Информационные технологии и программирование

Лекция 1. Основы языка C#

Содержание лекции:

- Компоненты программирования
- История создания языка C#
- Общие понятия
- Процесс преобразования C# кода в машинный код
- Среда разработки
- Создание первой программы
- Система типов

Преподаватель курса: Клюкин Даниил Анатольевич, ст. преподаватель каф. ПМиИТ

Преподаватель курса:

Клюкин Даниил Анатольевич,

инженер-программист 1-й категории

https://github.com/DaniilKlyukin

Длительность курса:

3 семестра.

Форма контроля:

1-й семестр – экзамен,

2-й семестр – зачет с оценкой,

3-й семестр – зачет с оценкой.

Для допуска к экзамену/зачету необходимо:

1. Не иметь прогулов.
2. Сдать лабораторные работы.

На экзамене/зачете оценка зависит от полноты ответа на билет.

Прогулы необходимо отработать: предоставить конспект лекции, решить дополнительные практические задания.

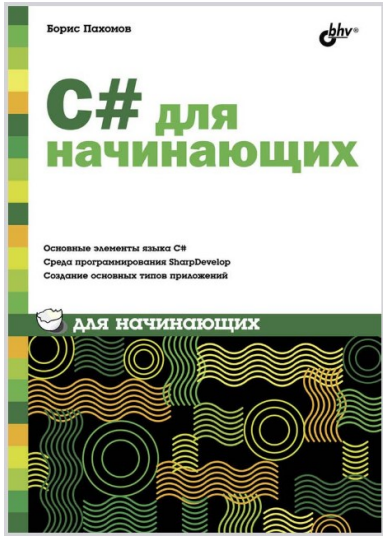
Литература

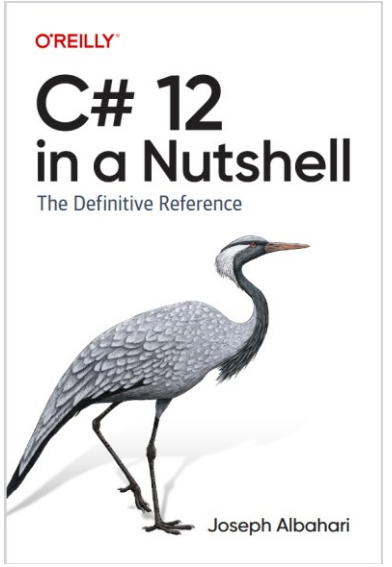
Основные книги:

1. **Пахомов Б.И.** C# для начинающих.
2. **Васильев А.Н.** Программирование на C#.
3. **Джеффри Рихтер.** CLR via C#.
4. **Джозеф Албахари.** C# 12 in a Nutshell.

Дополнительные книги:

1. **Стив Макконнел.** Совершенный код.
2. **Роберт Мартин.** Чистый код.
3. **Дональд Кнут.** Искусство программирования.
4. **Эндрю Таненбаум.** Архитектура компьютера.





Курсы по программированию на C#

1. **Полное руководство по языку программирования C# 13 и платформе .NET 9:** https://metanit.com/sharp/tutorial/
2. **Обучающие курсы** https://ulearn.me/

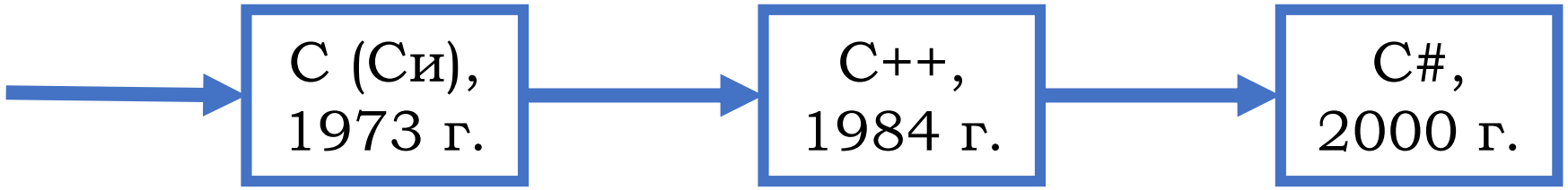
Интересные статьи:

1. **Ликбез по типизации:** https://habr.com/ru/articles/161205/
2. **Особенности строк в C#:** Особенности строк в .NET / Хабр

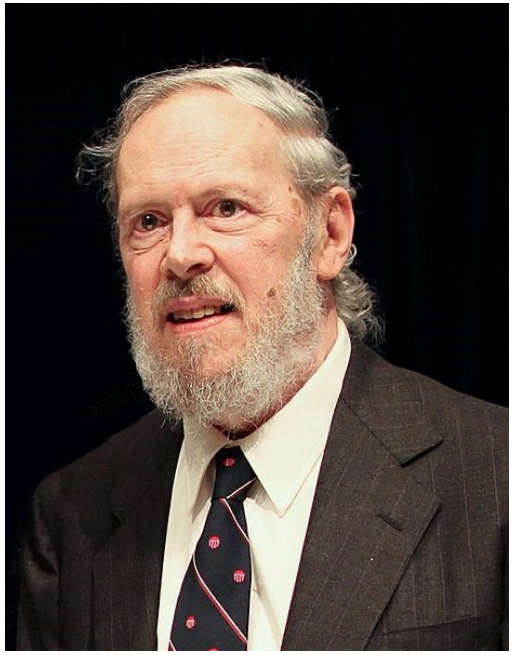
Компоненты программирования

Что нужно знать, чтобы уметь программировать?

История создания языка C#



C был разработан в начале 1970-х годов **Деннисом Ритчи** в **Bell Labs**. Он был создан как ЯП общего назначения, предназначенный для написания операционных систем и другого системного программного обеспечения. С отличался своим структурным подходом, портативностью и эффективностью.



Деннис Ритчи

В начале 1980-х годов **Бьярн Страуструп** в **Bell Labs** расширил возможности языка **C**, создав **C++**.

C++ быстро стал популярным благодаря своей универсальности и эффективности. Он используется для создания программного обеспечения разного рода: от игр до ОС. Этот язык также широко применяется в обработке данных и научных расчетах.

Бьярне Страуструп

Андерс Хейлсберг

В конце **1990-х** годов компания **Microsoft** начала разрабатывать новый язык программирования, который должен был стать более современным и безопасным, чем C++. Этот язык получил название **C#**.

