В представленных до сих пор примерах программ использовалась одна форма метода

Main(). Но у него имеется также целый ряд перегружаемых форм. Одни из них

могут служить для возврата значений, другие — для получения аргументов. В этом

разделе рассматриваются и те и другие формы.

**Возврат значений из метода Main()**

По завершении программы имеется возможность возвратить конкретное значение

из метода Main() вызывающему процессу (зачастую операционной системе). Для этой

цели служит следующая форма метода Main().

***static int Main()***

Обратите внимание на то, что в этой форме метода Main() объявляется возвращаемый

тип int вместо типа void. Условно нулевое возвращаемое значение обычно указывает

на нормальное завершение программы, а все остальные значения обозначают

тип возникшей ошибки.

**Передача аргументов методу Main()**

Многие программы принимают так называемые *аргументы командной строки,*

т.е. информацию, которая указывается в командной строке непосредственно после

имени программы при ее запуске на выполнение. В программах на C# такие аргументы

передаются затем методу Main(). Для получения аргументов служит одна из приведенных

ниже форм метода Main().

***static void Main(string[] args)***

***static int Main(string[] args)***

В первой форме метод Main() возвращает значение типа void, а во второй — целое

значение, как пояснялось выше. Но в обеих формах аргументы командной строки

сохраняются в виде символьных строк в массиве типа string, который передается методу

Main(). Длина этого массива (args) должна быть равна числу аргументов командной

строки, которое может быть и нулевым.

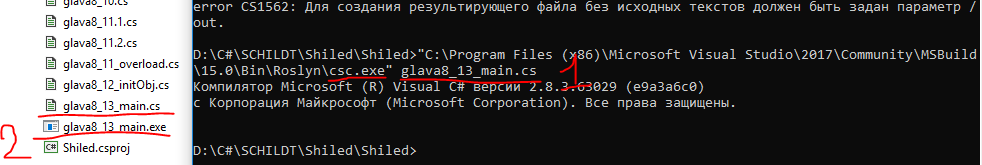
=====================================================================================

Для того, чтобы запускать свою программу из коммандной строки, надо сначала в папке с файлом кода который хочешь запустить, определить пусть к С# (***csc.exe***) вместе с названием нужного файла (в примере ***glava8\_13\_main.cs***). Путь по умолчанию (***кавычки обязательны***):

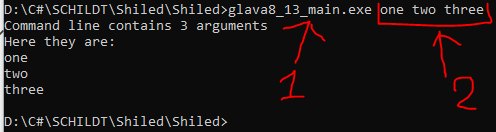
“C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\MSBuild\15.0\Bin\Roslyn\csc.exe”

Имя файла через пробел ***(1)***.

После чего, в папке создастся файл с идентичным названием нужного файла только с форматом ***.exe (2)***.



Далее, надо через командную строку запустить файл с расширением ***.exe (1)***, и через пробел вводить нужые слова ***(2)*** (зависит от реализации и кода, в данном случае проверка на вводимые слова).



=====================================================================================

В качестве примера ниже приведена программа, выводящая все аргументы командной

строки, вместе с которыми она вызывается.

class CLDemo

{

static void Main(string[] args)

{

Console.WriteLine("Command line contains "

+ args.Length + " arguments");

Console.WriteLine("Here they are: ");

for (int i = 0; i < args.Length; i++)

Console.WriteLine(args[i]);

}

}

Для того чтобы стало понятнее, каким образом используются аргументы командной

строки, рассмотрим еще один пример программы, в которой применяется простой

подстановочный шифр для шифровки или расшифровки сообщений. Шифруемое

или расшифровываемое сообщение указывается в командной строке. Применяемый

шифр действует довольно просто. Для шифровки слова значение каждой его буквы

инкрементируется на 1. Следовательно, Буква "А" становится буквой "Б" и т.д. А для

расшифровки слова значение каждой его буквы декрементируется на 1. Разумеется,

такой шифр не имеет никакой практической ценности, поскольку его нетрудно разгадать.

Тем не менее он может стать приятным развлечением для детей.

class Cipher

{

static int Main(string[] args)

{

//check for args

if (args.Length < 2)

{

Console.WriteLine("Do: "

+ "word 1: <crypt>/<decrypt> "

+ "[word2... wordN]");

return 1; //return bad execute programm

}

//If args are normal, first word should be

//<crypt> or word <uncrypt>

if (args[0] != "crypt" || args[0] != "decrypt")

{

Console.WriteLine("First argument has to be <crypt> or <decrypt>");

return 1; //return bad execute programm

}

//crypting / decrypting word

for (int n = 1; n < args.Length; n++)

{

for (int i = 0; i < args[n].Length; i++)

{

if (args[0] == "crypt")

Console.Write((char)(args[n][i] + 1));

else

Console.Write((char)(args[n][i] - 1));

}

Console.Write(" ");

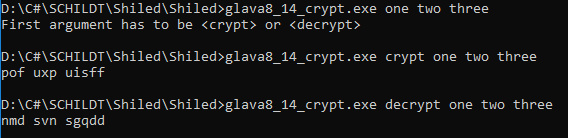
}

Console.WriteLine();

return 0;

}

}



Данная программа отличается двумя интересными свойствами. Во-первых, обратите

внимание на то, как в ней проверяется наличие аргументов командной строки

перед тем, как продолжить выполнение. Это очень важное свойство, которое можно

обобщить. Если в программе принимается во внимание наличие одного или более аргументов

командной строки, то в ней должна быть непременно организована проверка

факта передачи ей предполагаемых аргументов, иначе программа будет работать неправильно.

Кроме того, в программе должна быть организована проверка самих аргументов

перед тем, как продолжить выполнение. Так, в рассматриваемой здесь программе

проверяется наличие командного слова "зашифровать" или "расшифровать"

в качестве первого аргумента командной строки.

И во-вторых, обратите внимание на то, как программа возвращает код своего завершения.

Если предполагаемые аргументы командной строки отсутствуют или указаны

неправильно, программа возвращает код 1, указывающий на ее аварийное завершение.

В противном случае возвращается код 0, когда программа завершается нормально.