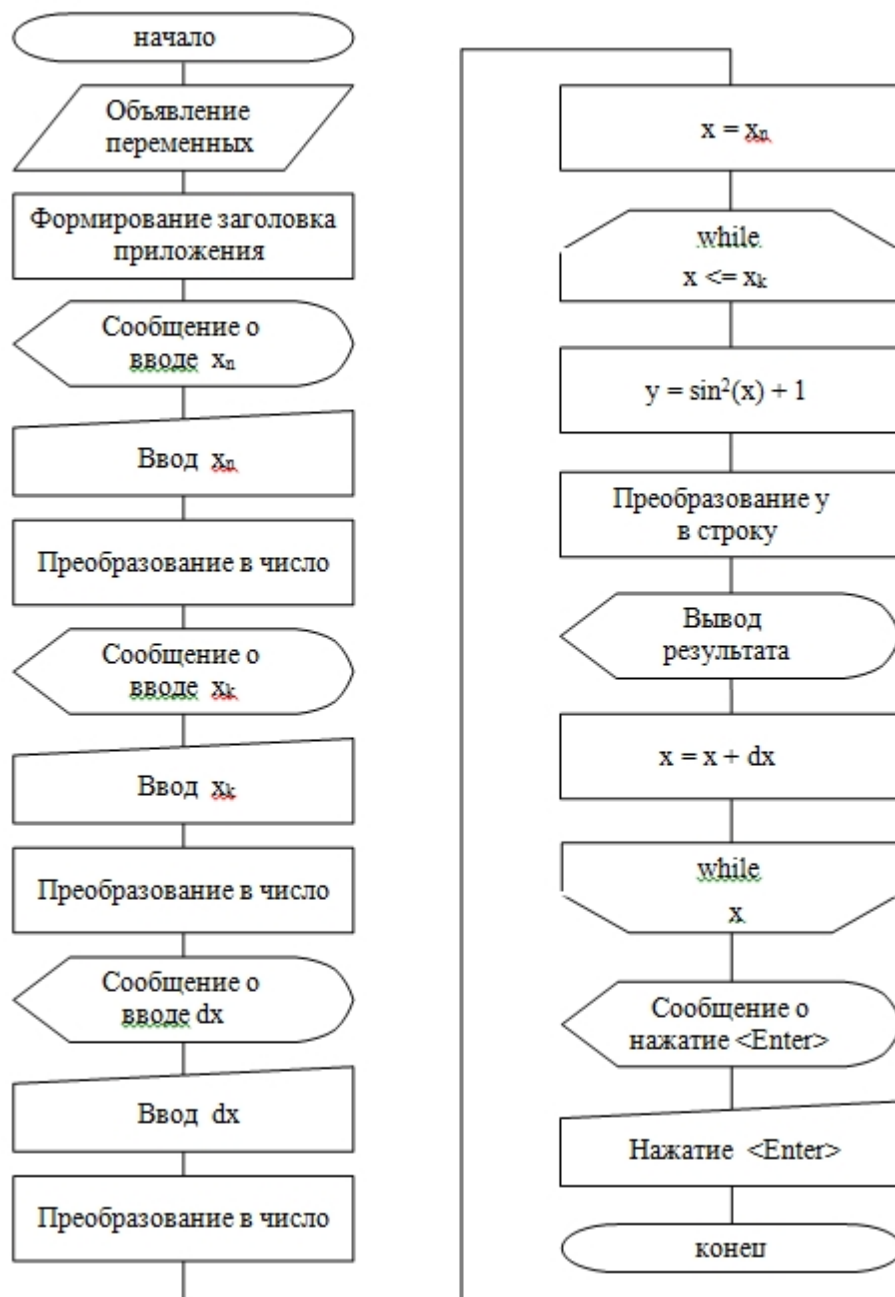


Рис. 3. Блок-схема алгоритма

Если в результате проверки (оператор 28) начальное или модифицированное значение угла  $x$  не принадлежит интервалу, т.е.  $x > x_k$ , управление передается оператору 35.

Один из вариантов результата решения задачи представлен на рис. 5.