C# был разработан командой под руководством **Андерса Хейлсберга**. Он был вдохновлен C++ и другими языками, такими как Java и Modula-3.

C# был официально выпущен в 2002 году и быстро завоевал популярность среди разработчиков. Он стал одним из ведущих языков программирования для разработки приложений **Windows**, **веб-приложений** и **мобильных приложений**.

Ключевые особенности C#

C# унаследовал многие функции от C++, но также добавил ряд новых функций, в том числе:

- **Управляемая среда:** C# выполняется в управляемой среде, которая обеспечивает автоматическое управление памятью и защиту от сбоев.
- **Безопасность типов:** C# использует строгую систему типов, которая помогает предотвратить ошибки во время выполнения.
- **Совместимость с .NET:** C# является частью платформы .NET, которая предоставляет богатый набор библиотек и служб для разработки приложений.

Общие понятия

Программирование — это наука, изучающая теорию и методы разработки, производства и эксплуатации программного обеспечения ЭВМ.

Суть программирования заключается в том, чтобы составить **алгоритм** и перевести его на **язык программирования**.

Алгоритм — это **описание последовательности операций**, направленных на решение поставленной задачи.

Язык программирования — это набор правил, с помощью которых программист записывает исходную программу. Из полученного текста специализированные программы (трансляторы, компоновщики и др.) формируют код, предназначенный для процессора.

Программа — комбинация компьютерных **инструкций** и **данных**, позволяющая аппаратному обеспечению вычислительной системы выполнять вычисления или функции управления.

Компилятор — это программа-транслятор, которая преобразует исходный код, написанный на языке программирования высокого уровня, в эквивалентный **машинный код** (**исполняемый файл**) или промежуточное представление (например, **байт-код**) перед выполнением программы. **Пример: GCC (C/C++).**

Интерпретатор — это программа, которая выполняет исходный код построчно, транслируя и исполняя каждую инструкцию непосредственно во время работы программы (**runtime**), без предварительного создания отдельного исполняемого файла. **(Python).**

Машинный код — это низкоуровневая последовательность двоичных или шестнадцатеричных инструкций, которые могут быть **непосредственно выполнены процессором** компьютера без дополнительной трансляции.

Пример машинного кода: B8 2A 00 00 00

Байт-код — это промежуточное представление кода, предназначенное для выполнения **виртуальной машиной (ВМ)** или интерпретатором, а не процессором напрямую.

По степени соответствия конструкций языка машинному (процессорному) коду языки программирования делятся на

- **Низкоуровневые** (машинно-ориентированные)
- **Высокоуровневые.**
  - **Процедурные** – функции обрабатывающие данные.

  C, Паскаль, Basic, Алгол и др.
  - **Объектно-ориентированные** – взаимодействующие объекты.

  C#, C++, Python, Java др.

Процесс преобразования C# кода в машинный код

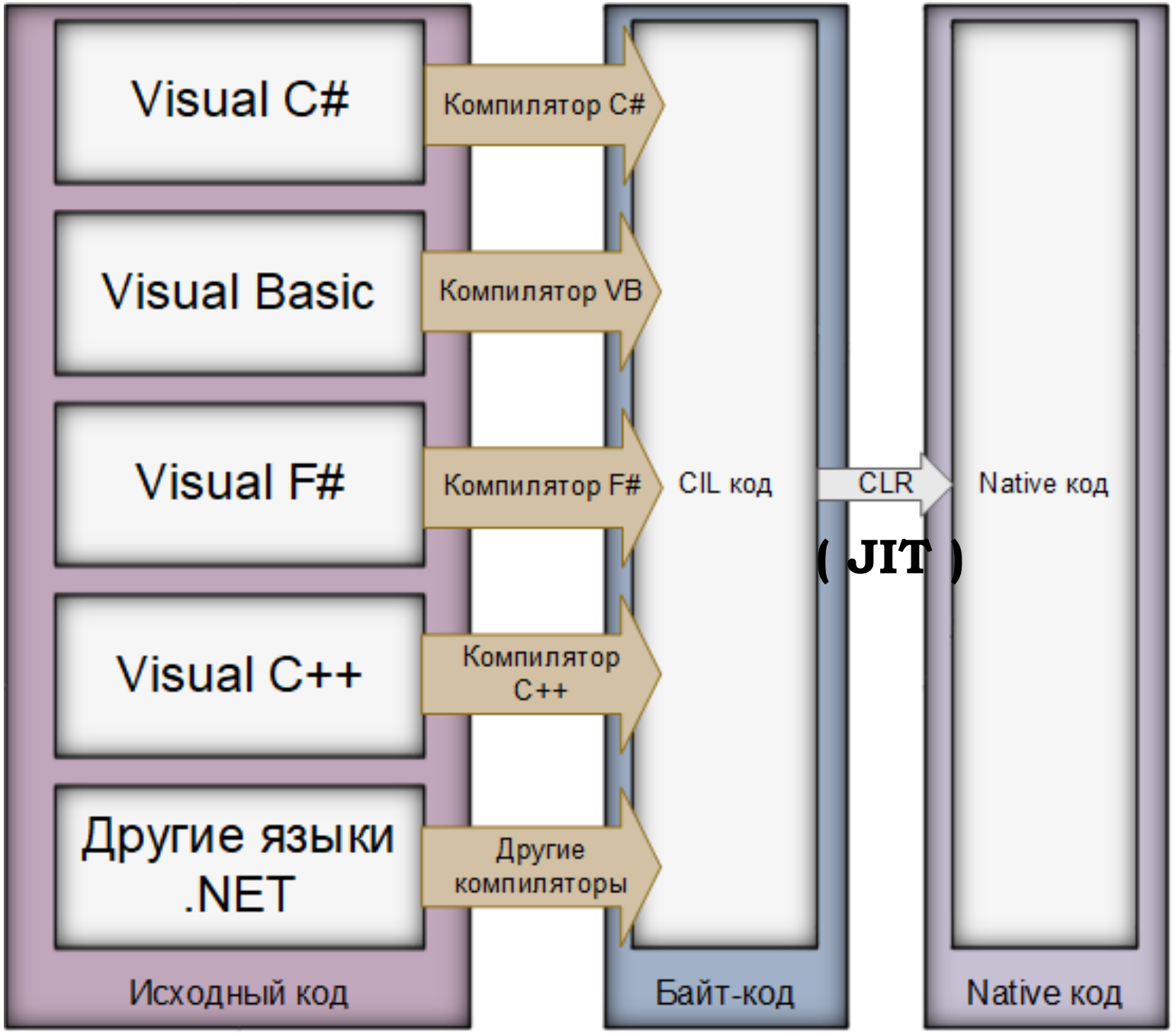
Компиляция – трансляция (перевод) программы, составленной на исходном языке высокого уровня, в эквивалентную программу на низкоуровневом языке, близком машинному коду (иногда на язык ассемблера), выполняемая компилятором.

