

# РАЗРАБОТКА НА C++

## Урок 1. Введение

# РАЗРАБОТКА НА C++

Что такое C++ и с чем его едят?

# Что такое программирование?

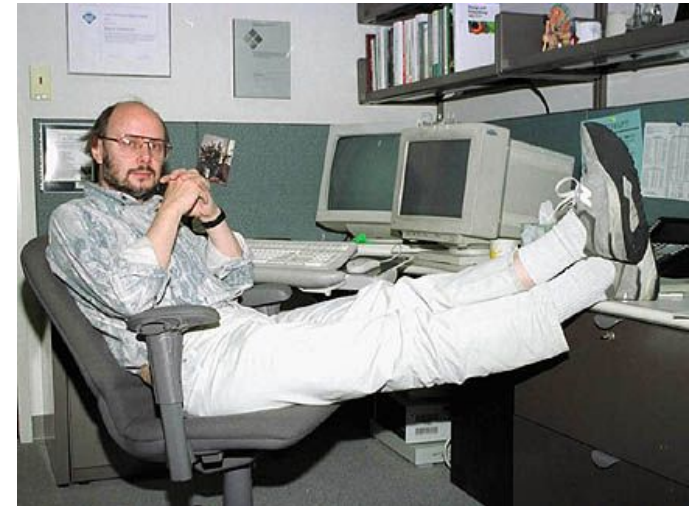
**Программирование** - это процесс создания набора инструкций, которые компьютер может понять и выполнить. Эти инструкции называются *программами*, и они определяют, какие действия должны быть выполнены компьютером для достижения конкретных целей.

Программирование позволяет нам создавать разнообразные приложения, веб-сайты, игры, программы для автоматизации задач, и многое другое.

# Что такое C++?

**C++** - это высокоуровневый язык программирования, который является расширением языка C.

Он был разработан в начале 1980-х годов Бьёрном Страуструпом (Bjarne Stroustrup) в компании Bell Labs в США.

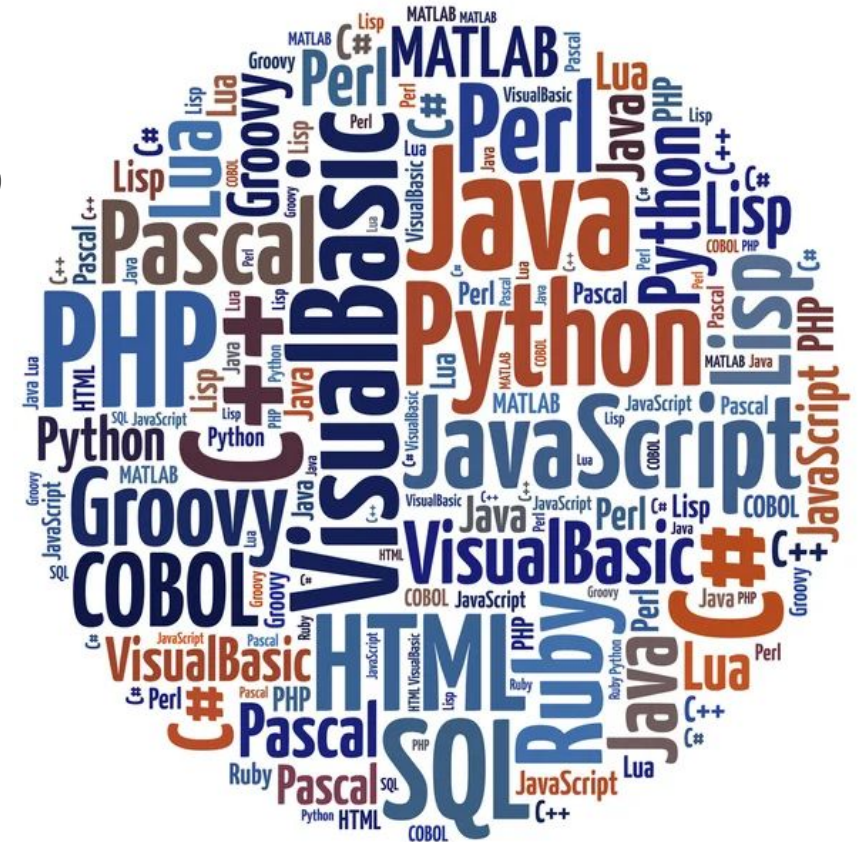




# Что такое язык C?

До появления C++, одним из наиболее популярных языков программирования был язык C. Он был разработан в 1972 году и быстро стал популярным для системного программирования и разработки операционных систем.

Кроме того, в то время были также популярны языки программирования, такие как **Fortran** (для научных вычислений), **COBOL** (для бизнес-приложений) и **Pascal** (для образовательных целей)



# Зачем изучать C++?

- “На Python можно написать всё что угодно, а на C++ можно написать Python”
- C++ крайне востребован как в России, так и во всём мире из-за своей специфики написания более низкоуровневого кода.
- Изучение одного из самых сложных языков программирования позволит в будущем быстрее и эффективнее изучать другие инструменты IT-сферы.
- C++ часто интегрируют с кодом написанным на других языках, для повышения производительности или оптимизации ресурсов.
- C++ очень близок к “железу”, поэтому его изучение также поможет понять как работает оперативная память, видеокарта и процессор.

# Зачем изучать C++?

По данным 2023 года рейтинг языков программирования по GitHub такой:

- 1 место — Python;
- 2 место — Java;
- 3 место — Go;
- 4 место — C++;
- 5 место — JavaScript;

C++ будет востребован всегда, но писать на нём абсолютно всё бывает невыгодно в плане времени и ресурсов, так как кода в среднем в 3-4 раза больше чем на Python или Java. При этом для написания расширяемого и высокопроизводительного кода C++ незаменим.

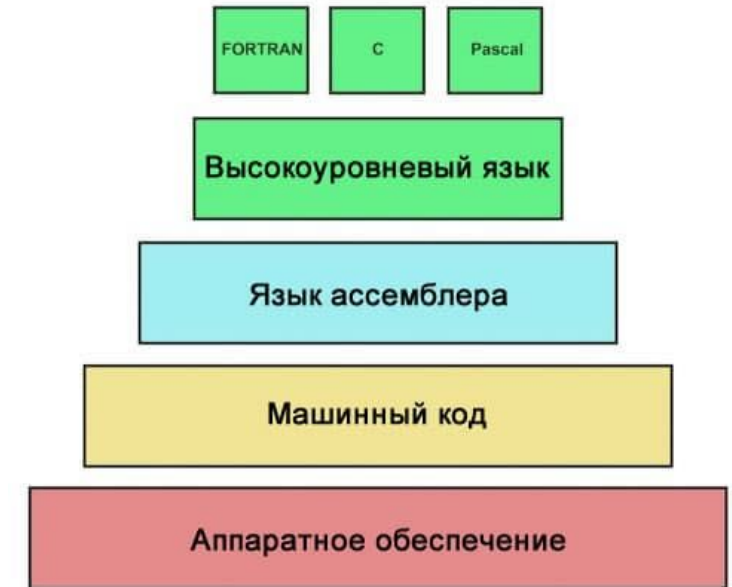


# Какие бывают языки?

**Низкоуровневый язык программирования** — язык программирования, близкий к языку единиц и нулей. Низкоуровневые языки больше понятны компьютеру, чем человеку.

**Высокоуровневый язык программирования** — язык программирования, который более понятен человеку, чем компьютеру.

К языкам высокого уровня относятся практически все популярные языки программирования, включая C++.





# Где чаще используют разные языки программирования?

## **C++ для разработки игр и графических движков:**

Это связано с высокой производительностью C++, что необходимо для обеспечения плавной и реактивной игровой динамики, освобождения ресурсов в памяти. Например, игры с трехмерной графикой, такие как Grand Theft Auto V и Call of Duty и многие другие, разрабатываются с использованием C++

## **Java для мобильных приложений:**

Java используется для создания приложений на платформе Android. Примером может служить приложение WhatsApp, которое разработано с использованием Java. Java также известен своей переносимостью между разными платформами и огромным количеством фреймворков и библиотек для быстрой разработки и оптимизации.

