**Методичка C# Урок 4: Ввод данных**

Введение

В предыдущем уроке мы изучили унарные и бинарные операторы, а также научились изменять переменные с помощью составного присваивания и выводить целые числа на консоль.

Цели урока

В этом уроке вы:

1. Узнаете отличие статических программ от интерактивных.

2. Познакомитесь с понятиями «входные данные» и «строковый тип данных».

3. Научитесь писать код программы с использованием входных данных через практические задания.

Проблема

Все предыдущие программы выводили на экран данные, известные в момент написания кода. Например:

```csharp

Console.WriteLine("Привет! Как тебя зовут?");

string name = "Андрей";

Console.WriteLine("Привет, " + name);

```

Эта программа всегда будет выводить одно и то же:

```bash

Привет! Как тебя зовут?

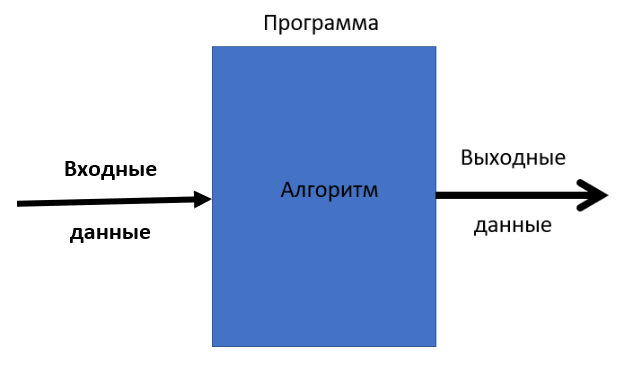
Привет, Андрей

```

Это статическая программа, которая не вызывает интереса.

Интерактивные программы, напротив, могут принимать данные от пользователя и выдавать разные результаты. Эти данные называются входными данными.

Схема работы программы



Данная схема является общей для всех современных программ:

1. **Программа считывает данные от пользователя**. В нашем случае программа будет считывать информацию с консоли. Это и есть входные данны(input).
2. Программа **обрабатывает**входные данные по заложенному алгоритму.
3. **Программа дает пользователю обратную связь.**В нашем случае программа будет выводить информацию на консоль. Данные, которые выводит программа, называются выходными данными (output).

В зависимости от входных данных один и тот же алгоритм будет выдавать разные выходные данные.

Решение

Для считывания данных в C# используется команда`Console.ReadLine()`.

Рассмотрим пример:

```csharp

Console.WriteLine("Привет! Как тебя зовут?");

string name = Console.ReadLine(); // пользователь вводит свое имя

Console.WriteLine("Привет, " + name); // приветствуем пользователя

```

Разбор программы:

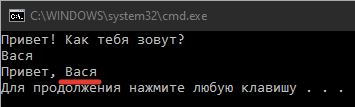
1. Программа выводит текст на экран.
2. Ждет ввода данных от пользователя через `Console.ReadLine()`. Введенная строка сохраняется в переменной `name`.

3. Выводит приветствие с именем пользователя.

**Запомните:** `Console.ReadLine()` получает данные в виде строки (`string`) и сохраняет их в переменной.

Подчеркну еще раз, что все данные, которые считываются с консоли, имеют **строковый** тип данных (**string**).

Давайте запустим программу и введем имя  Вася:



Если мы хотим, чтобы пользователь ввел n строк, то команд Console.ReadLine() в программе должно быть n штук.

