

# ПЯВУ. Лекция 1.

Основы программирования.

А.М. Задорожный

# Содержание

1. Вводная часть
2. Анализ простейшей программы
3. Процесс построения программы  
(компиляция)
  - а) Виды ошибок
4. Переменные
5. Целочисленное деление
6. Представления программиста о компьютере
7. Двоичная система счисления

# Инструменты и средства обучения

- Учебный материал:
  - Электронные материалы от преподавателей в сети;
  - Шилдт Г. - C# 4.0 полное руководство.
- Visual Studio C#.

# Модель обучения

- Программирование – основа разработки современной электроники

- **Цель:** Быстро научиться программировать и понимать программы

Среда

Язык  
программирования

Алгоритмы

- “Импрессионизм” => Формализация

ЭВМ

Представление  
данных

# Среда программирования

- VisualStudio 2013 C# или более поздняя версия
  - Шилдт. Стр. 46 – “Применение интегрированной среды разработки VisualStudio”. *То что понятно.*

- Язык C#. Потомок C => C++ => C#.

C-подобные языки: Java, PHP, Perl, JavaScript, ...

- **Консольные** приложения. Console.

*Позволяет сосредоточиться на  
программировании*

# Первая программа на C#

```
using System;  
using System.Text;
```

```
namespace MyProgram
```

```
{  
    class Program
```

```
{  
    static void Main(string[] args)
```

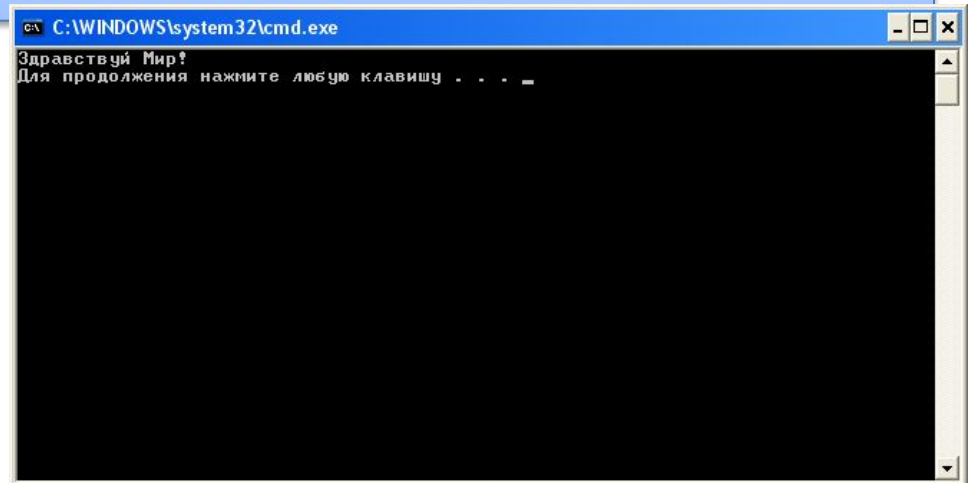
```
{  
    Console.WriteLine("Здравствуй Мир!");
```

```
}
```

```
}
```

```
}
```

Это Блок



The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The window contains the following text: "Здравствуй Мир?" followed by "Для продолжения нажмите любую клавишу . . .". This indicates that the program has successfully executed and printed the message "Здравствуй Мир!" to the console.

# Простейшая программа на C#

**Console**.WriteLine("Здравствуй Мир!"); // Объект консоль

Console.**WriteLine**("Здравствуй Мир!"); // Операция вывода

Console.WriteLine("Здравствуй Мир!"); // Параметр операции

Console.WriteLine("**Здравствуй Мир!**"); // Строка-литерал

Console.WriteLine("Здравствуй Мир!"); // ; завершает команду

Console.**Writeline**("Здравствуй Мир!"); // Регистр важен!

# Построение программы

- Текст программы (программа)
- Компилятор
- Выполняемая программа (программа)





# Синтаксические ошибки на С#

## Не там размещена команда!

```
using System;  
using System.Text;
```

```
namespace MyProgram
```

```
{
```

```
class Program
```

```
{
```

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

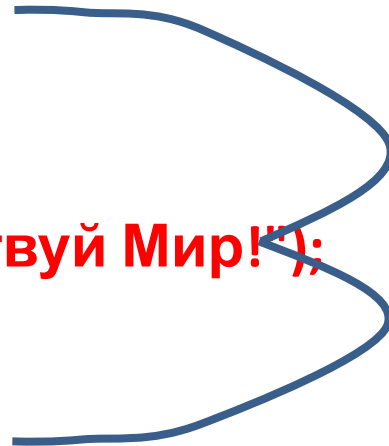
```
{
```

```
Console.WriteLine("Здравствуй Мир!");
```

```
}
```

```
}
```

```
}
```



# Синтаксические ошибки на C#

## Нет точки с запятой!

```
using System;  
using System.Text;  
  
namespace MyProgram  
{  
    class Program  
    {  
        static void Main(string[] args)  
        {  
            Console.WriteLine("Здравствуй Мир!");  
        }  
    }  
}
```

# Синтаксические ошибки на C#

## Не тот регистр!

```
using System;  
using System.Text;
```

```
namespace MyProgram  
{  
    class Program  
    {  
        static void Main(string[] args)  
        {  
            Console.WriteLine("Здравствуй Мир!");  
        }  
    }  
}
```

# Синтаксические ошибки

- Команды размещены не в том блоке
- Неполный блок ( { ) или лишняя скобка }
- Команда не завершается точкой с запятой
- Неправильно набрана команда
- .... много других.