## **Python для веб-сайтов и быстрой разработки:**

Python предоставляет простой и чистый синтаксис, что делает его отличным выбором для быстрой разработки прототипов будущих программ. Также язык обладает библиотеками для удобной работы с веб-приложениями, например: Instagram использует фреймворки Python для своей веб-платформы.

*Важно отметить, что выбор языка зависит от конкретных потребностей проекта и предпочтений разработчика. В реальных проектах часто используются разные языки и технологии для разных компонентов приложения или системы.*

# Игры созданные с использованием C++

C++ - это один из наиболее распространенных языков программирования в **индустрии видеоигр**, и множество игр было разработано с его использованием:



# Немного про Minecraft :)

**Minecraft:** Хотя игра написана на Java, ее множество модов и плагинов, в том числе и клиентские модификации, используют C++ для оптимизации и расширения функциональности.

Существует также версия Minecraft, использующая C++:

Эта версия Minecraft, называемая "Minecraft: Bedrock Edition" или "Minecraft (Bedrock)", является *мультиплатформенной* версией игры (доступна на разных устройствах, включая Windows 10, Xbox, iOS, Android и другие платформы). Ее код частично написан на C++, что обеспечивает высокую производительность и возможность запуска на разных устройствах.

# Что такое игровой движок?

Это как невидимый мозг, который делает видеоигру работающей. Он содержит правила и инструкции:

- как отображать графику
- как обрабатывать действия игроков
- как создавать мир внутри игры

Игровой движок позволяет создателям игр **сосредотачиваться на создании контента** и взаимодействии с игроками, не заботясь о сложных деталях технической реализации игры.



# Игровые движки

Существует множество игровых движков и фреймворков для разработки игр на C++. Некоторые из наиболее популярных и известных включают:

**Unreal Engine** - является одним из самых мощных и популярных игровых движков. Он использует C++ для программирования игровой логики и оснащен богатым набором инструментов для создания 2D и 3D игр. Этот движок разработан компанией Epic Games.

**Unity** - это многоплатформенный игровой движок, который поддерживает C# и C++ для разработки игр. C++ чаще используется для создания нативных плагинов и оптимизации.

**CryEngine** - является мощным движком, использующим C++ для разработки высококачественных графических игр, таких как серия Crysis. Компания Crytek.

**Godot Engine** - это бесплатный и открытый исходный код игровой движок, который поддерживает как GDScript (схожий с Python), так и C++. Он отличается легковесностью и простотой использования.



# Плюсы C++

- **Высокая производительность:** C++ предоставляет близкое к металлу управление памятью и ресурсами, что позволяет создавать высокопроизводительные приложения, особенно в области игр и системного программирования.
- **Универсальность:** C++ можно использовать для разработки широкого спектра приложений, от встроенных систем и мобильных приложений до серверных приложений и научных вычислений.
- **Объектно-ориентированное программирование (ООП):** C++ поддерживает ООП, что делает его более организованным и повышает переиспользуемость кода.
- **Близкое к железу программирование:** C++ позволяет программистам более прямо управлять аппаратными ресурсами, что полезно в системном и встраиваемом программировании.
- **Большое сообщество и ресурсы:** C++ имеет большое и активное сообщество, а также обширную документацию и ресурсы для обучения.

# Минусы C++

- **Сложность:** C++ - это сложный язык с множеством возможностей и нюансов, что может сделать его изучение и разработку более трудоемкими.
- **Управление памятью\*:** В C++ нет автоматического управления памятью, и это может привести к ошибкам, таким как утечки памяти или обращения к освобожденной памяти (\*с другой стороны программист должен работать с памятью на прямую, оптимизируя программу под конкретные задачи)
- **Переносимость кода\*:** Из-за различий в компиляторах и архитектурах, код на C++ может быть менее переносимым между разными платформами. (\*важно отметить что речь идет про чистый C++, который мы будем изучать на этом курсе)
- **Меньшая скорость разработки:** Написание кода на C++ может быть более медленным и трудозатратным процессом, чем на более высокоуровневых языках.
- **Отсутствие встроенных инструментов для некоторых задач:** Для некоторых задач, таких как обработка текста или создание веб-приложений, C++ может быть менее удобным по сравнению с другими языками.

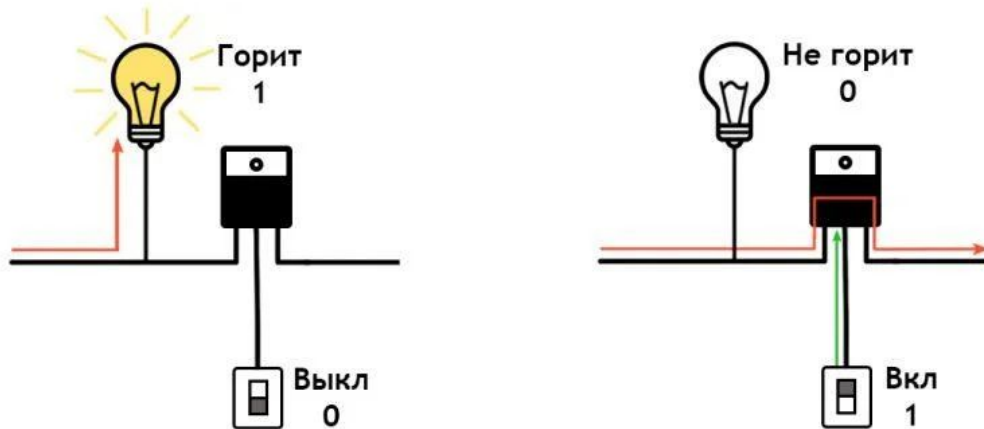
# РАЗРАБОТКА НА C++

В чем состоит задача программиста?

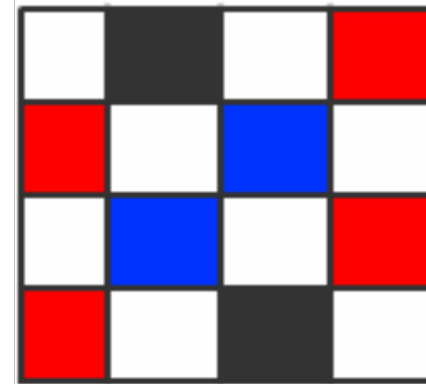
# Физика в программировании

Всё что отображает монитор, вычисляет компьютер, а также действия при нажатии на клавиши завязано на физических процессах: сигналы, транзисторы, устройство железа компьютера.

Логическая операция НЕ (Not)



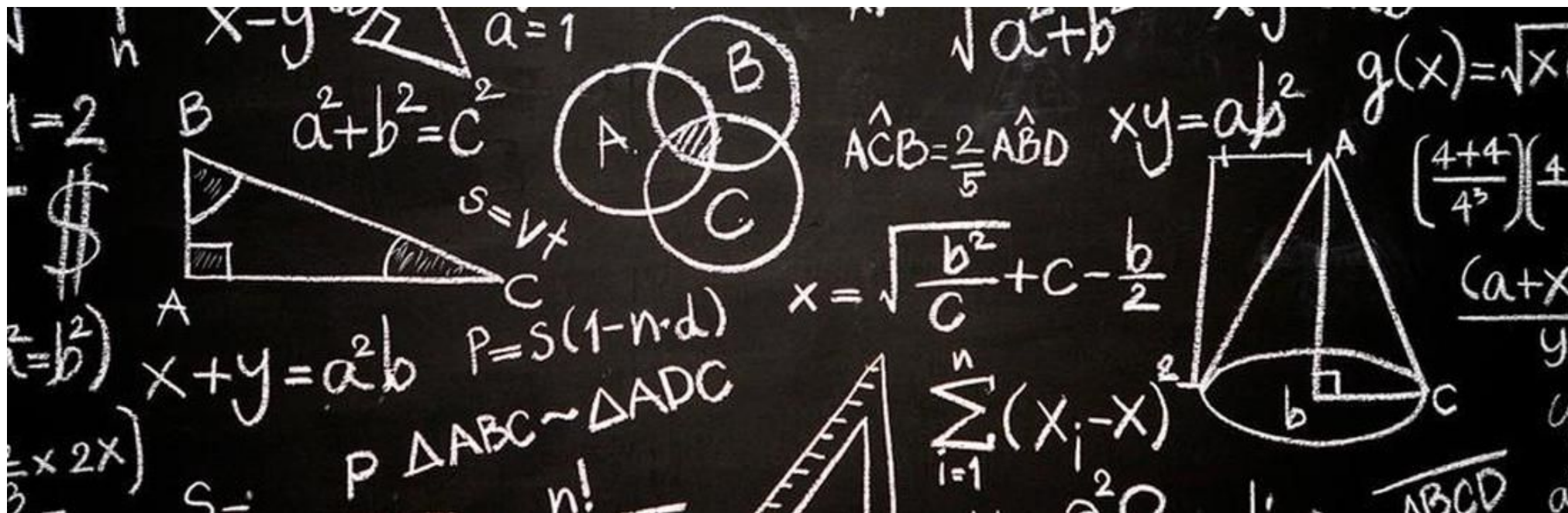
1 0 0 1  
0 1 1 0  
0 1 0 0  
1 0 0 1



00 11 00 01  
01 00 10 00  
00 10 00 01  
01 00 11 00

# Физика в программировании

Так или иначе физика и математика программисту нужны, так как есть необходимость разрабатывать алгоритмы для сложных вычислений и отображения графики





