Язык C# и платформа .NET

лекция1

История языка

Название «Си шарп» (англ: sharp — диез) происходит из обозначения, которые исползуются для записи музыкальных композиций: где латинской букве *C* соответствует нота »до», а знак ♯ означает повышение соответствующего звука на полутон. Но, на практике из-за того, что знак диеза ♯ не представлен на стандартной клавиатуре компьютера, при записи имени языка программирования используют ” знак решётки” (#).

Первая версия языка вышла в январе-феврале 2002 года.

Язык C# является языком семейства C, а точнее гибридом языков C, Java, Visual Basic.

C# функционирует на платформе .NET.

Версии

На протяжении разработки языка C# было выпущено несколько его версий:

Версия	Дата	.NET Framework	Visual Studio
C# 1.0	Январь 2002	.NET Framework 1.0	Visual Studio .NET (2002)
C# 1.2	Апрель 2003	.NET Framework 1.1	Visual Studio .NET 2003
C# 2.0	Ноябрь 2005	.NET Framework 2.0	Visual Studio 2005
C# 3.0	Ноябрь 2007	.NET Framework 3.5	Visual Studio 2008
C# 4.0	Апрель 2010	.NET Framework 4	Visual Studio 2010
C# 5.0	Август 2012	.NET Framework 4.5	Visual Studio 2012
C# 6.0	Июль 2015	.NET Framework 4.6	Visual Studio 2015
C# 7.0	Март 2017	.NET Framework 4.6.2	Visual Studio 2017
C# 8.0	Сентябрь 2019	.NET Framework 4.8	Visual Studio 16.3.0

Платформа .Net

Платформа представляет собой программную среду для создания приложений на разных языках, для разных операционных систем (не только из семейства Windows). Все языки, поддерживаемые платформой .Net имеют общий испольняюший маханизм.

Язык C# был создан специально для работы с фреймворком .NET (для этой платформы).

Фреймворки – это программные продукты, которые упрощают создание и поддержку технически сложных или нагруженных проектов. Фреймворк, как правило, содержит только базовые программные модули, а все специфичные для проекта компоненты реализуются разработчиком на их основе.

Это некий набор библиотек, который облегчает разработку любых продуктов: web-сайтов и web-сервисов, мобильных или десктопных приложений.

фреймворк – это рабочая среда, которая помогает разработчику быстро и качественно создавать программный продукт, не отвлекаясь на мелочи. Собрал основной каркас программы по шаблону, присоединил необходимые функции из библиотек и можно

Виды фреймворков

framework привязан к конкретной технологии и/или языку программирования:

- Zend Framework (The Laminas Project). Это программный продукт используют многие профессиональные PHP-программисты. Он делает жизнь web-разработчика легче, прежде всего, потому, что содержит множество полезных библиотек. К ним относятся возможность интеграции проекта с YouTube и другими сервисами, упрощение работы с базами данных, пользователями, и пр.
- Bootstrap. Популярный framework, помогающий быстро и качественно верстать макеты сайтов. Включает в себя шаблоны для создания слоев, кнопок, форм, блоков навигации и других элементов web-страниц.
- Yii. Объектно-ориентированный framework для создания масштабных web-приложений: интернет-магазинов. Главными достоинствами Yii является высокая производительность и безопасность.
- Corona SDK (Solar2D). Инструментарий для разработки игр и приложений для Android. Его отличительной особенностью является то, что он работает на собственном языке программирования, который носит название Lua.

Платформа .NET

Платформа .NET состоит из двух основных компонентов:

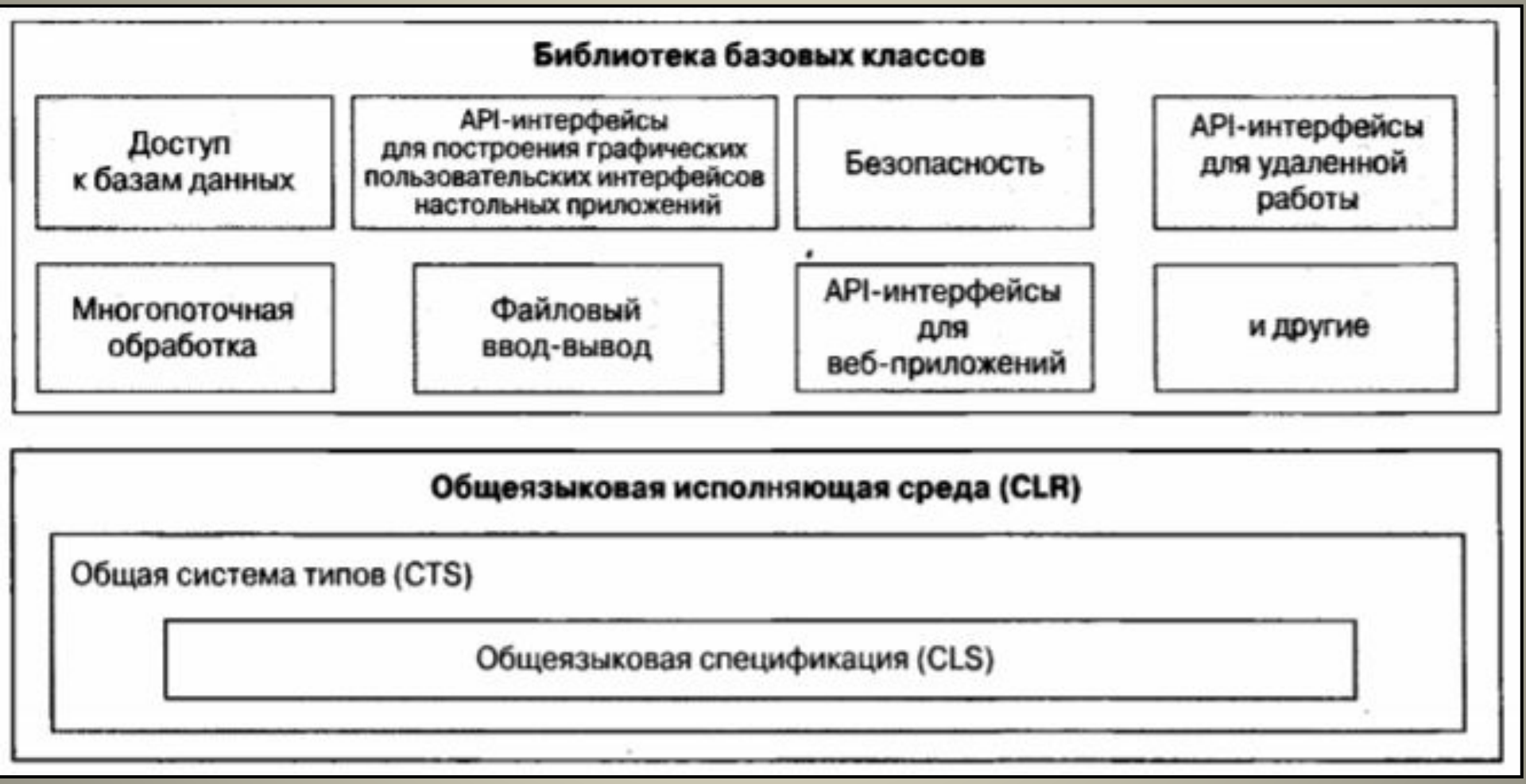
исполняющей среды общего языка (Common Language Runtime, CLR),