CIL (MSIL) – Common Intermediate Language (Microsoft Intermediate Language). Промежуточный язык, разработанный компанией Microsoft для платформы .NET Framework. Является независимым от ЦП набором инструкций, которые можно эффективно преобразовать в машинный код.

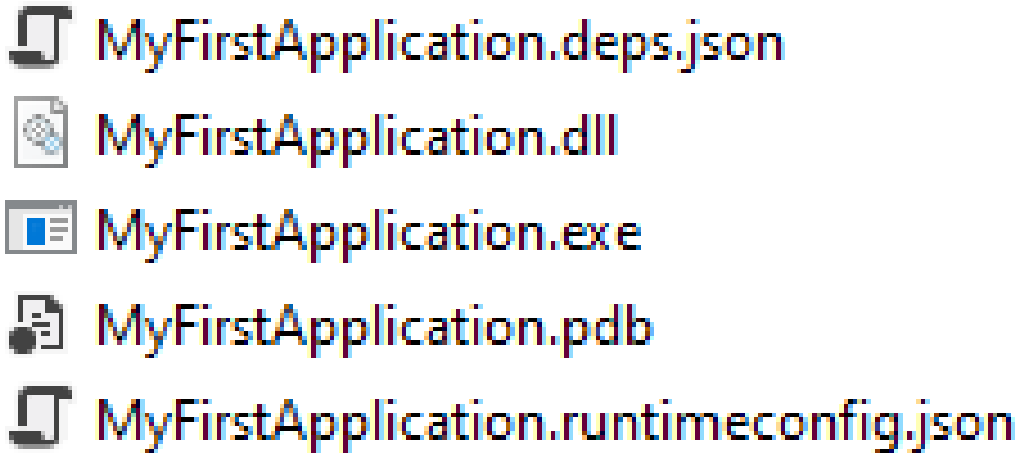
CLR – Common Language Runtime исполняющая среда для байт-кода CIL, в который компилируются программы, написанные на .NET-совместимых языках программирования.

JIT – Just In Time компиляция.

Преобразование байт кода в машинные инструкции.




```
// Здесь пишут программный код
Console.WriteLine("Hello, World!");
```



Среда разработки

Интегри́рованная среда́ разрабо́тки (*Integrated development environment* — ***IDE***), также **единая среда разработки** – комплекс программных средств, используемый программистами для разработки программного обеспечения (ПО).

Microsoft Visual Studio 2022 – IDE от компании Майкрософт.

Visual Studio включает один или несколько компонентов из следующих: Visual Basic .NET, Visual C++, Visual C#, JavaScript, Python (включён начиная с Visual Studio 2019), TypeScript, XAML.

JetBrains Rider (Win/macOS/Linux, платный).

MonoDevelop (Win/macOS/Linux, бесплатный).

Основные компоненты IDE:

1.Текстовый редактор с подсветкой синтаксиса, автодополнением кода (*IntelliSense*) и навигацией по проекту.

2.Транслятор (компилятор и/или интерпретатор) для преобразования исходного кода в исполняемую программу.

3.Средства автоматизации сборки (*build tools*) для управления зависимостями и компиляцией проекта.

4.Отладчик (*debugger*) для пошагового выполнения кода, анализа переменных и диагностики ошибок.

IntelliSense — технология автодополнения кода от компании Microsoft, наиболее известная в Microsoft Visual Studio. Дописывает элементы кода при вводе начальных букв (появляется при вводе).

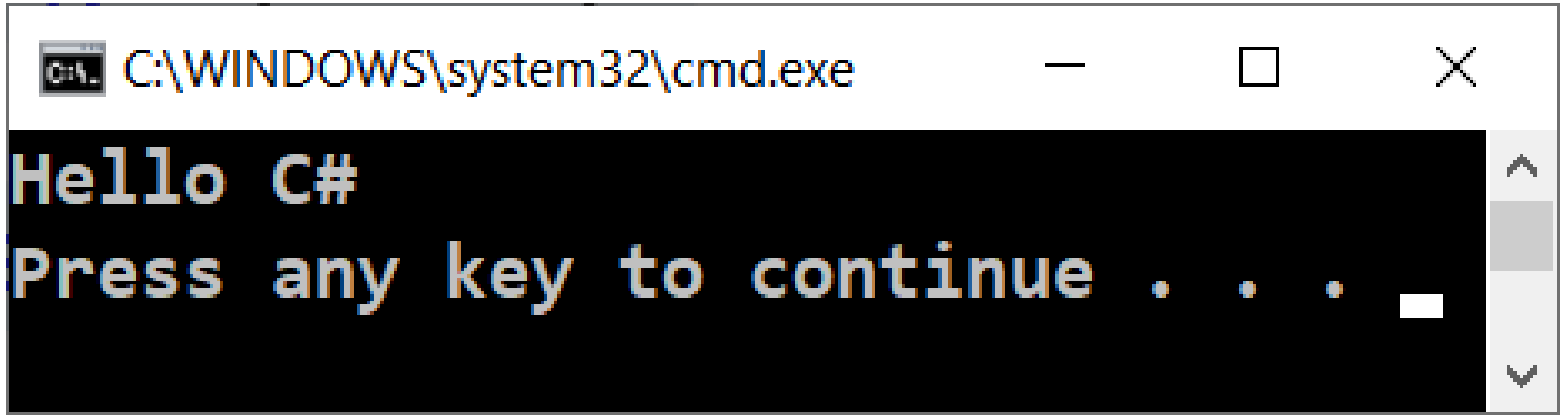
Принятие автодополнения происходит по нажатию на **Tab** или **Enter**.