# В чём заключается задача программиста?

Попробуйте решить пример без калькулятора:

$$5+10$$

Теперь усложним задачу:

$$5/10-13*2$$

Ну это все не серьезно, давайте так:

$$12.24155 * 2561.1129$$

А теперь:

$$\int_{-3}^1 (2x^2 + 3x - 1) dx$$

ответ: 2.66(7)

Конечно же мы всё это посчитаем, **но как быстро?** А что если такие примеры будут поступать **десятками тысяч в минуту?**

Наша задача автоматизировать, мы можем заставить компьютер считать за нас



# В чём заключается задача программиста?

В современном мире начинается распространение нейросетей для рисования логотипов, картин и дизайнов. Компьютер обучают создавать сценарии фильмов, игр, книг, а также писать посты, доклады, генерировать музыку, звуки, голоса актеров и популярных людей.

С каждым годом всё больше повседневных вещей в нашей жизни и в жизни всего общества начинает делать компьютер, а инструкции для этого ему пишут программисты.

**Основная задача программиста** переложить решение задачи на компьютер

# РАЗРАБОТКА НА C++

Как происходит взаимодействие программиста с компьютером?

# История современных компьютеров

Архитектура фон Неймана - основная концепция построения компьютеров, которая была разработана Джоном фон Нейманом в середине 20-го века. Она работает для большинства современных компьютеров и устройств.

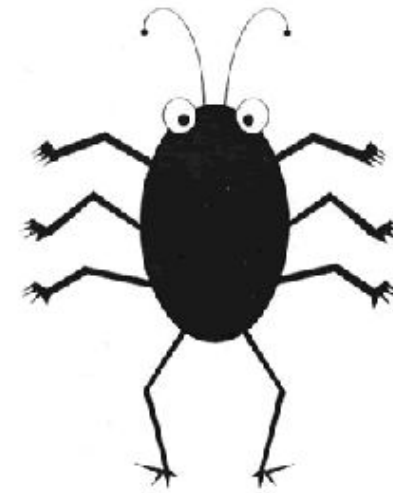
Эта концепция обеспечивает удобство и универсальность, так как **позволяет программировать** компьютер для различных задач, **изменяя только программу, но не аппаратное обеспечение.**

На более ранних этапах создания компьютеров чтобы задать программу, приходилось особым образом подсоединять провода и многие другие компоненты: это могло продолжаться много часов и даже дней. Примерно в это же время впервые появилось слово “баг”

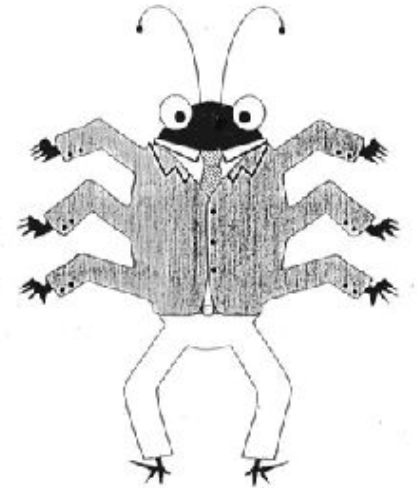
# Откуда взялось слово “баг” в IT?

**Баг** - в IT используется для обозначения ошибок или неисправностей в программном обеспечении или аппаратуре компьютера.

*Происхождение:* Инженеры обнаружили, что компьютер Марк I (1947 год) перестал работать из-за того, что внутри него застрял мотыль и вызвал сбой. Они сохранили мотыля и прикрепили к нему бумажку с записью "First actual case of bug being found" ("Первый фактический случай обнаружения бага"). Этот случай считается одним из первых документированных случаев использования термина "баг" в контексте компьютерных сбоев.



Баг



Фича



# Архитектура фон Неймана

Простыми словами, в архитектуре фон Неймана компьютер работает следующим образом:

**Память:** Все данные, включая программы, хранятся в одной и той же памяти. Это означает, что и ваши фотографии, и приложения, и даже сама операционная система находятся в одном месте.

**Центральный процессор (ЦП):** ЦП выполняет команды, которые извлекает из памяти. Он обрабатывает данные, выполняет вычисления и управляет работой компьютера.

**Управление и исполнение команд:** ЦП поочередно извлекает команды из памяти, исполняет их и затем переходит к следующей команде. Это происходит в цикле.

**Ввод и вывод данных:** Для обмена данными с внешним миром, таким как клавиатура, монитор или другие устройства, компьютер использует специальные команды в программе.

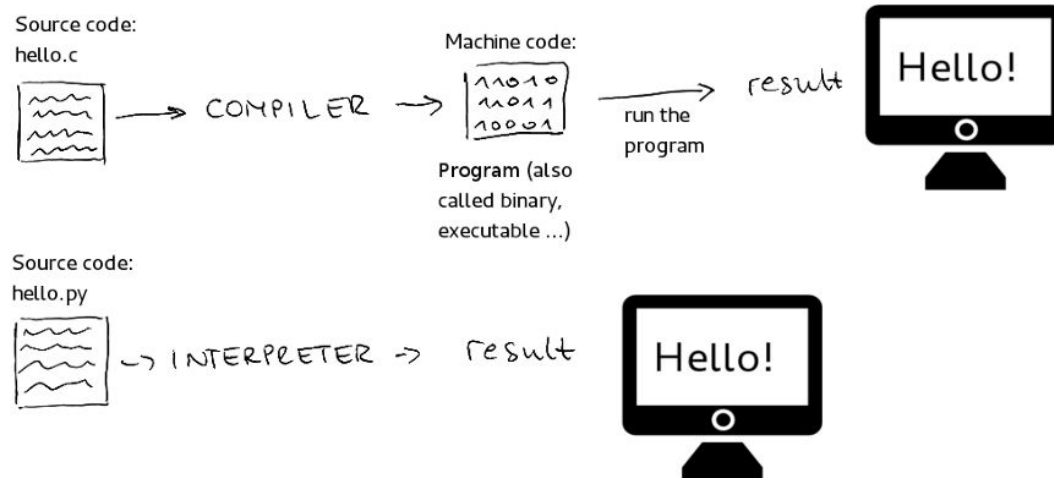
# Как происходит перевод кода на язык компьютера?

## Компилируемый язык программирования

Программы заранее и полностью переводятся в машинный код. Если в коде будут ошибки или компилятору что-то не понравится, то и программа не запустится.

## Интерпретируемый язык программирования

Программы переводятся в машинный код пооперационно при каждом запуске. Код будет продолжаться пока не дойдет до строки с ошибкой, а потом завершится.