библиотеки классов .NET Framework (готовые классы и код, которые разработчики могут использовать в своих приложениях).

Сборка, компиляция, выполнение

- общий промежуточный язык (Common Intermediate Language — **CIL**) и
- оперативная компиляция (Just-In-Time Compilation — **JIT**).
- **общеязыковая исполняющая среда (Common Language Runtime — CLR),**
- общая система типов (Common Type System — **CTS**)
- общеязыковая спецификация (Common Language Specification — **CLS**).

Отношения между CLR, CTS, CLS



Встроенные типы данных CTS

Тип данных CTS	Ключевое слово VB	Ключевое слово C#	Ключевое слово C++/CLI
System.Byte	Byte	byte	unsigned char
System.SByte	SByte	sbyte	signed char
System.Int16	Short	short	short
System.Int32	Integer	int	int или long
System.Int64	Long	long	__int64
System.UInt16	UShort	ushort	unsigned short
System.UInt32	UInteger	uint	unsigned int или unsigned long
System.UInt64	ULong	ulong	unsigned __int64
System.Single	Single	float	float
System.Double	Double	double	double
System.Object	Object	object	object^
System.Char	Char	char	wchar_t
System.String	String	string	String^
System.Decimal	Decimal	decimal	Decimal
System.Boolean	Boolean	bool	bool

Управляемый код

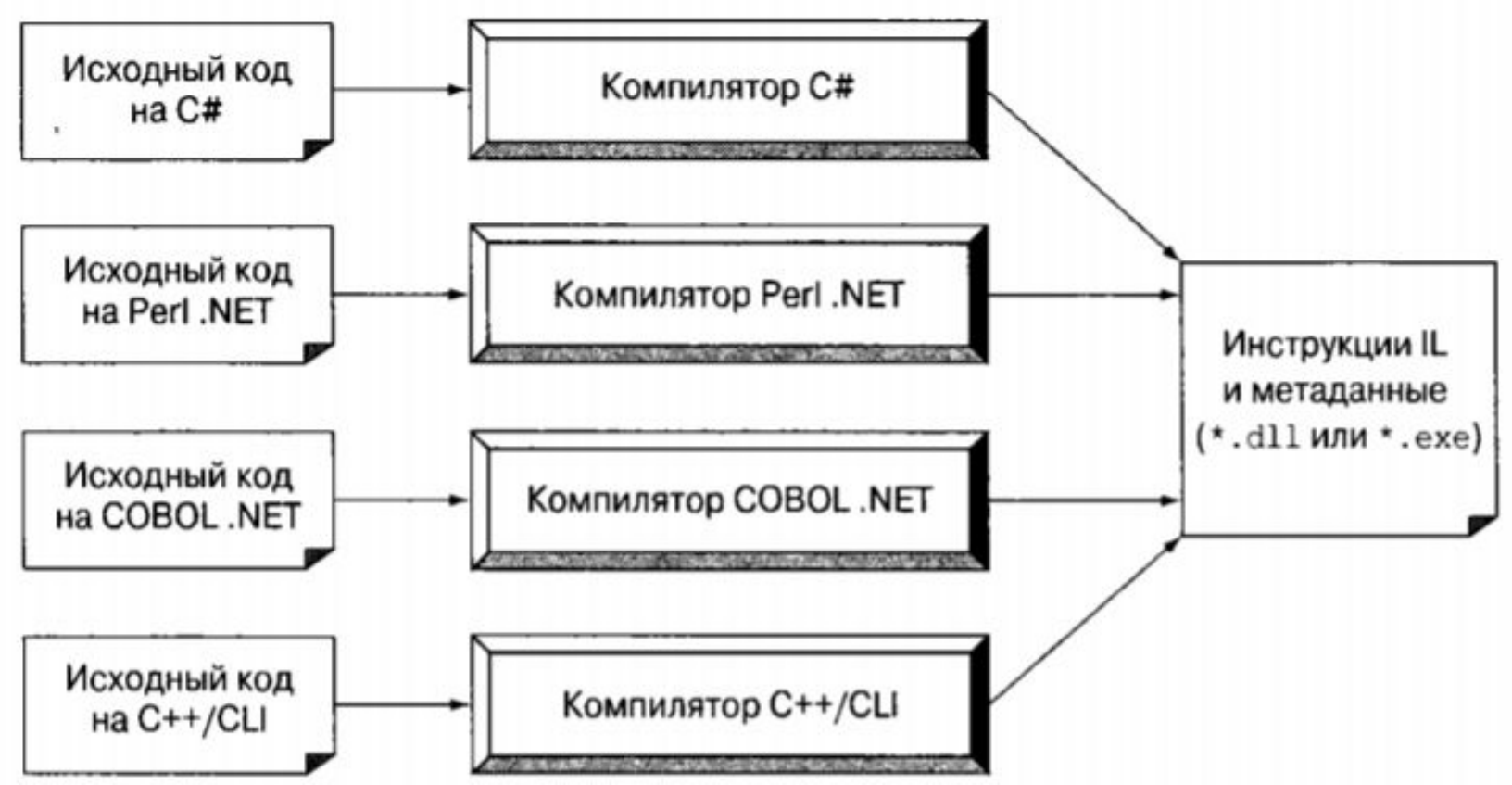
Согласно технологии .Net при компиляции программы на Язык C# создается исполняемый код, который может выполняться только в рамках исполняющей среды .NET (CLR).