**Компилятор не сможет построить программу по тексту, содержащему синтаксические ошибки!**

# Ошибки времени исполнения

Если компилятор построил программу, то она может содержать ошибки! *Пример:*  
*(1/0)*

Программа будет *прерываться, зависать*  
и т.п.

**Это Ошибки времени исполнения**

# Контрольные вопросы

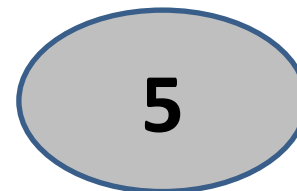
1. Что такое 'Блок' в языке C#? Каковы правила для объявления блоков в C#?
2. Где мы будем писать свою часть текста программы на первых занятиях?
3. Как в языке C# заканчивается команда?
4. Что означает утверждение: "C# чувствителен к регистру"?
5. Объясните термины: "Среда разработки", "Текст программы", "Компилятор", "Ошибка компиляции", "Ошибка времени исполнения программы".
6. Как в тексте программы на C# объявить неизменяемые текстовые данные?
7. Какая команда позволяет вывести текст на консоль?

# Развиваем программу. Переменные.

```
static void Main(string[] args)
{
    Console.WriteLine("Здравствуй Мир!");
}
```



```
static void Main(string[] args)
{
    int x = 5;
    Console.WriteLine(x);
}
```



x

*Переменная*  
*Я*

# О переменных

```
int x = 5;
```

**Выдели место (в программе оно будет называться 'x') в памяти под целое число (int) и положи туда 5.**

```
Console.WriteLine(x);
```

**Выведи на консоль значение из памяти, которую назвали 'x'**



# Следствия понимания переменной

1. Нельзя объявить 2 переменных с одинаковым именем 5  
X
2. Нельзя использовать имя переменной до ее объявления
3. Нельзя использовать переменную, которой не присвоено значение с иной целью, чем присваивание значения.

# Форматированный вывод

```
int x = 5;
```

```
Console.WriteLine(x);
```

```
>5
```

```
Console.WriteLine("x = {0}", x);
```

```
>x = 5
```

Два параметра: текстовый и целочисленный.  
Разделяются запятыми...

# Развиваем программу дальше

```
int x = 2, y = 3;
```

```
Console.WriteLine("{0} + {1} = {2}", x, y, x + y);
```

`>2 + 3 = 5` параметр - выражение

Можно применять и другие операции над целыми числами: **+**, **-**, **\***, **/**.

# Целочисленное деление

$$3 / 2 = 1$$

$$5 / 3 = 1$$

Результат деления целых чисел всегда целый!

$$3 \% 2 = 1$$

$$5 \% 3 = 2$$

Остаток от деления (%) то же целый!

$$(X / N) * N + X \% N \Rightarrow X$$

# Контрольные вопросы

1. Что такое “переменная”?
2. Как объявить переменную целого типа?
3. Как задать переменной начальное значение?
4. Как объявить несколько целочисленных переменных?
5. Является ли строка программы: `int a = 1;` инструкцией для выполнения?
  
6. Что изменится, если в строке `Console.WriteLine(“{0} + {1} = {2}”, x, y, x + y);` заменить **первый** значок ‘+’ на ‘-’?
  
7. Что изменится, если в строке `Console.WriteLine(“{0} + {1} = {2}”, x, y, x + y);` заменить **второй** значок ‘+’ на ‘-’?
  
8. Что изменится, если последний параметр взять в кавычки?
  
9. Что изменится, если последний параметр удалить: `Console.WriteLine(“{0} + {1} = {2}”, x, y);`?

# Операция присваивания

$x = 5;$

В память названную  $x$   
положить  $5$

$x = y;$

В память названную  $x$   
положить значение из  
памяти названной  $y$

$x = y + 5;$

В память названную  $x$   
положить сумму значения из  
памяти названной  $y$  и  $5$

# Присваивание - не равенство

$x = y;$        $y = x;$

$x = x + 1;$

Присваивание – команда!

Слева всегда ТОЛЬКО имя переменной!

Справа – выражение.

Выражение вычисляется и результат помещается в память соответствующую имени переменной.

$x <= \dots$       так бы было правильнее, но ' $<=$ ' уже занято!