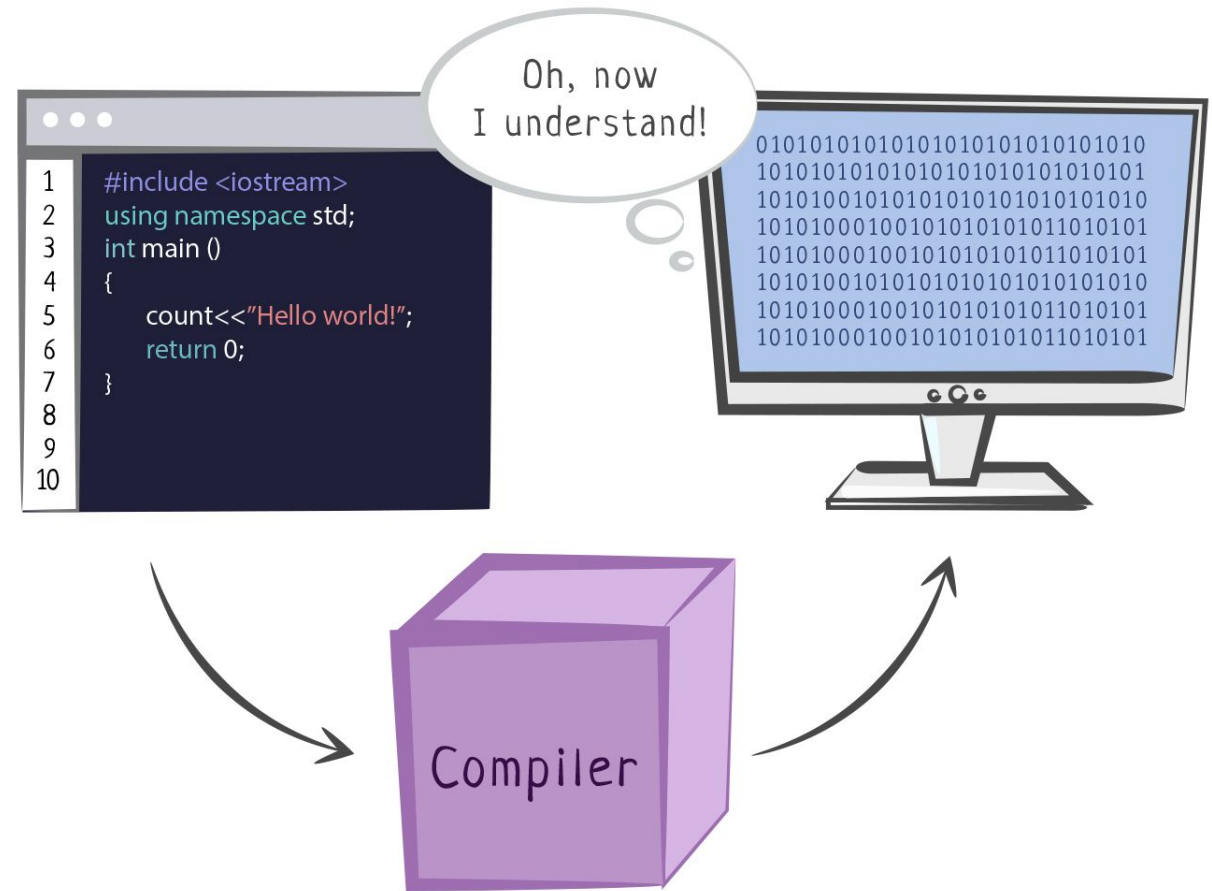


# Что такое компилятор?

**Компилятор** - программа, которая переводит код на языке программирования в машинный код.

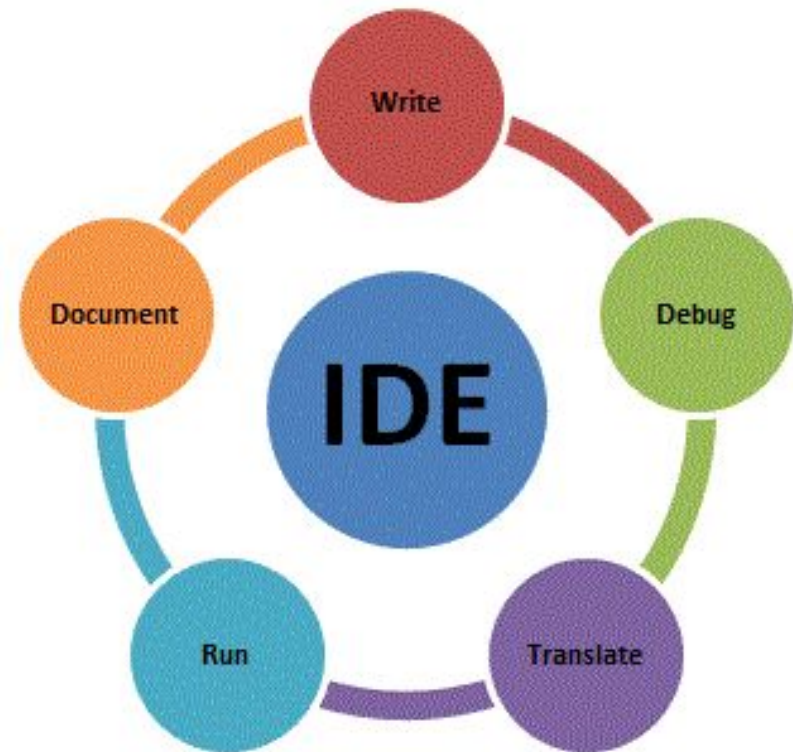
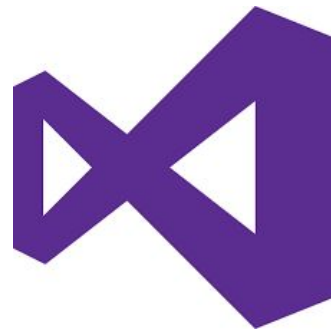
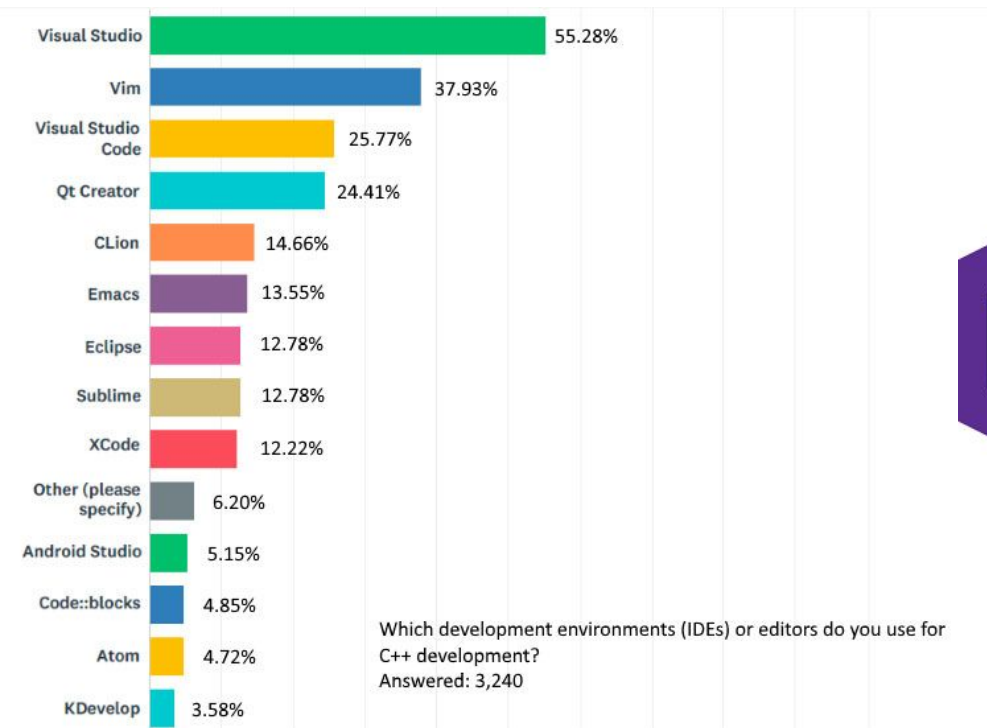
В результате этого перевода компилятор создаёт исполняемый файл, который можно запустить.

То есть компилятор — утилита-посредник.



# Что такое IDE?

IDE - это аббревиатура "Integrated Development Environment" (Интегрированная среда разработки). Это программное обеспечение, предназначенное для упрощения процесса создания, отладки и управления программами компьютерных приложений.