Поэтому, для обозначения исполняемого кода, ориентированного на исполняющую среду CLR, используется термин: **управляемый код**.

Двоичный модуль, который содержит этот управляемый код, называется **сборкой**.

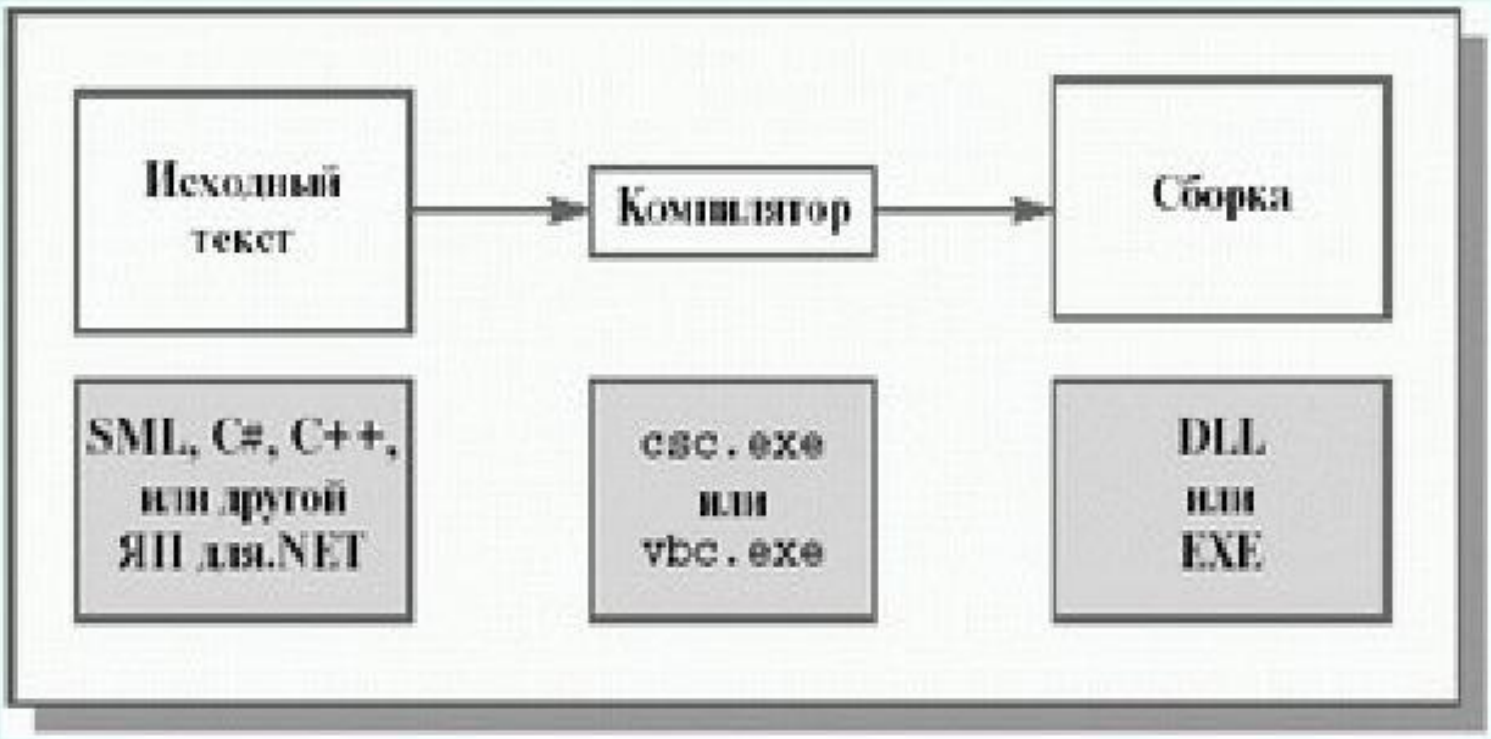
В противоположность этому, код, который не может обслуживаться непосредственно исполняющей средой .NET, называется **неуправляемым кодом (обычный бинарный код)**.