Быстрые команды

- Закомментировать код: Ctrl + K, Ctrl + C.
- Форматирование кода: Ctrl + K, Ctrl + D.
- Переименование элемента кода с учётом всех зависимостей: Ctrl + R, Ctrl + R.

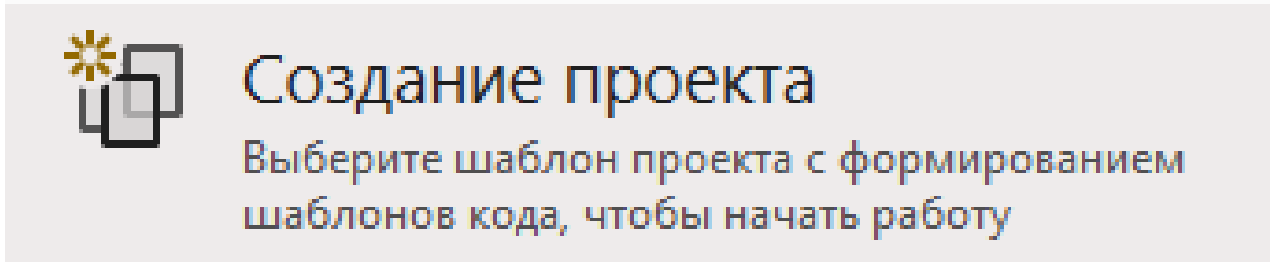
Создание первой программы

Консольное приложение — это простейший тип программы, которая взаимодействует с пользователем через **текстовый интерфейс командной строки** (терминал). В отличие от графических приложений (GUI), оно не имеет окон, кнопок или визуальных элементов, а ввод/вывод данных происходит через текст.

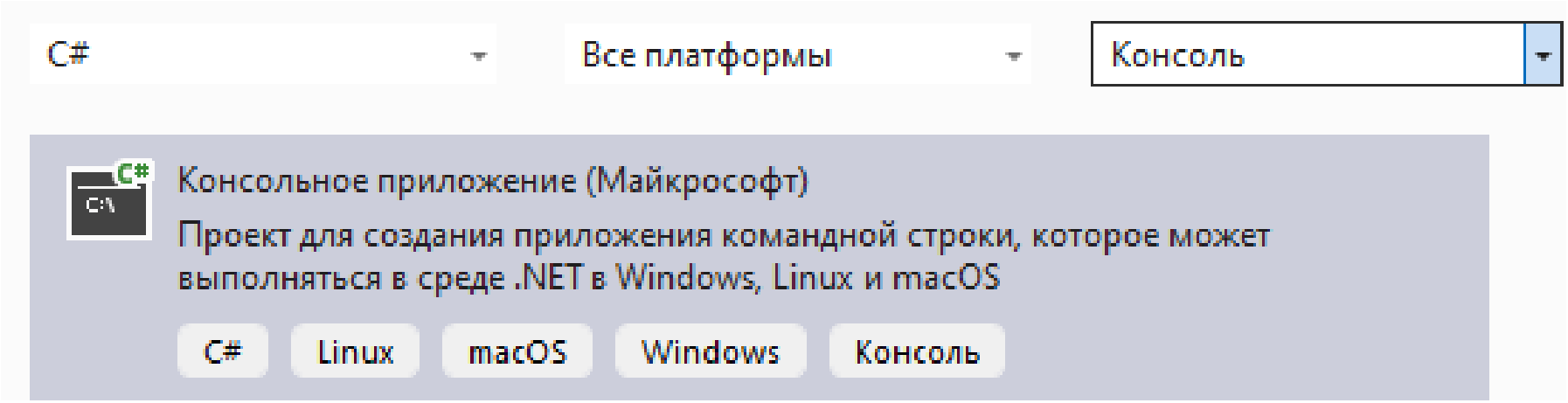


1. Необходимо скачать и установить программу Visual Studio 2022 Community (https://visualstudio.microsoft.com/ru/vs/)
2. Запустить Visual Studio 2022

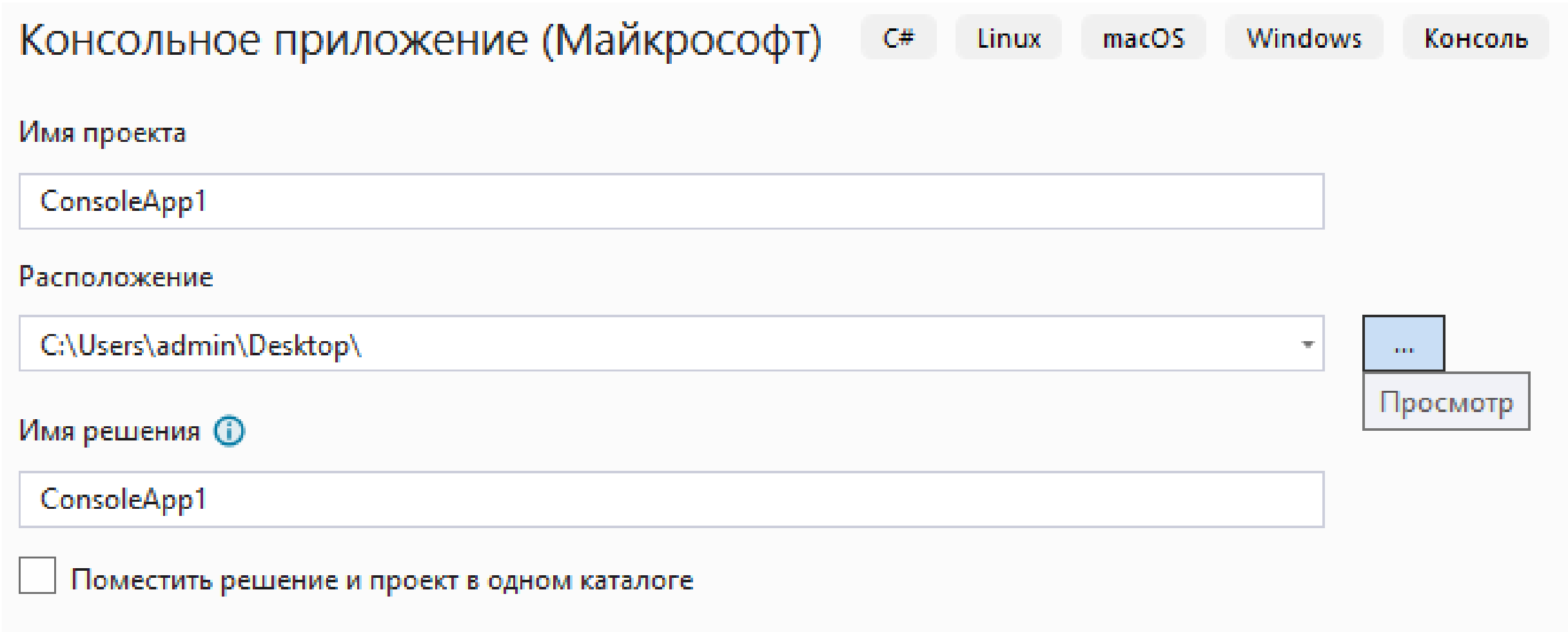
3. Выбрать «Создание проекта»