Например, если пользователь вводит 3 строки, то для этого нужно написать:

```csharp

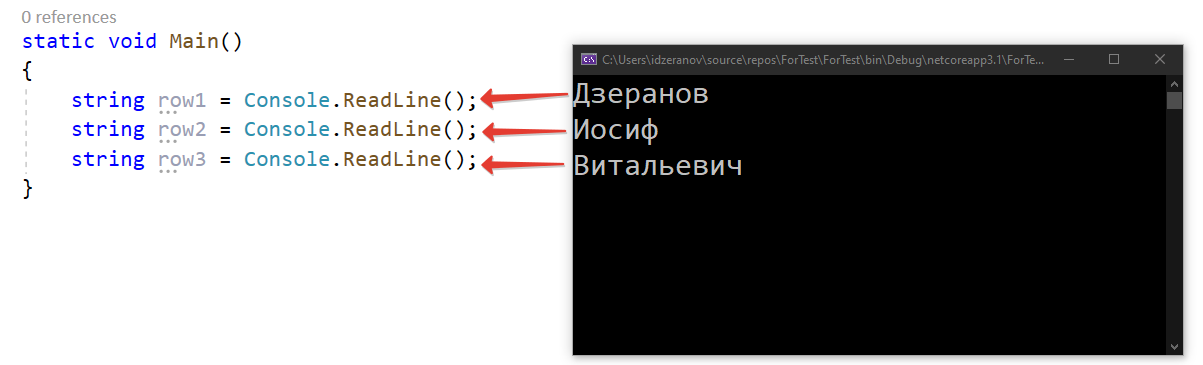
string row1 = Console.ReadLine();

string row2 = Console.ReadLine();

string row3 = Console.ReadLine();

```

Таким образом, первая строка, введенная пользователем запишется в переменную row1, вторая строка – в row2, третья – в row3:



Типичные ошибки

Обратим внимание, что просто Console.ReadLine() недостаточно, чтобы считать строку с консоли. Нужно ее сохранить в переменную, чтобы в дальнейшем ее использовать.

Например:

```csharp

Console.WriteLine("Привет! Как тебя зовут?");

Console.ReadLine(); // Это не сохранит данные

```

На второй строке мы просим пользователя ввести строку. Но после этого никуда ее не сохранили. Спрашивается, а зачем тогда просили? Это бессмысленная операция.

Надо обязательно сохранить в переменную строкового типа:

```csharp

Console.WriteLine("Привет! Как тебя зовут?");

string name = Console.ReadLine(); // Сохраняем данные

```

А дальше мы можем обращаться к этой строке в программе по названию переменной name.

Практика

1. Какой оператор используется для ввода данных с консоли?

1) Console.Write()

2) Console.WriteLine()

3) Console.ReadLine()

Ответ: 3) Console.ReadLine()

1. Какой тип данных имеют данные, которые считываются с консоли?

1) Зависит от того, что вводится (число ввели - имеет тип int, строку - string)

2) String

3) int

Ответ: 2) String

3. Программа «Приветствие»:

**Условие:**

На вход программе подается строка текста – имя человека.

Напишите программу, которая выводит на экран приветствие в виде слова «Привет» (без кавычек).

После должна стоять запятая и пробел, а затем введенное имя.

**Формат входных данных:**

На вход программе подаётся одна строка — имя человека.

**Формат выходных данных:**

Программа должна вывести текст в соответствии с условием задачи.

**Шаблон:**

```csharp

using System;

public class MainClass

{

public static void Main()

{

// put your c# code here

}

}

```

Ответ:

```csharp

using System;

public class MainClass

{

public static void Main()

{

string username = Console.ReadLine();

Console.WriteLine("Привет, " + username);

}

}

```

4. Программа «Любимая игра»:

**Условие:**

На вход программе подается строка текста – название игры. Напишите программу, которая повторяет ее на экране со словами « - лучшая игра!» (без кавычек).

**Формат входных данных:**

На вход программе подается название игры.

**Формат выходных данных:**

Программа должна вывести текст согласно условиям задачи.

**Шаблон:**

```csharp

using System;

public class MainClass

{

public static void Main()

{

// put your c# code here

}

}

```

Ответ:

```csharp

using System;

public class MainClass

{

public static void Main()

{

string gamename = Console.ReadLine();

Console.WriteLine(gamename + " - лучшая игра!");

}

}

```

5. Программа «Повторяй за мной»:

**Условие:**

Напишите программу, которая считывает три строки по очереди, а затем выводит их в той же последовательности, каждую на отдельной строчке.

**Формат входных данных:**

На вход программе подаются три строки, каждая на отдельной строке.

**Формат выходных данных:**

Программа должна вывести введенные строки в той же последовательности, каждую на отдельной строке.