Common Intermediate Language – CIL



Сборка и компиляция

Common Intermediate Language – CIL

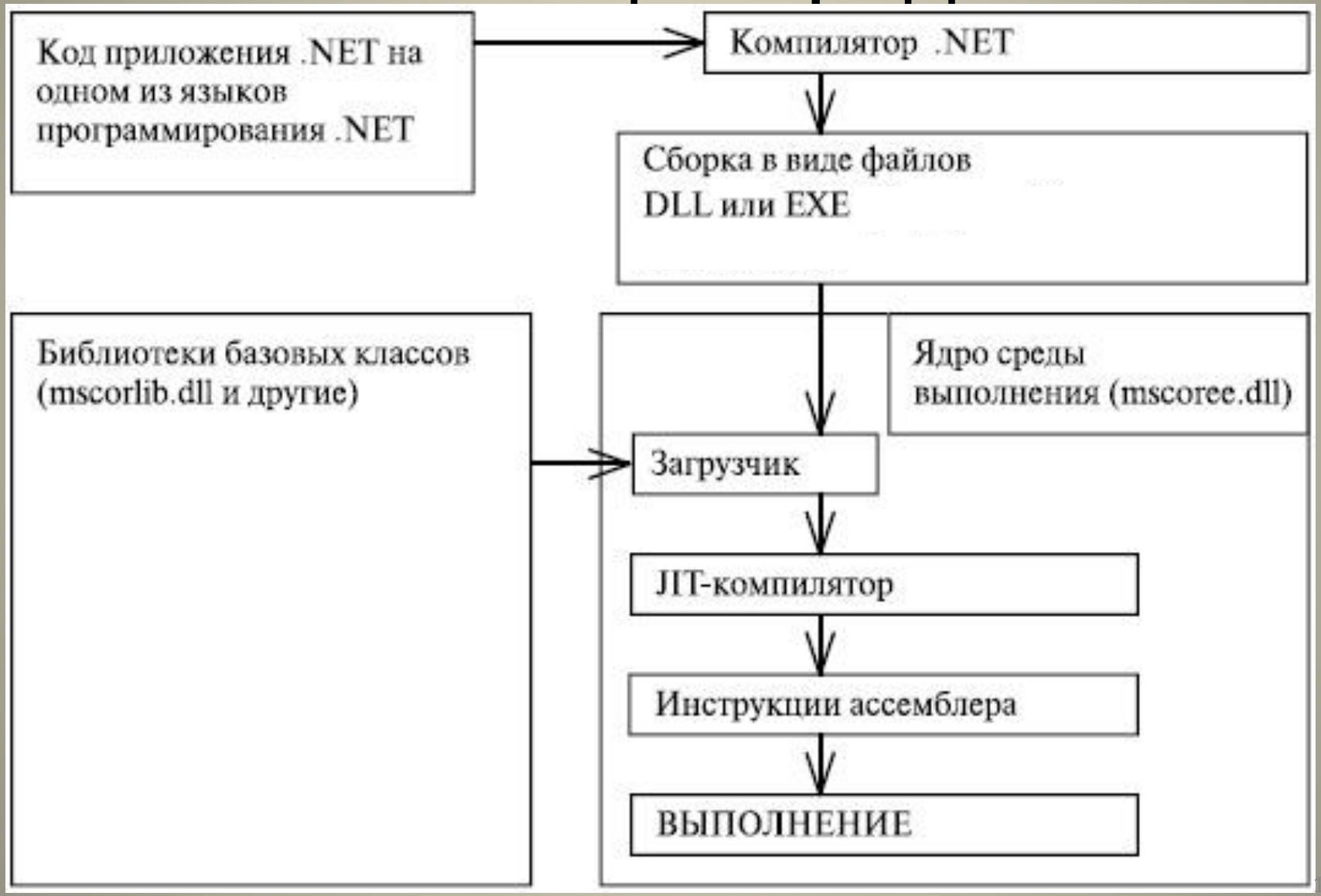


Сборка и компиляция