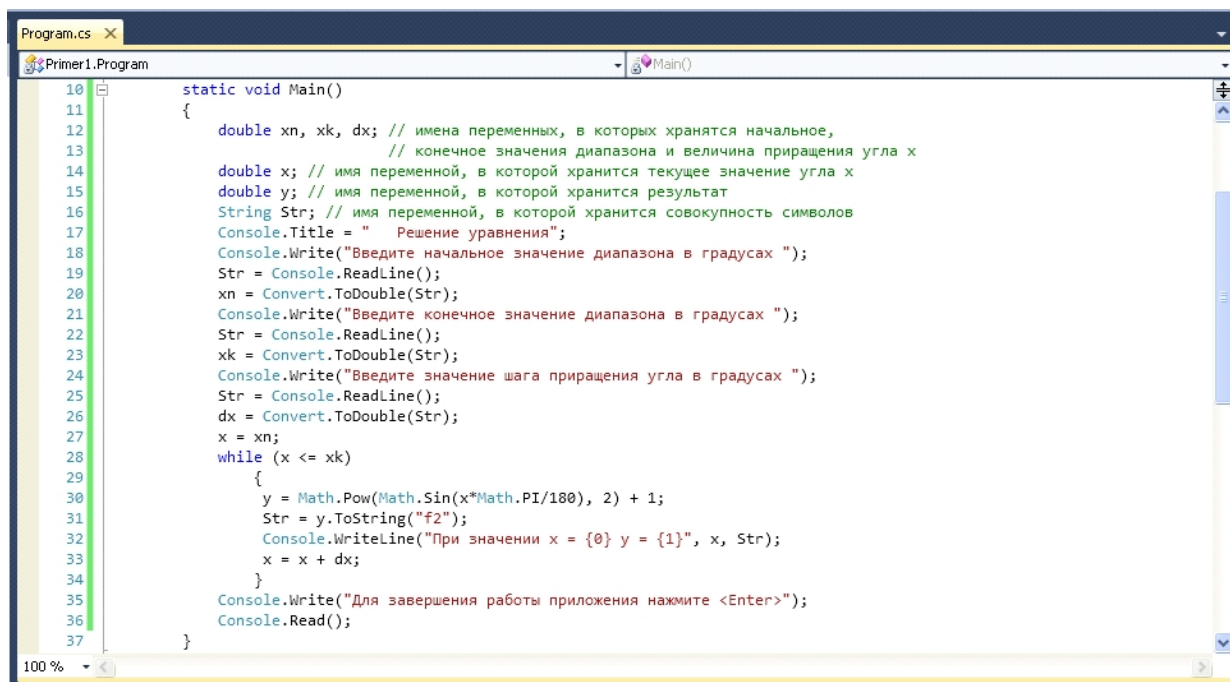


Рис. 4. Программный код решения уравнения

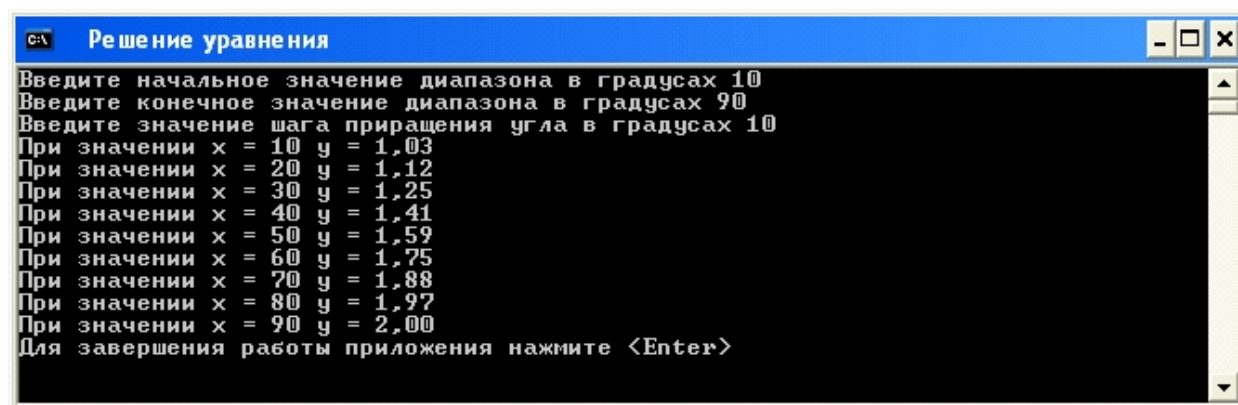


Рис. 5. Результат решения уравнения

Распространенный прием программирования — организация бесконечного цикла с заголовком **while (true)** либо **while (1)** и принудительным выходом из тела цикла по выполнению какого-либо условия.

## 1.2. Цикл с постусловием

Цикл с постусловием реализует структурную схему, приведенную на рис. 1.б, и имеет вид:

**do оператор while (выражение);**

Сначала выполняется простой или составной оператор, составляющий тело цикла, а затем вычисляется выражение. Если оно истинно (равно **true**), тело цикла выполняется еще раз. Цикл завершается, когда выражение станет равным **false** или в теле цикла будет выполнен какой-либо оператор передачи управления. Тип выражения должен быть арифметическим или приводимым к нему.