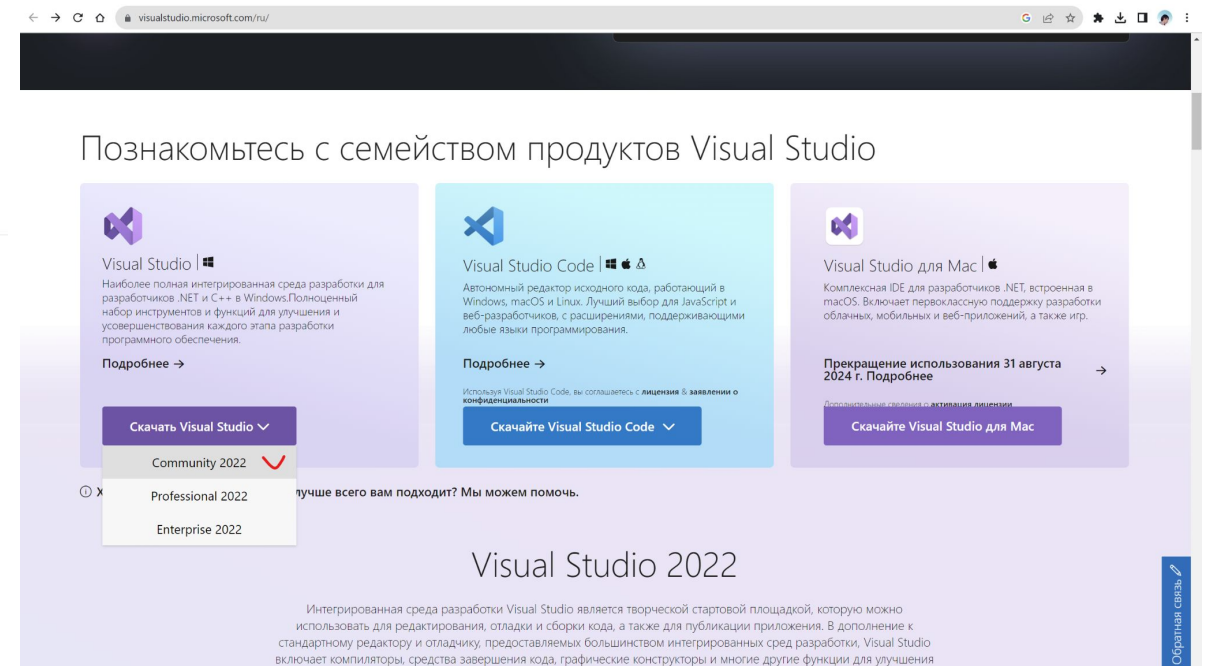
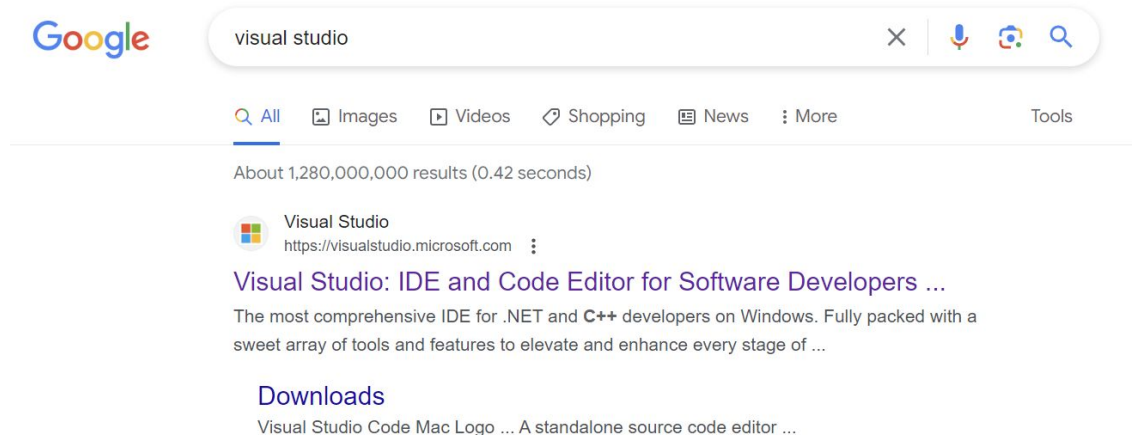
# РАЗРАБОТКА НА C++

Установка IDE



# Установка Visual Studio

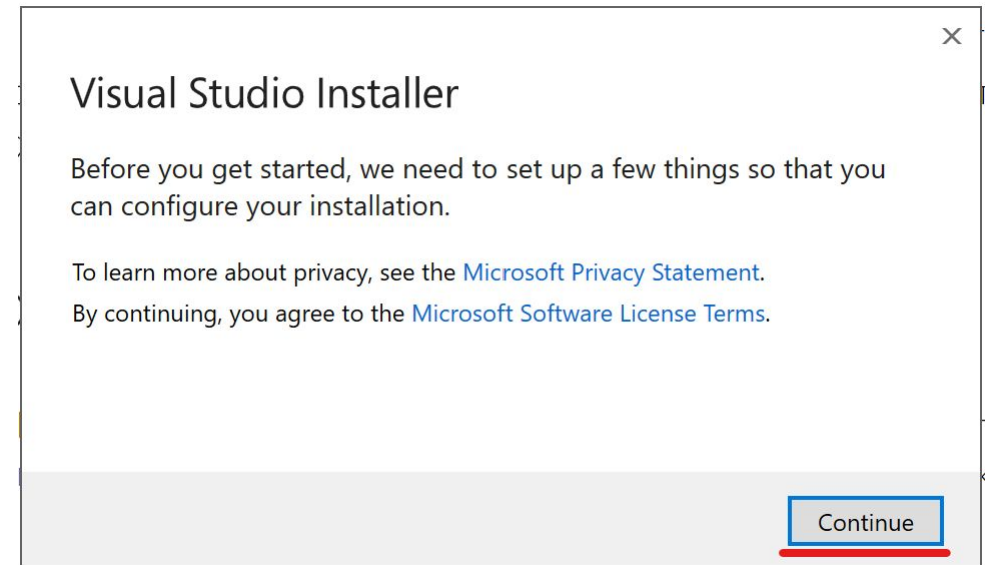
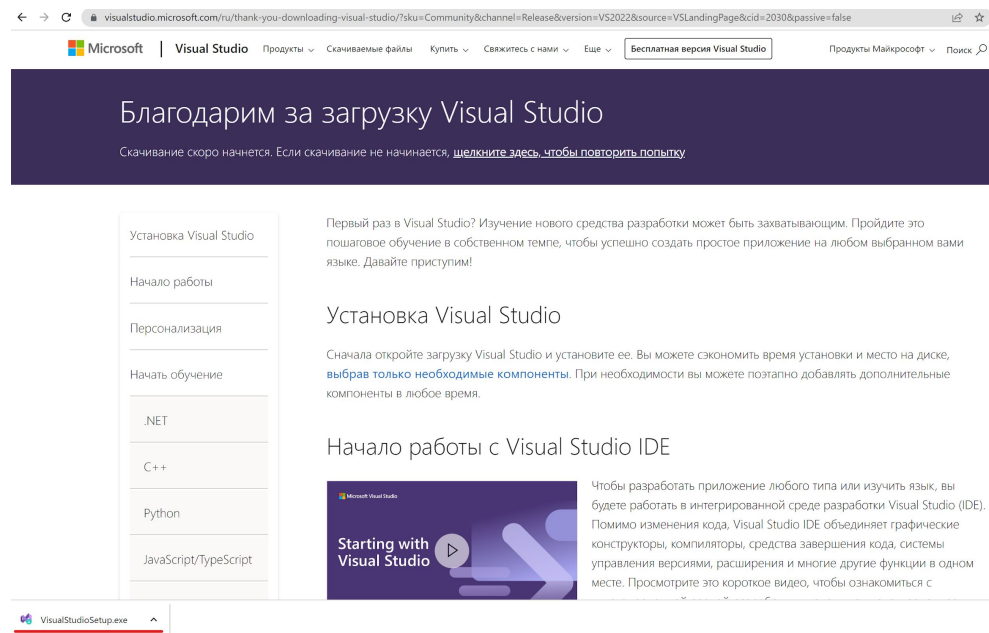
Visual Studio - это IDE, в которой мы будем работать на протяжении всех вебинаров, если у вас есть опыт или желание изучить другую IDE (например Visual Code) это никак не мешает в обучении :)