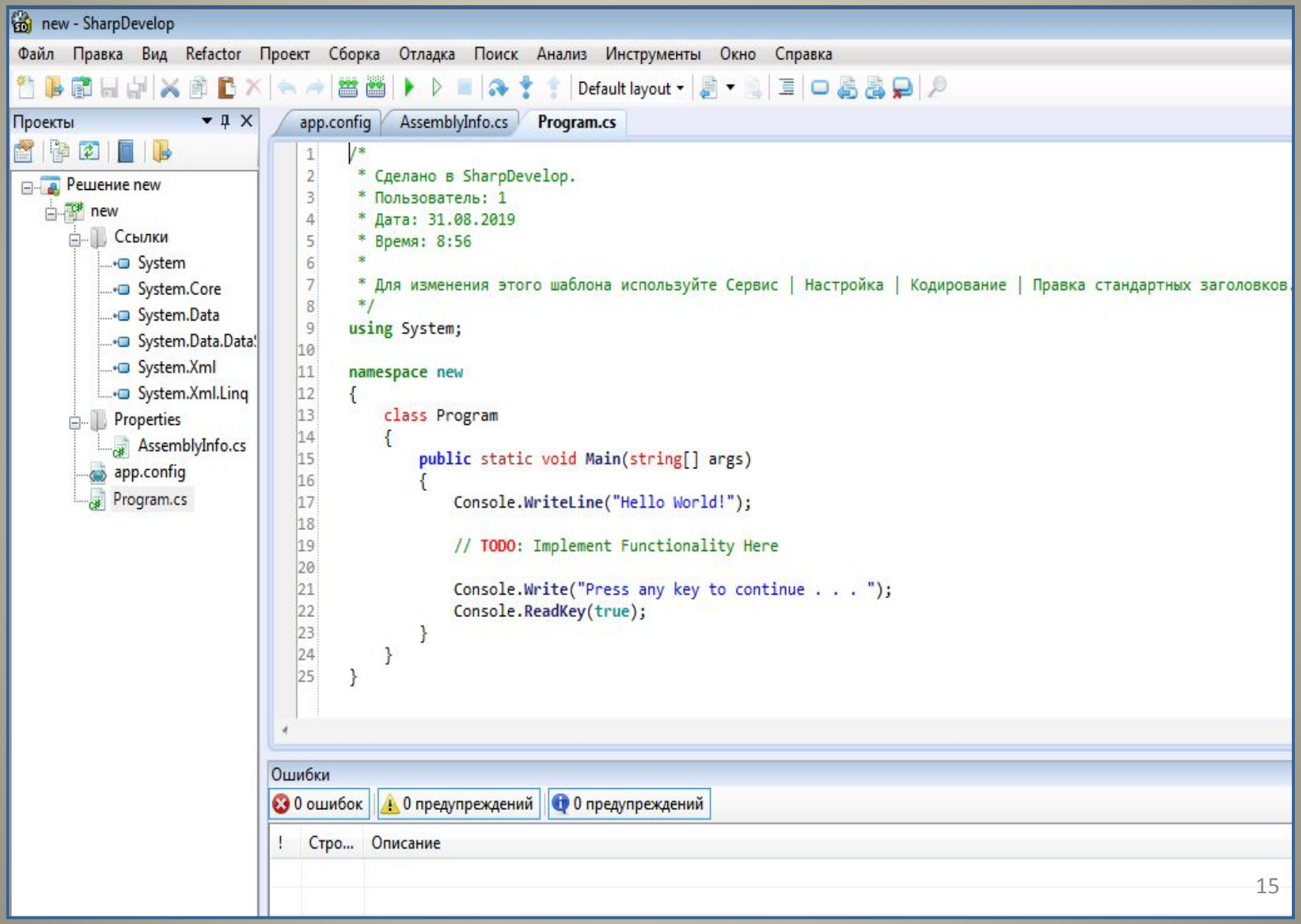
Исполняющая среда – CLR

- mscoree.dll - реализация CLR
- mscorlib.dll – реализация библиотеки базовых классов