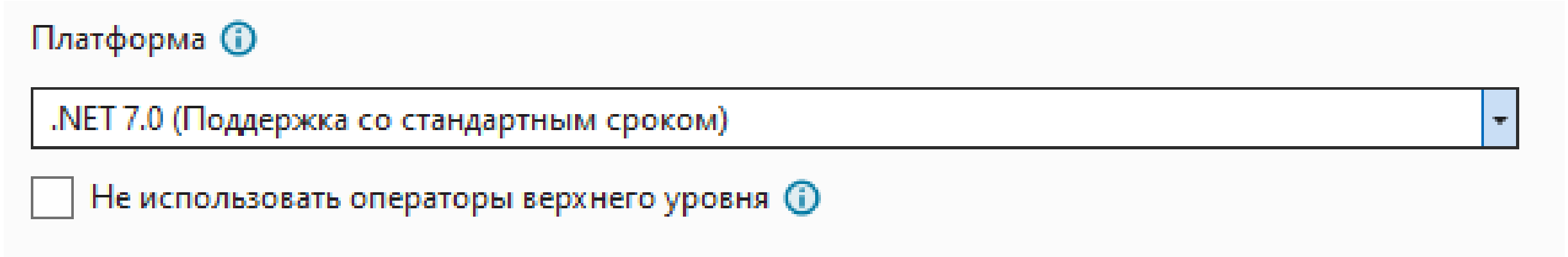
4. Выбрать шаблон проекта «Консольное приложение (Майкрософт)»