**Шаблон:**

```csharp

using System;

public class MainClass

{

public static void Main()

{

// put your c# code here

}

}

```

Ответ:

```csharp

using System;

public class MainClass

{

public static void Main()

{

string str\_1 = Console.ReadLine();

string str\_2 = Console.ReadLine();

string str\_3 = Console.ReadLine();

Console.WriteLine(str\_1);

Console.WriteLine(str\_2);

Console.WriteLine(str\_3);

}

}

```

Пример других способов решения это задачи:

Я - Питонист, я люблю выводить всё с помощью одной функции/метода, написав код в 1-2 строки.

C#: не так быстро, друг..

\*\* Забежим немного вперед:

Тут я задумался, а почему я должен для вывода значения каждой переменной писать отдель Console.WriteLine() ? Почему нельзя просто взять и одной командой всё вывести?

Мой ответ таков: C# - зануда :-)

\*\* Есть другие способы (по-сложнее) :

1. **Использование форматирования строки**

Если вы хотите вывести все строки в одном вызове, вы можете использовать интерполяцию строк или метод string.Format():

```csharp

using System;

public class MainClass

{

public static void Main()

{

string str\_1 = Console.ReadLine();

string str\_2 = Console.ReadLine();

string str\_3 = Console.ReadLine();

Console.WriteLine($"{str\_1}\n{str\_2}\n{str\_3}");

}

}

```

1. **Использование массива**

Еще один способ — сохранить строки в массиве и вывести их с помощью цикла:

```csharp

using System;

public class MainClass

{

public static void Main()

{

string[] lines = new string[3];

for (int i = 0; i < 3; i++)

{

lines[i] = Console.ReadLine();

}

foreach (var line in lines)

{

Console.WriteLine(line);

}

}

}

```

6. Программа «Кастомный разделитель»

**Условие:**

Напишите программу, которая считывает строку-разделитель и три строки, а затем выводит значения этих строк через разделитель.

Например: Раз\*Два\*Три

**Формат входных данных:**

На вход программе подаётся строка-разделитель и три строки, каждая на отдельной строке.

**Формат выходных данных:**

Программа должна вывести на одной строке значения введённых выше строк, через разделитель.

**Шаблон:**

```csharp

using System;

public class MainClass

{

public static void Main()

{

// put your c# code here

}

}

```

Ответ:

```csharp

using System;

public class MainClass

{

public static void Main()

{

string separator = Console.ReadLine();

string str\_1 = Console.ReadLine();

string str\_2 = Console.ReadLine();

string str\_3 = Console.ReadLine();

Console.Write(str\_1);

Console.Write(separator);

Console.Write(str\_2);

Console.Write(separator);

Console.Write(str\_3);

}

}

```

Бонус:

Преобразование C# кода в исполняемый файл (.exe)

1. Создание проекта в **VS CODE**:

```bash

dotnet new console -o HelloNameAPP

```

2. Напишите код в модуле **Program.cs**:

```csharp

using System;

class Program

{

static void Main(string[] args)

{

Console.Write("Введите ваше имя: ");

string name = Console.ReadLine();

Console.WriteLine($"Привет, {name}!");

// Ожидаем ввода перед закрытием консоли

Console.WriteLine("Нажмите любую клавишу для выхода...");

Console.ReadKey(); // Ожидает нажатия клавиши

}

}

```

3. Перейдите в директорию **HelloNameAPP**:

```bash

cd HelloNameAPP

```

4. Соберите проект (преобразуйте в исполняемый файл):

```bash

dotnet publish -c Release -o ./output // Это создаст исполняемый файл в папке output.

```

**Как это работает:**

1. Запустите ваш .exe файл.

2. Введите имя и нажмите Enter.

3. Программа выведет приветствие.

4. После этого она попросит вас нажать любую клавишу, и только после этого консоль закроется.

Заключение

В этом уроке вы узнали, как вводить данные в C# и обрабатывать их, а также как создавать интерактивные программы. Практика с кодом поможет вам лучше понять, как работать с входными и выходными данными. Если у вас остались вопросы или вы хотите рассмотреть дополнительные примеры, не стесняйтесь обращаться! Удачи в программировании! 😊