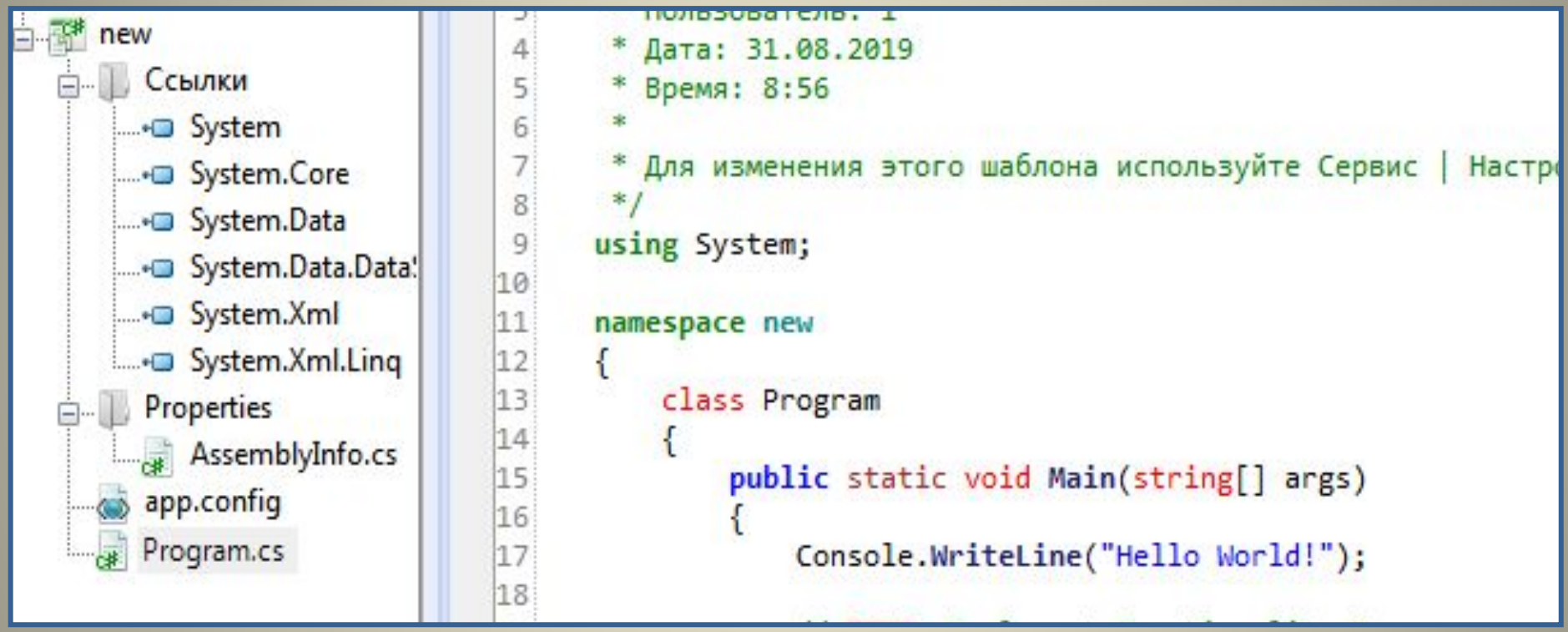
Исполняющая среда – CLR



Первая программа



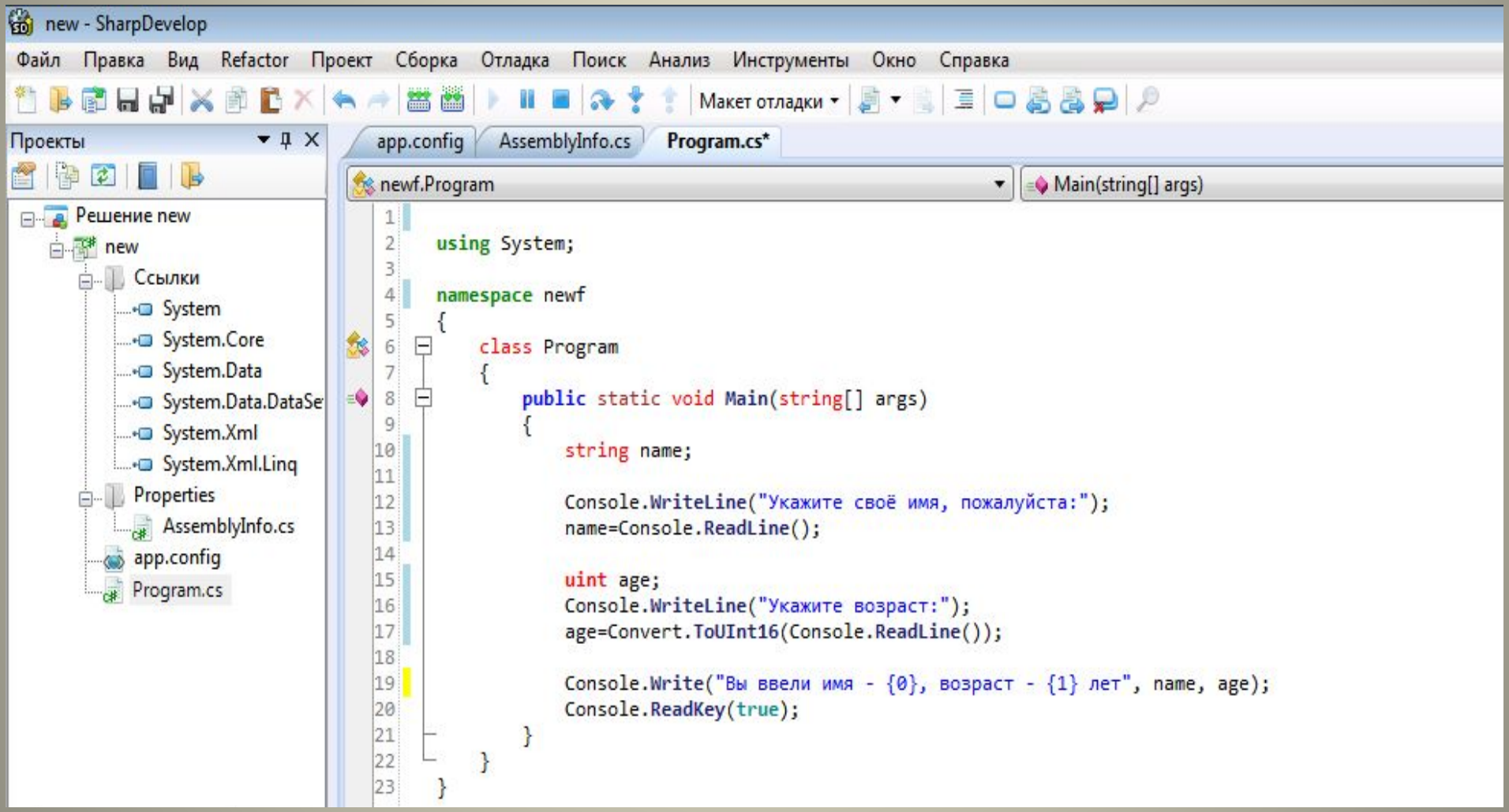
Первая программа



Первая программа

```
using System;

namespace new
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.Write("Введите свое имя: ");
            string name = Console.ReadLine();       // вводим имя
            Console.WriteLine("Привет {0}", name);    // выводим имя на консоль
            Console.ReadKey();
        }
    }
}
```