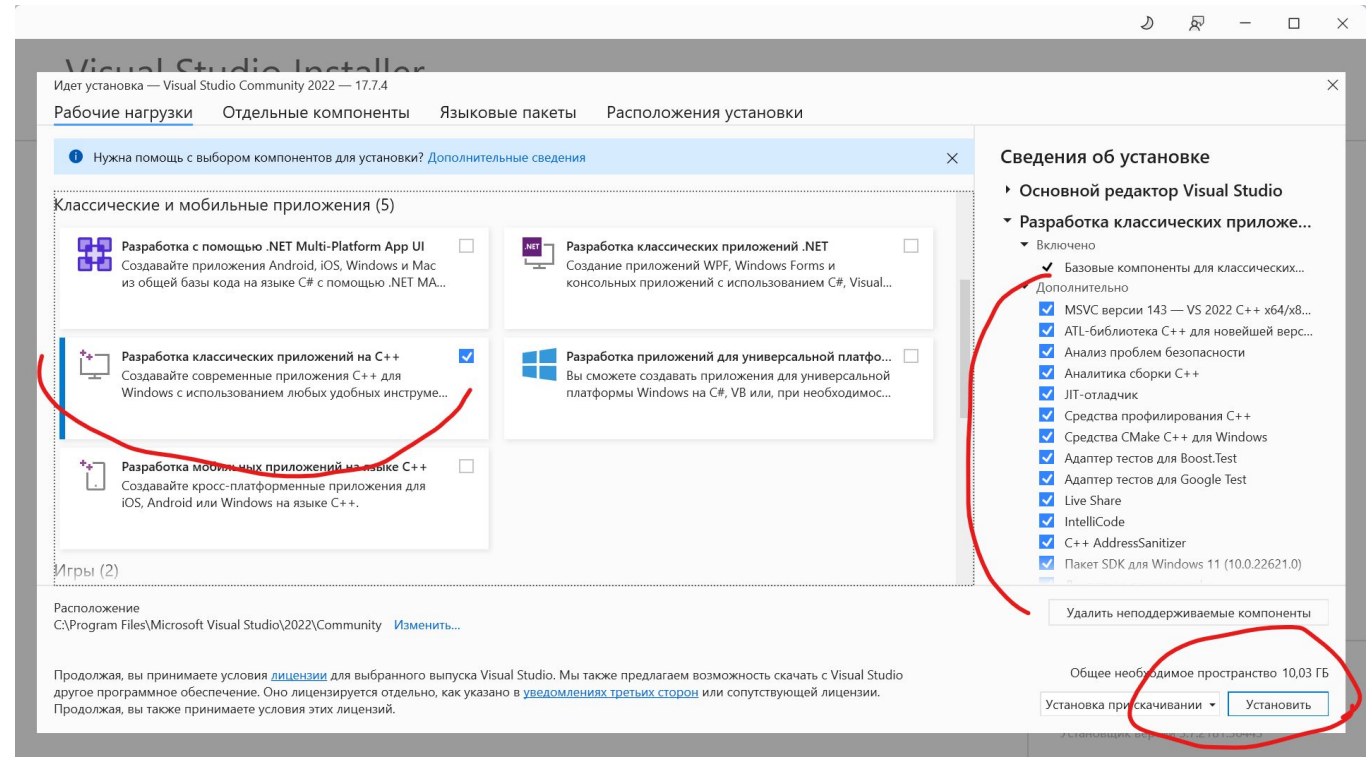
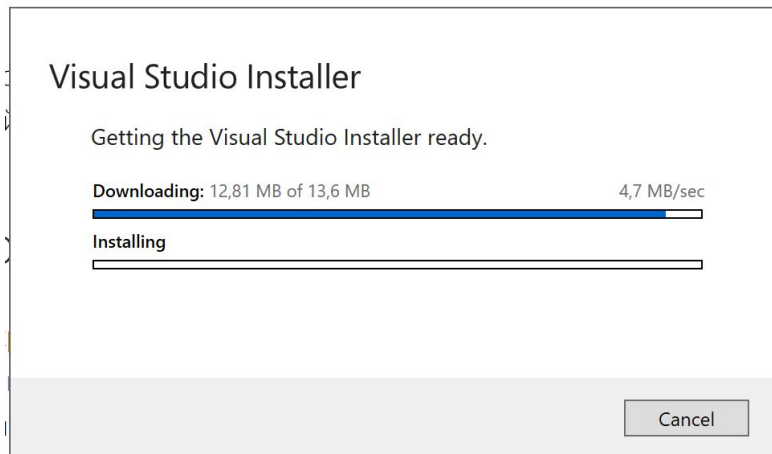
# Установка Visual Studio

Visual Studio последних версий поддерживается *только на Windows 10 и выше*, для работы в этом модуле можно использовать **онлайн компиляторы C++**, если Visual Studio не получилось установить

На Windows 7 должна работать Visual Studio 2017  
(<https://visualstudio.microsoft.com/ru/vs/older-downloads/>)