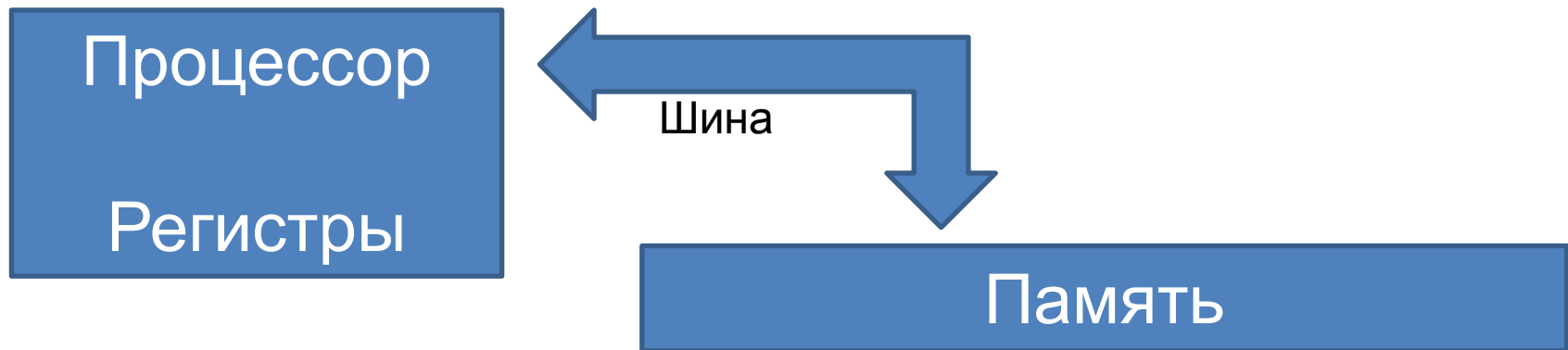
# Короткий итог

1. Научились создавать простые программы; (*Main*)
2. Познакомились с некоторыми синтаксическими правилами C#; (*регистр, точка с запятой*)
3. Можем выводить сообщения на консоль; (*Console.WriteLine*)
4. Познакомились с понятием переменной; (*int x = 5;*)
5. Можем объявлять целочисленные переменные, присваивать им значения и выполнять над ними операции; (*=, +, -, \*, /, %*)
6. Узнали об особенностях деления целых чисел в C#;
7. Познакомились с операцией присваивания, которая позволяет изменять значение переменной; (*x = y; y = x; x = x + 1 – команды*)
8. Познакомились с форматированным выводом; ("*{0} + {1} = {2}*")
9. Понимаем что такое текст программы и как из него получается программа; (*компилятор*)
10. Коснулись видов ошибок, возникающих при разработке программ;



# Модель компьютера

Процессор, память и шина.



Процессор выполняет команды над данными

Память хранит команды и данные

Шина связывает процессор с памятью

# Программа – команды и данные

- Сложить значение по адресу A1 со значением по адресу A2, а результат поместить по адресу A3.

Команда – сложить, A1, A2 и A3 – параметры команды.

Когда команда выполнена, в процессор подгружается следующая команда.

Если не указано иное, команды выполняются последовательно!

# Организация памяти

- Бит – 0 или 1
- Байт –  $2^8 = 256$  различных значений

**Адрес в памяти – фактически номер байта**

Память может хранить только целые числа!

Все данные команды в памяти компьютера представлены кодами (целыми числами)

199 69 248 1 0 0 0

пример кода команды, которая помещает 1 в одну из

# Двоичная система счисления

- Только 2 цифры – 0 и 1
- $a_n a_{n-1} \dots a_1 a_0$ , где  $a_i$  – одна из этих цифр
- $X = 2^n * a_n + 2^{n-1} * a_{n-1} + \dots + 2 * a_1 + a_0$
- $111 = 4+2+1 = 7$
- $1010 = 8+2 = 10$
- $10101 = 16+4+1 = 21$

# Операции в двоичной системе

- $1+1=10$ ,  $10+1=11\dots$

Особенности операций в компьютере.

- $11111111 + 1 = 0$
- $\Rightarrow 11111111 = -1$
- В байте могут храниться целые числа **от -128 до 127**

# Контрольные вопросы

1. Какую роль в компьютере выполняет Процессор? Память? Шина?
2. Что представляет собой готовая компьютерная программа в процессе выполнения (из чего состоит)?
3. Что такое Бит? Байт?
4. Как представлена информация в компьютере? Что такое Код?
5. Что означает термин 'позиционная система счисления'?
6. Программисты часто используют числа: 2,4,8, 16, **32, 64**, 128, **256**, 512, **1024**. Чем эти числа замечательны?
7. Где встречаются каждое из чисел **32, 64, 256** и **1024**?
8. Чем отличается компьютерное представление целых чисел от их двоичного представления в математике?
  - a. Ограничены ли целые числа в математике?
  - b. Как представляются отрицательные целые числа в компьютере?