Первая программа

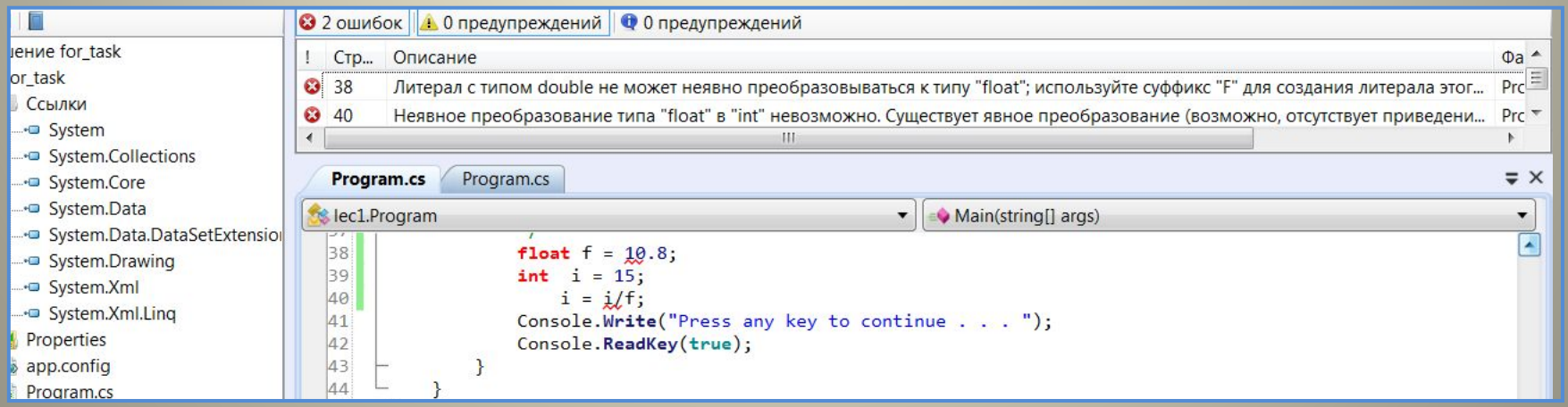


Программа

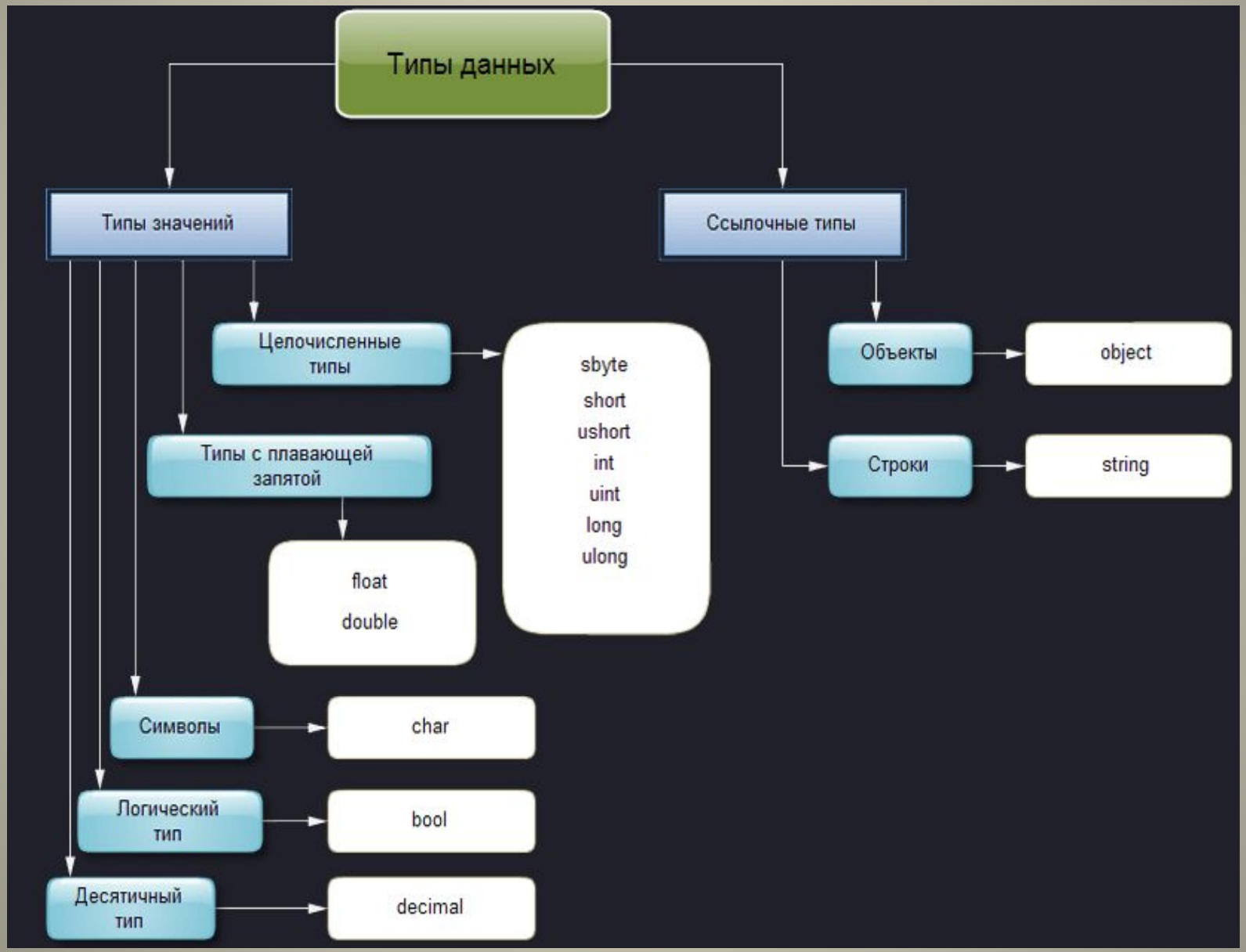
```
{
    class Program
    {
        public static void Main(string[] args)
        {
            int age;
            double length;
            string name;
            try{
                Console.Write("\nInput: name, age, length (using backspace)\n:");
                string temp = Console.ReadLine();
                string [] datas = temp.Split(new char[]{' '}, StringSplitOptions.RemoveEmptyEntries);
                name = datas[0].Substring(0,1).ToUpper()+datas[0].Substring(1).ToLower();
                age = int.Parse(datas[1]);
                length = double.Parse(datas[2]);

                Console.WriteLine("\n New staff: {0}, age:{1}, length:{2:F2}\n", name, age, length);
            }

            catch(Exception ex){
                Console.Write(ex.Message);
            }


            Console.Write("Press any key to continue . . . ");
            Console.ReadKey(true);
        }
    }
```

Типы

- C# строго типизированный язык.
- Все операции подвергаются строгому контролю со стороны компилятора на соответствие типов, причем недопустимые операции не компилируются.
- Все переменные, выражения и значения должны принадлежать к определенному типу.
- Тип значения определяет те операции, которые разрешается выполнять над ним.