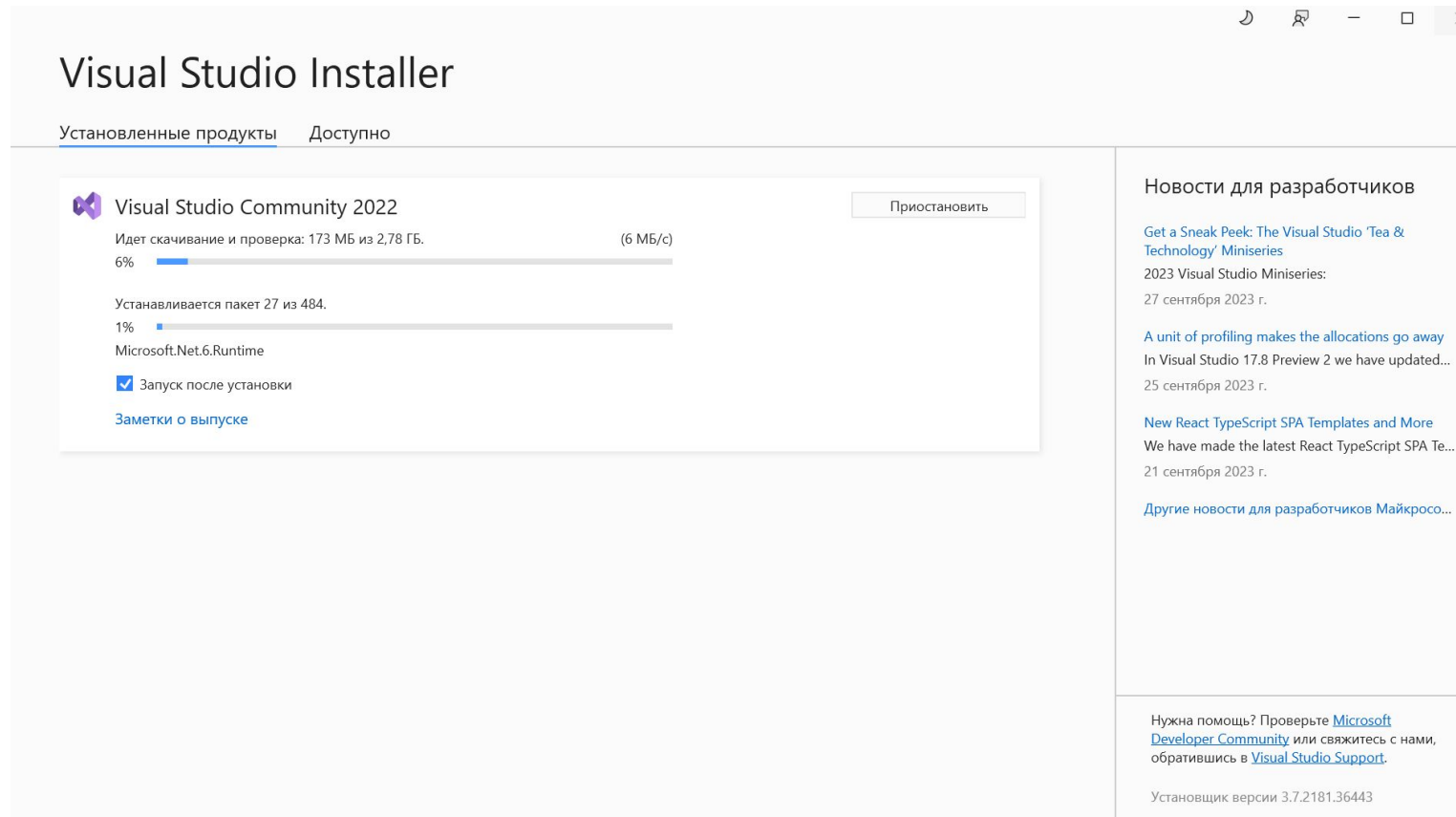


# Установка Visual Studio

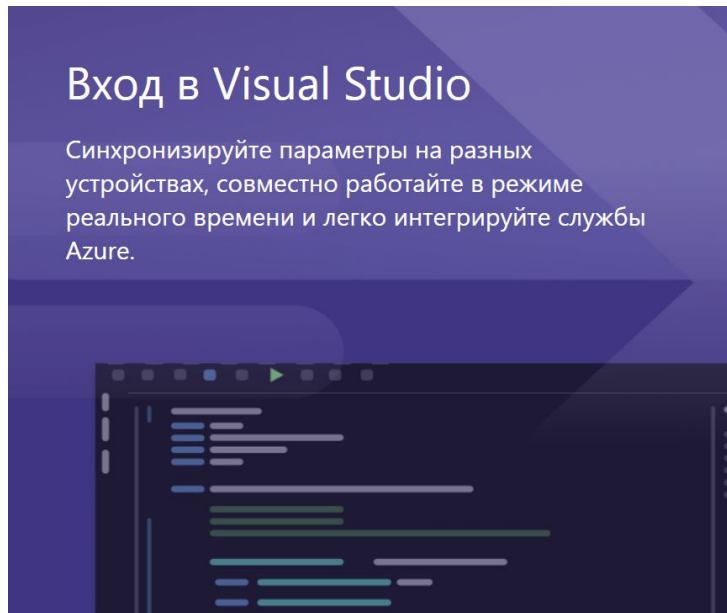


# Установка Visual Studio

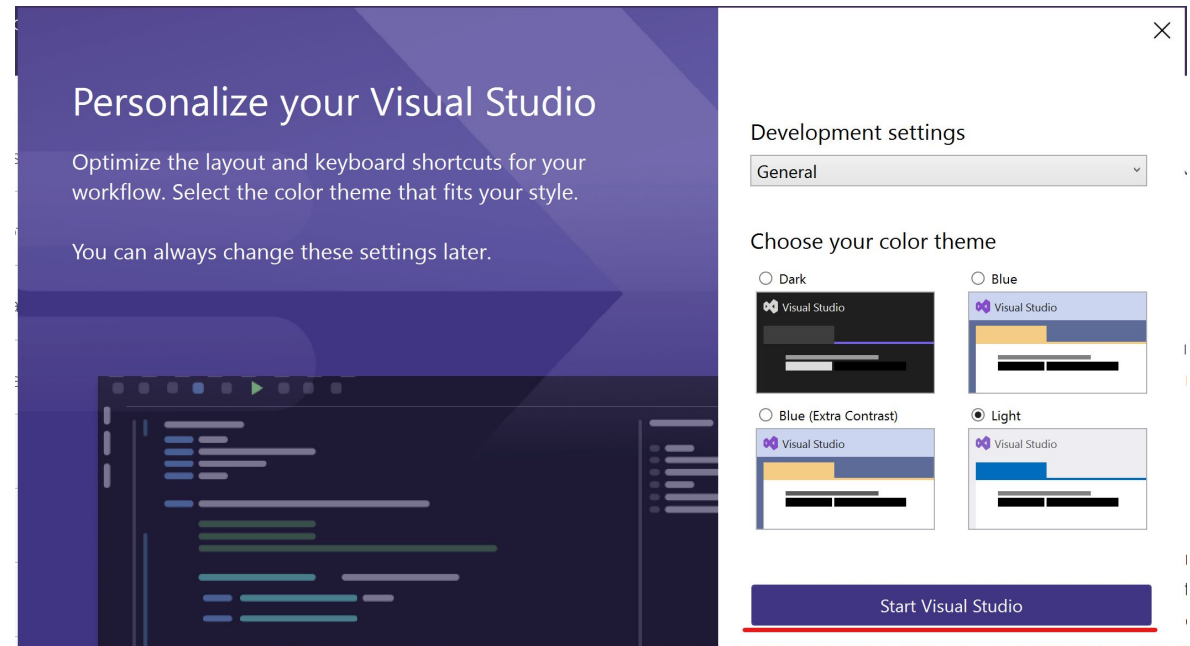
Установка в среднем от 5 до 20 минут в зависимости от скорости интернета, также после установки нужна перезагрузка.