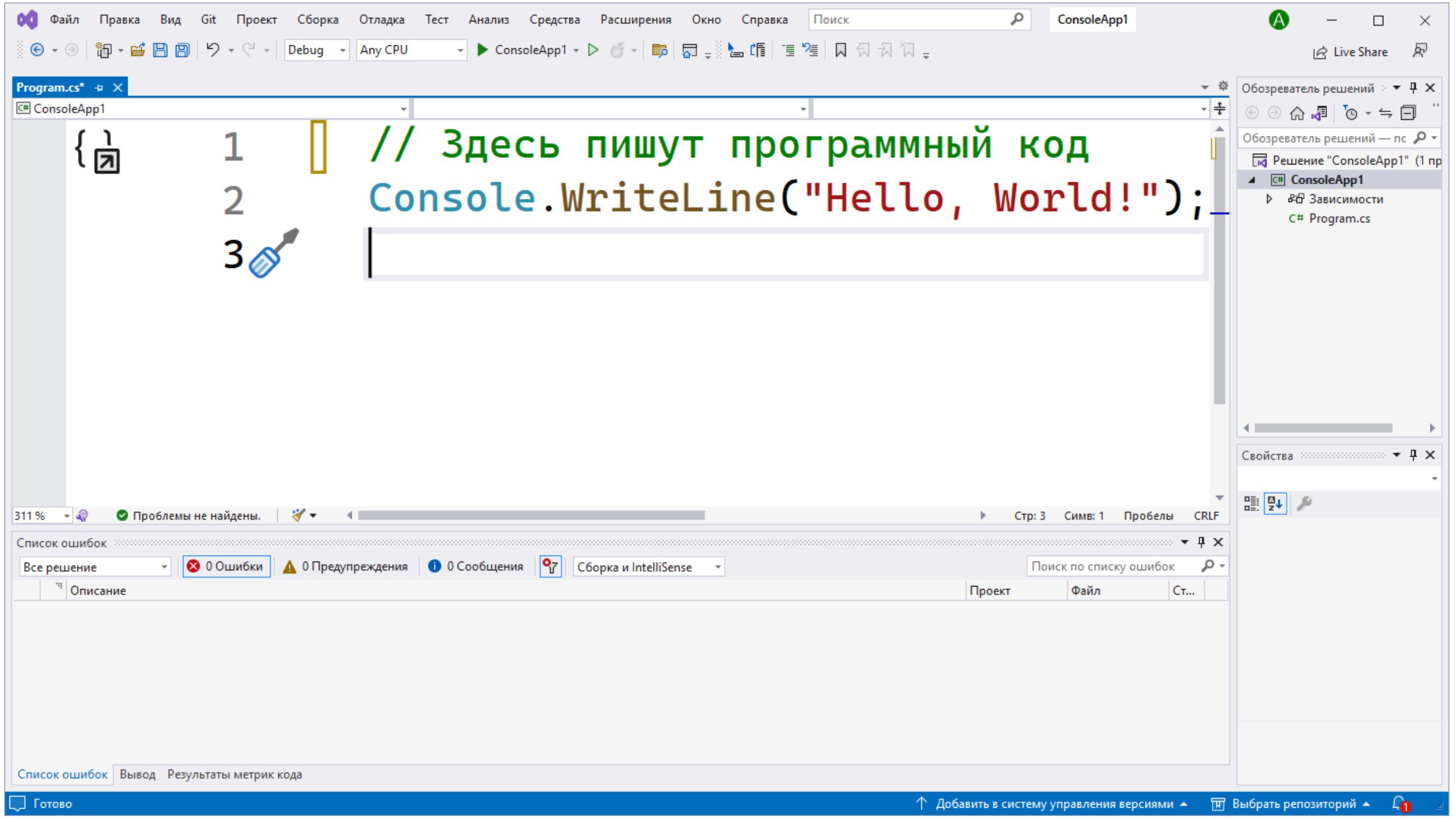


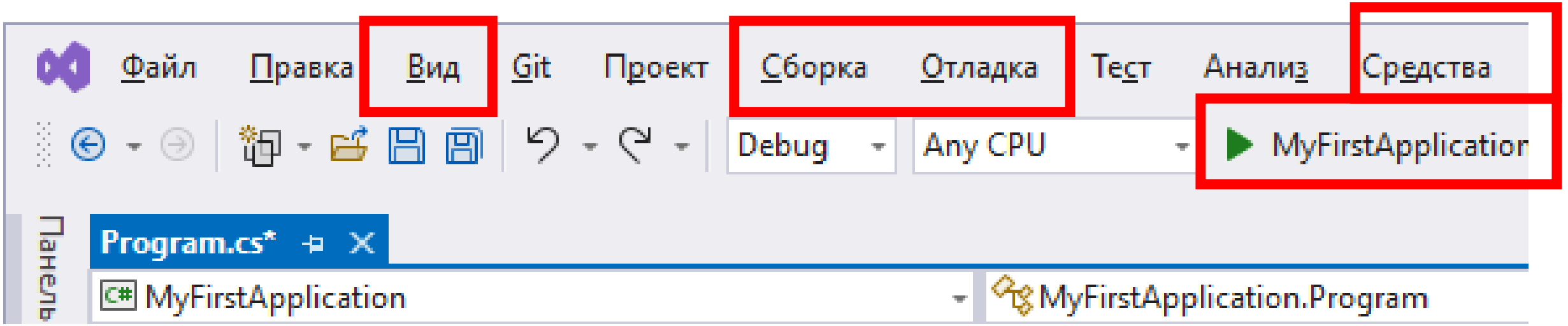
5. Далее выбираем расположение проекта

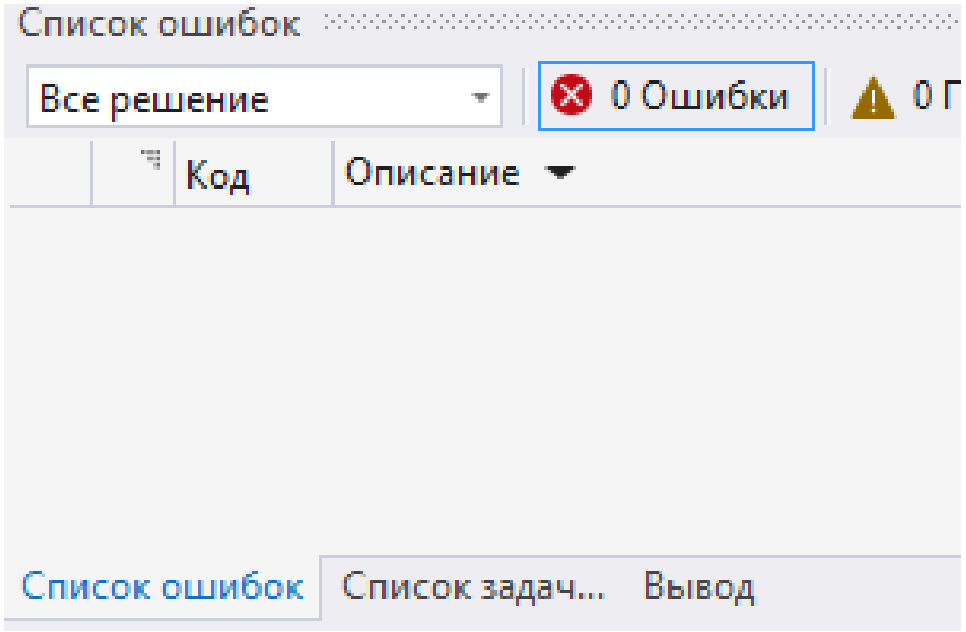


6. Выбираем актуальную платформу (.Net 7, .Net 8 и далее)