Типы значений и ссылочные типы



Целочисленные типы

Тип	Тип CTS	Разрядность в битах	Диапазон
byte	System.Byte	8	0:255
sbyte	System.SByte	8	-128:127
short	System.Int16	16	-32768 : 32767
ushort	System.UInt16	16	0 : 65535
int	System.Int32	32	-2147483648 : 2147483647
uint	System.UInt32	32	0 : 4294967295
long	System.Int64	64	-9223372036854775808 : 9223372036854775807
ulong	System.UInt64	64	0 : 18446744073709551615

Базовые типы

char: хранит 16-битный символ Unicode. Наименьшее возможное значение для символа Unicode равно 0, наибольшее 65535.
Float (System.Single): хранит 32-битное число с плавающей запятой, со знаком. Наименьшее возможное значение примерно 1.5 * 10 в -45 степени, наибольшее примерно 3.4 * 10 в 38 степени.
double: хранит 64-битное число с плавающей запятой, со знаком. Наименьшее возможное значение примерно 5 * 10
в 324 степени, наибольшее примерно 1.7 * 10 в 308 степени.
decimal: хранит 128-битное число с плавающей запятой, со знаком. Переменные типа decimal хорошо подходят для финансовых вычислений

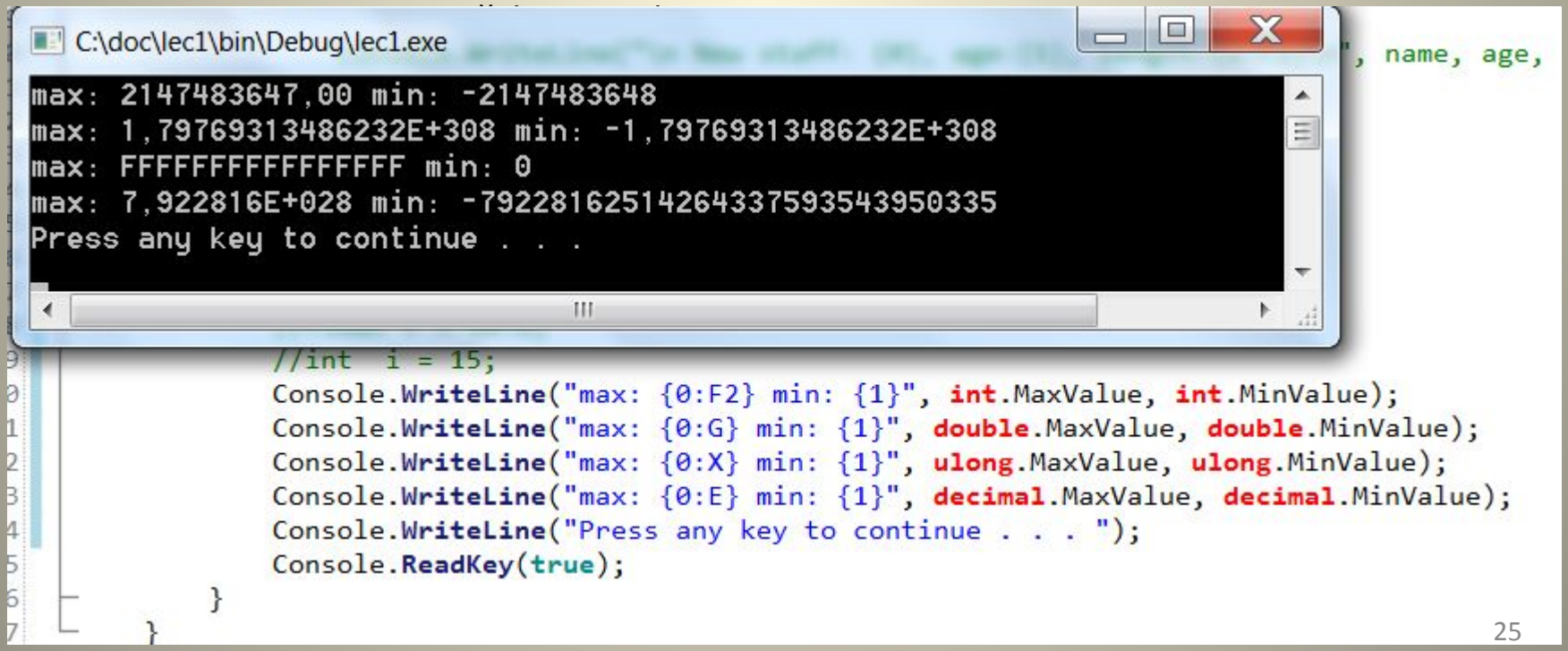
Ссылочные типы

string: представляет строку символов Unicode. Позволяет просто манипулировать строками и назначать их. Строки не являются мутируемыми (immutable), т. е. будучи созданными, они не могут быть модифицированы.

object: представляет тип общего назначения (general purpose type). В C# все предопределенные типы и заданные пользователем типы наследуются от типа object или класса System.Object.

Форматированный вывод

D или d – десятичное число (Decimal);
E или e – научный формат (Scientific, exponential)
F или f – формат с фиксированным значением после запятой (Fixed-point)
G или g – общие (General)
N или n – Number (Number)
X или x – шестнадцатеричный формат (Hexadecimal)



Создание классов

C# является полноценным объектно-ориентированным языком.
Программу на C# можно представить в виде взаимосвязанных взаимодействующих между собой объектов.

Создание классов

Класс — это группа сущностей (объектов), обладающих сходными свойствами, а именно: **данными и поведением**. В дальнейшем отдельного представителя некоторого класса будем называть **объектом** класса или просто объектом.

Можно сказать, что объектам класса присущи их общие **свойства и поведение**. Однако каждый объект всегда имеет свое, **уникальное состояние**, определяемое текущими значениями его свойств. Функциональное назначение (поведение) класса определяется **возможными действиями** над его представителями.

Создание класса

Вся функциональность класса представлена его членами:

Полями (полями называются переменные класса),
Свойствами,
Методами,
Конструкторами,
Событиями и др. определениями.

Создание класса