**Пример 2.** Нижеприведенный код консольного приложения вычисляет квадратный корень вещественного аргумента  $x$  с заданной точностью  $\epsilon$  по итерационной формуле:

$$y_n = \frac{1}{2} \left( y_{n-1} + \frac{x}{y_{n-1}} \right),$$

где  $y_{n-1}$  — предыдущее приближение к корню (в начале вычислений выбирается произвольно),  $y_n$  — последующее приближение. Процесс вычислений прекращается, когда приближения станут отличаться друг от друга по абсолютной величине менее, чем на величину заданной точности. Для вычисления абсолютной величины используется стандартная функция **Abs()**.

```
static void Main()
{
    double X;          // имя переменной, в которой хранится аргумент x
    double Eps;        // имя переменной, в которой хранится заданная точность
    double Yp, Y=1;    // имена переменных, в которых хранятся предыдущее и
                      // последующее приближение
    Console.Title = "  Вычисление квадратного корня";
    Console.Write("Введите значение аргумента X ");
    X = Convert.ToDouble(Console.ReadLine());
    Console.Write("Введите значение точности Eps ");
    Eps = Convert.ToDouble(Console.ReadLine());
    do
    {
        Yp = Y;
        Y = (Yp + X / Yp) / 2;
    } while (Math.Abs(Y-Yp)>=Eps);
    Console.WriteLine("Корень из {0:f2} равен {1:f2}", X, Y);
    Console.Write("Для завершения работы приложения нажмите <Enter>");
    Console.Read();
}
```

## 2. Рабочее задание



**Задание 1.** Руководствуясь теоретическим материалом раздела 1 изучить возможности языка C# по созданию приложений циклической структуры и выполнить практически все примеры, описанные в этом разделе.



**Задание 2.** Используя оператор цикла **while**, разработать приложение с заголовком «Оператор while», которое выводит на экран значения функции  $y_i = f(x_i)$ , вычисляемой по формуле:

$$y = \frac{\sqrt{1+|x|}}{1-x} + \begin{cases} \sin\left(\frac{x+1}{4}\right), & x < -1 \\ 1 - \cos\left(\frac{8}{2x+3}\right), & x \geq -1 \end{cases},$$

где  $x$  — переменная величина, изменяющаяся в интервале  $[x_n; x_k]$  с шагом  $dx$  (значения  $x_n$ ,  $x_k$ ,  $dx$  вводятся с клавиатуры).

В отчете представить блок-схему, исходный код и результаты выполнения приложения



**Задание 3.** Используя оператор цикла **do ... while**, разработать приложение с заголовком «Оператор do ... while», с помощью которого вычисляется с заданной точностью  $\varepsilon$  корень уравнения:

$$f(x) = x^3 - 2x^2 - 4x - 7 = 0,$$

находящийся в промежутке [3; 4]. Значение точности  $\varepsilon$  вводится с клавиатуры.

На экран монитора вывести значение корня уравнения и количество итераций, позволяющих найти корень уравнения.

Так как функция  $f(x)$  на концах интервала [3; 4] имеет противоположные знаки, то корнем решения уравнения будет значение  $x$ , при котором значение функции  $f(x)$  изменится на противоположное. Поиск значения  $x$  осуществляется либо от начала интервала либо от конца интервала с шагом  $dx$  (это и есть заданная точность  $\varepsilon$ ).

### 3. Контрольные вопросы

1. Для чего предназначены операторы циклы?
2. Из чего состоят циклы?
3. Что такое итерация?
4. Какие циклы бывают и в чем разница между видами цикла?
5. Дайте характеристику циклу с предусловием.
6. Дайте характеристику циклу с постусловием.

### Литература

1. Голощапов А.Л. Microsoft Visual Studio 2010. – СПб.:БХВ-Петербург, 2011. – 544 с.: ил.
2. Петцольд Ч. Программирование для Microsoft Windows на C#. В 2-х томах. Том 1. Пер. с англ. - М.: «Русская Редакция», 2002.- 576 с.: ил.
3. Петцольд Ч. Программирование для Microsoft Windows на C#. В 2-х томах. Том 2. Пер. с англ. - М.: «Русская Редакция», 2002.- 624 с.: ил.
4. Троелсен Э. Язык программирования C# 2010 и платформа .NET 4.0. Пер. с англ. - М.: Издательский дом "Вильямс", 2011. — 1392 с.: ил.
5. Фленов М.Е. Библия C#. - СПб.: БХВ-Петербург, 2011. – 560с.: ил.
6. Шилдт Г. C# Учебный курс. – СПб.: Питер, Издательская группа BHV, 2003. – 512 с.: ил.