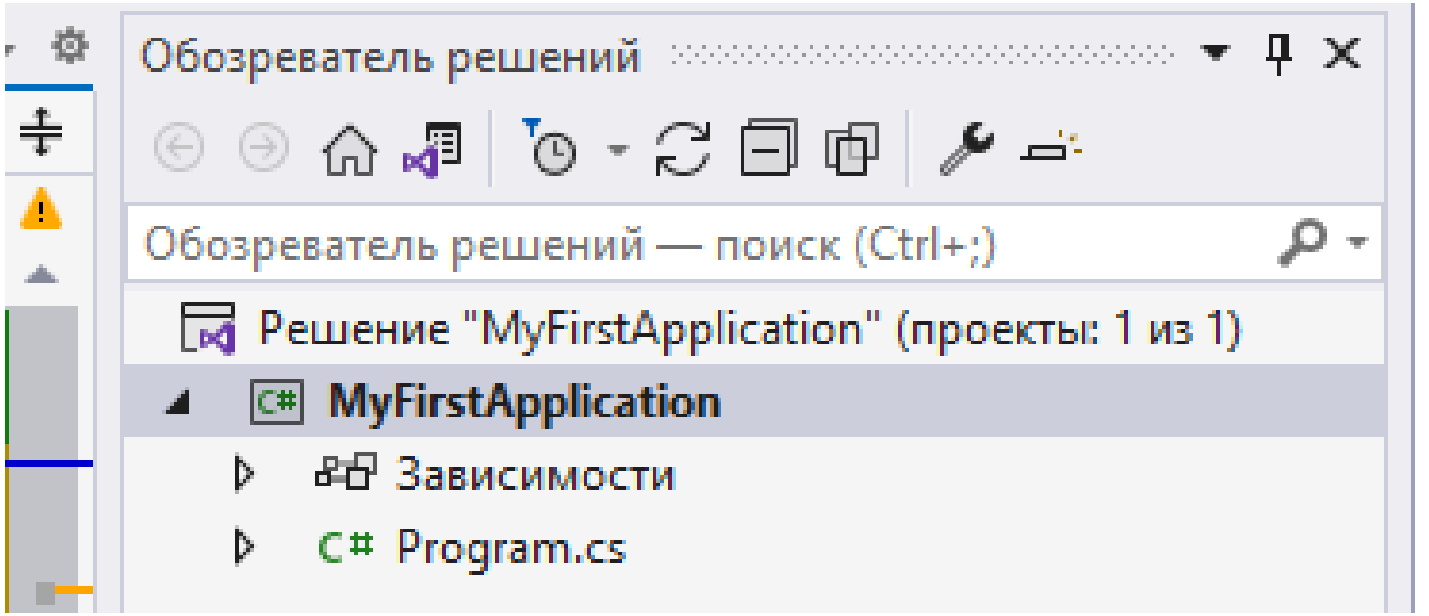



Файл Правка Вид Git Проект Сборка Отладка Тест Анализ Средства Расширения Окно Справка
Поиск
ConsoleApp1
Live Share
Debug
Any CPU
ConsoleApp1
Program.cs*
ConsoleApp1
1 // Здесь пишут программный код
2 Console.WriteLine("Hello, World!");
3
Обозреватель решений
Обозреватель решений — пс
Решение "ConsoleApp1" (1 пр
ConsoleApp1
Зависимости
Program.cs
Свойства
311 %
Проблемы не найдены.
Стр: 3 Симв: 1 Пробелы CRLF
Список ошибок
Все решение
0 Ошибки
0 Предупреждения
0 Сообщения
Сборка и IntelliSense
Поиск по списку ошибок
Описание
Проект
Файл
Ст...
Список ошибок
Вывод
Результаты метрик кода
Готово
Добавить в систему управления версиями
Выбрать репозиторий





Список ошибок



Файлы программы

Обозреватель решений (Solution Explorer)

Где найти: Вид → Обозреватель решений (или Ctrl + Alt + L).

Зачем нужно:

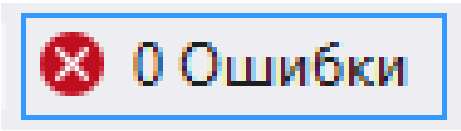
- Показывает структуру вашего проекта (файлы, папки, зависимости).
- Позволяет быстро переключаться между файлами (например, Program.cs).

Вкладка "Вид" (View)

Содержит инструменты для навигации по коду и отображения дополнительных окон.

Полезные пункты:

- Ошибки (Ctrl + \, E) — список ошибок компиляции.



- Вывод (Ctrl + Alt + O) — логи сборки и запуска программы.

Вкладка "Сборка" (Build)

Отвечает за компиляцию проекта.

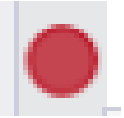
Основные действия:

- Собрать решение (Ctrl + Shift + B) — проверить код на ошибки без запуска.
- Очистить решение — удалить временные файлы сборки.

Вкладка "Отладка" (Debug)

Здесь настраивается запуск программы в режиме отладки.

Ключевые кнопки:

- Запуск с отладчиком (F5) (остановка на ошибках). 
- Запуск без отладки (Ctrl + F5). 
- Точки останова (F9 на строке кода). 

Комментарии

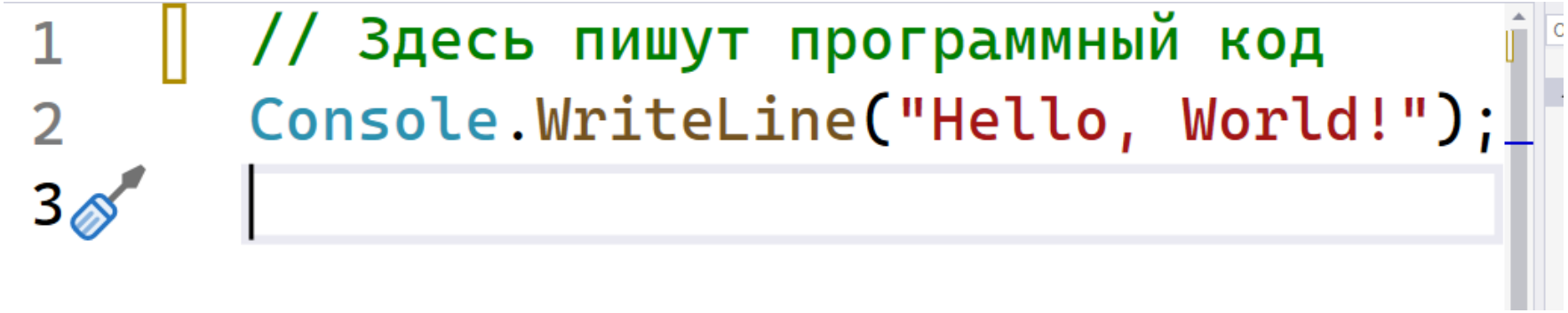
Комментарий – это текст, который предназначен только для читающего программу человека и компилятором игнорируется.

Виды:

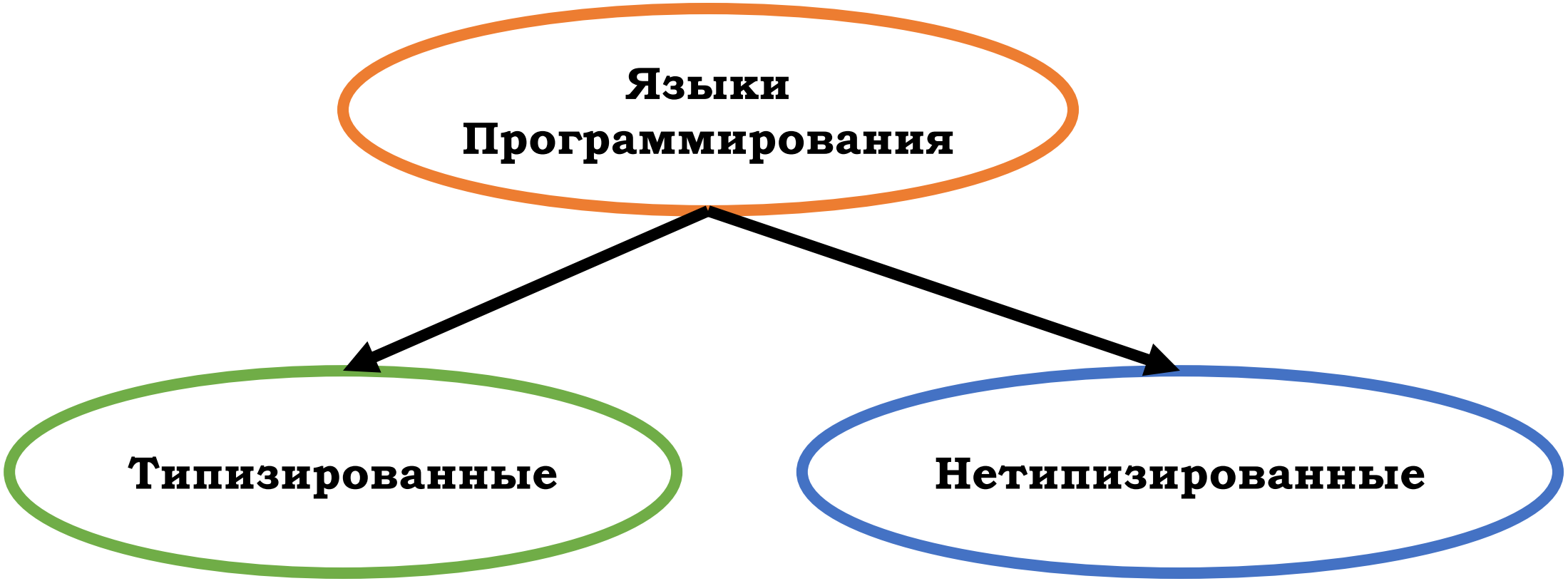
1. Однострочный;
2. Многострочный;
3. XML – комментарий.

```
//Способ 1. Однострочный комментарий

/*
 Способ 2
 Многострочный комментарий
 */
```



Система типов



Типизированные языки: C, C#, Java, Python, PHP.

Нетипизированные языки: язык ассемблера, Forth.

Тип данных это

- множество допустимых значений, которые могут принимать данные, принадлежащие к этому типу;

  Примеры:

  Целые числа – 1, 2, 3 и т.д.

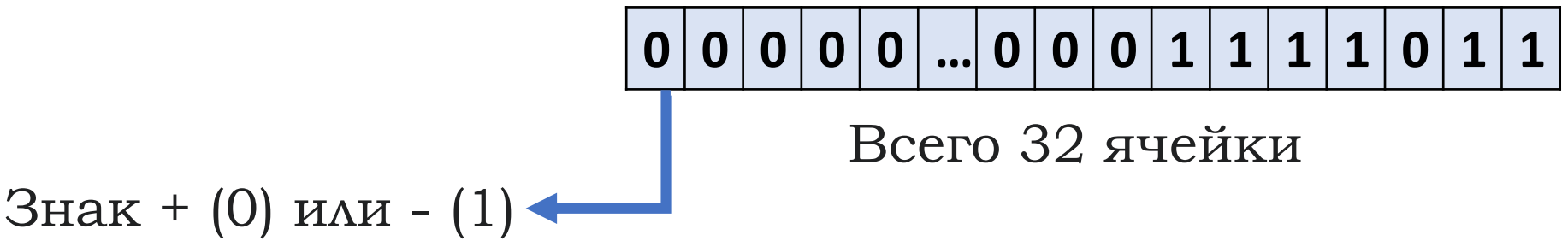
  Вещественные числа 1,5; 3,14; 0,123 и т.д.

- набор операций, которые можно осуществлять над данными, принадлежащими к этому типу.

  Примеры: сложение, умножение, деление, вычитание и др.

Тип данных	Псевдоним .NET Framework	Размер, байт	Диапазон
byte	System.Byte	1	-128...127
sbyte	System.SByte	1	0...255
short	System.Int16	2	-32768...32767
ushort	System.UInt16	2	0...65535
int	System.Int32	4	-2 147 483 648 ... 2 147 483 647
uint	System.UInt32	4	0 ... 4 294 967 295

Хранение числа **123** типа **Int32** в памяти:



Тип данных	Псевдоним .NET Framework	Размер, байт	Диапазон
long	System.Int64	8	-9 223 372 036 854 775 808 ... 9 223 372 036 854 775 807
ulong	System.UInt64	8	0 ... 18 446 744 073 709 551 615
char	System.Char	2	Символ Юникода
string	System.String	14+2*кол-во символов	Символ Юникода
bool	System.Boolean	1	True, False

Тип данных	Псевдоним .NET Framework	Размер, байт	Диапазон
float	System.Single	4	$\pm 1{,}5 \times 10^{-45} \ldots \pm 3{,}4 \times 10^{38}$
double	System.Double	8	$\pm 5 \times 10^{-324} \ldots \pm 1{,}7 \times 10^{308}$
decimal	System.Decimal	16	$\pm 1{,}0 \times 10^{-28} \ldots \pm 7{,}9 \times 10^{28}$

Вещественные числа представляются в памяти в соответствии со стандартом **IEEE754**, пример для числа **206,116** типа **float**:

Переменная — это именованная область памяти, предназначенная для хранения данных определённого типа, значение которой может изменяться во время выполнения программы.

Объявление переменной — это указание её имени и типа в исходном коде программы. Оно сообщает компилятору или интерпретатору о существовании переменной, но не выделяет под неё память.

```
int myIntVariable;
float myFloatVariable;
char myCharVariable;
```

Инициализация – это процесс присвоения начального значения при создании переменной. Перед инициализацией происходит выделение памяти.

<Тип данных> <Имя переменной> = <Значение>;

```
int myIntVariable;
myIntVariable = 11; ИЛИ int myIntVariable = 11;

float myFloatVariable = 10.2f;
char myCharVariable = 'q';
decimal myFloatVariable = 10.2m;
```

char — это тоже число (код символа), просто компьютер отображает его как букву. Поэтому 'A' + 1 будет работать.

decimal — используется для денег (из-за повышенной точности). Это единственное, что мы должны запомнить про него сейчас.

Вопросы

Какой тип данных нужно выбрать если мы хотим хранить в нем:

1) Целые числа от 1 до 100
2) Целые числа
3) Натуральные числа от 1 до 1000
4) Огромные целые числа > 10 000 000
5) Единичные символы, например, ‘a’; ‘ь’; ‘1’; ‘@’ …
6) Символы и, возможно, слова: «Вася»; «Петя»; «012345»; «Я» …
7) Вещественные числа
8) Вещественные числа, когда не требуется большое количество знаков после запятой (не нужна высокая точность): 0,15; 0,3; -0,0025
9) Очень большие вещественные числа
10) Вещественные числа, когда требуется высокая точность (много знаков после запятой)
11) Выражения, для которых всего 2 варианта ответа, например, «Да» или «Нет».