# Установка Visual Studio



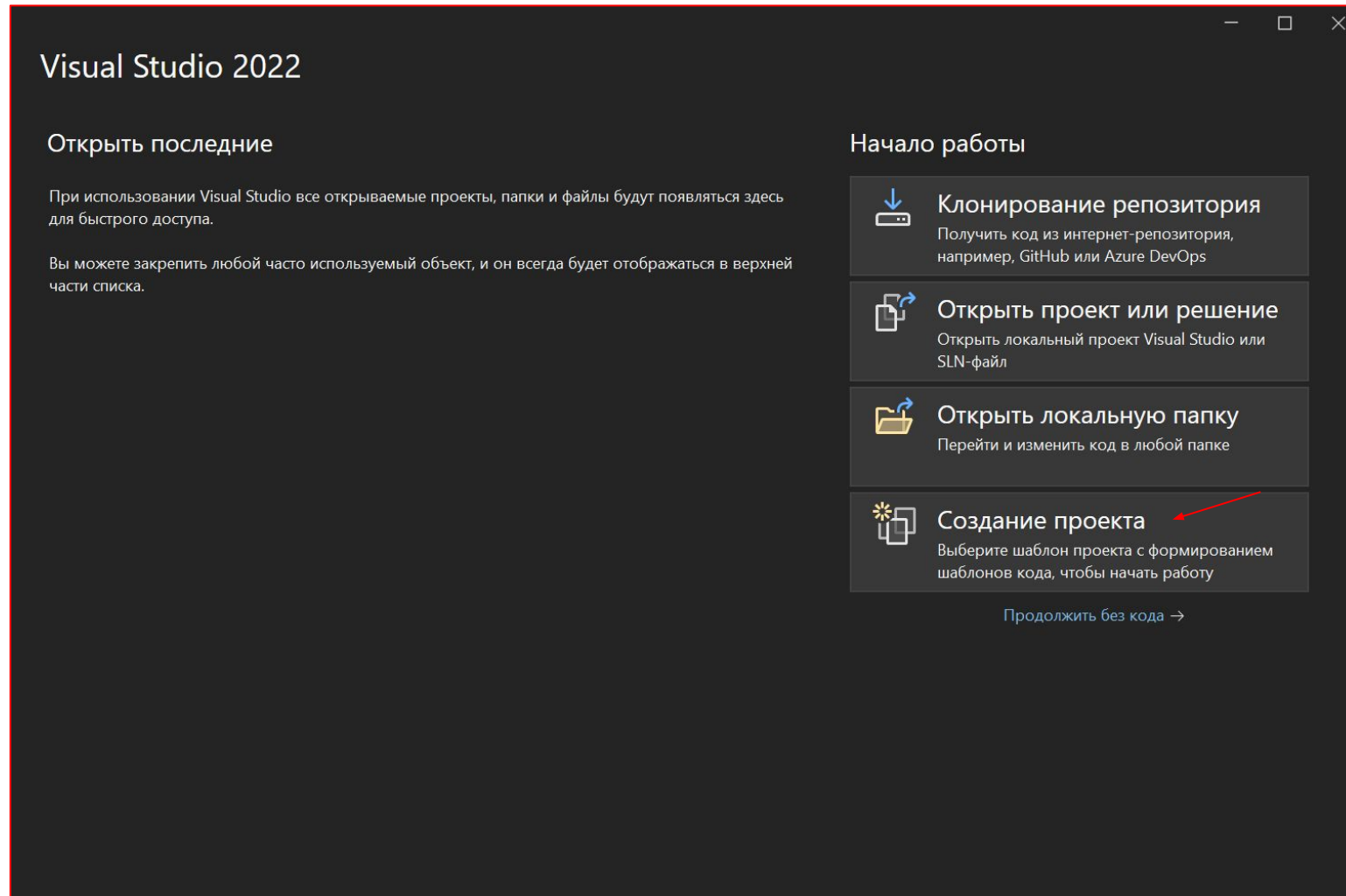
×



# РАЗРАБОТКА НА C++

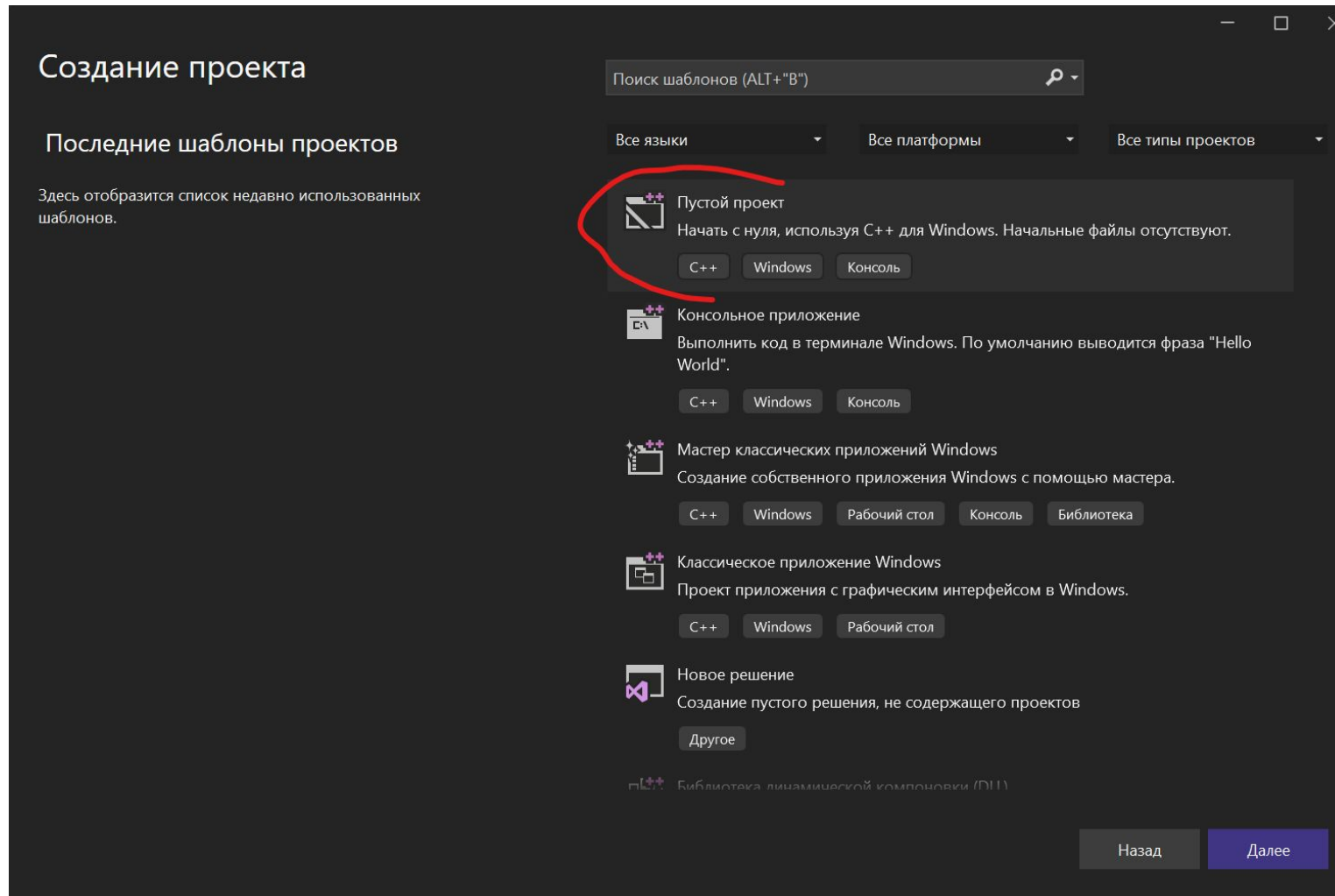
Создание первого проекта

# Создание проекта



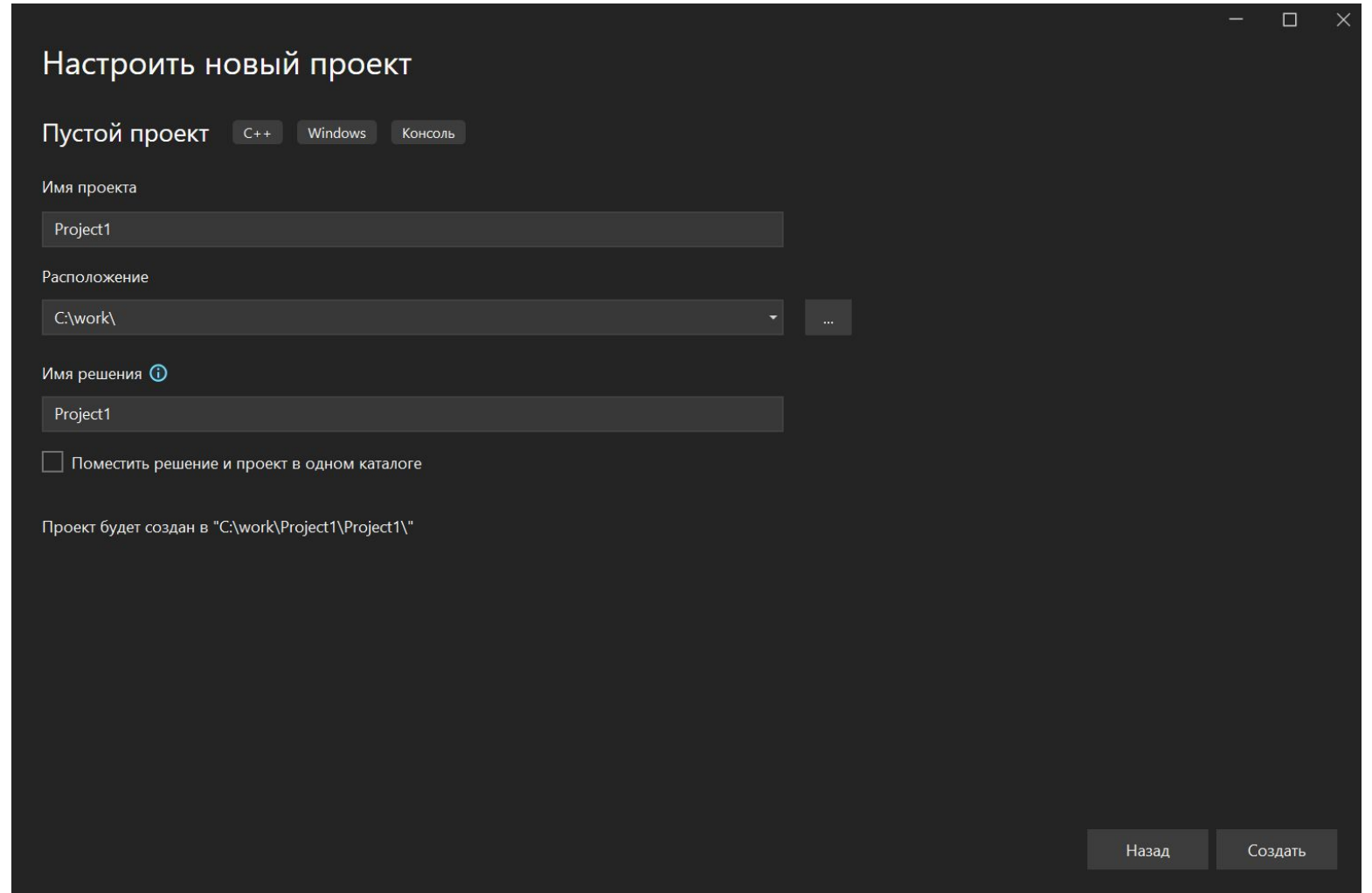


# Создание проекта



# Создание проекта

Лучше всегда создавать проект без кириллицы в названии и пути до проекта



Настроить новый проект

Пустой проект C++ Windows Консоль

Имя проекта

Project1

Расположение

C:\work\ ...

Имя решения ⓘ

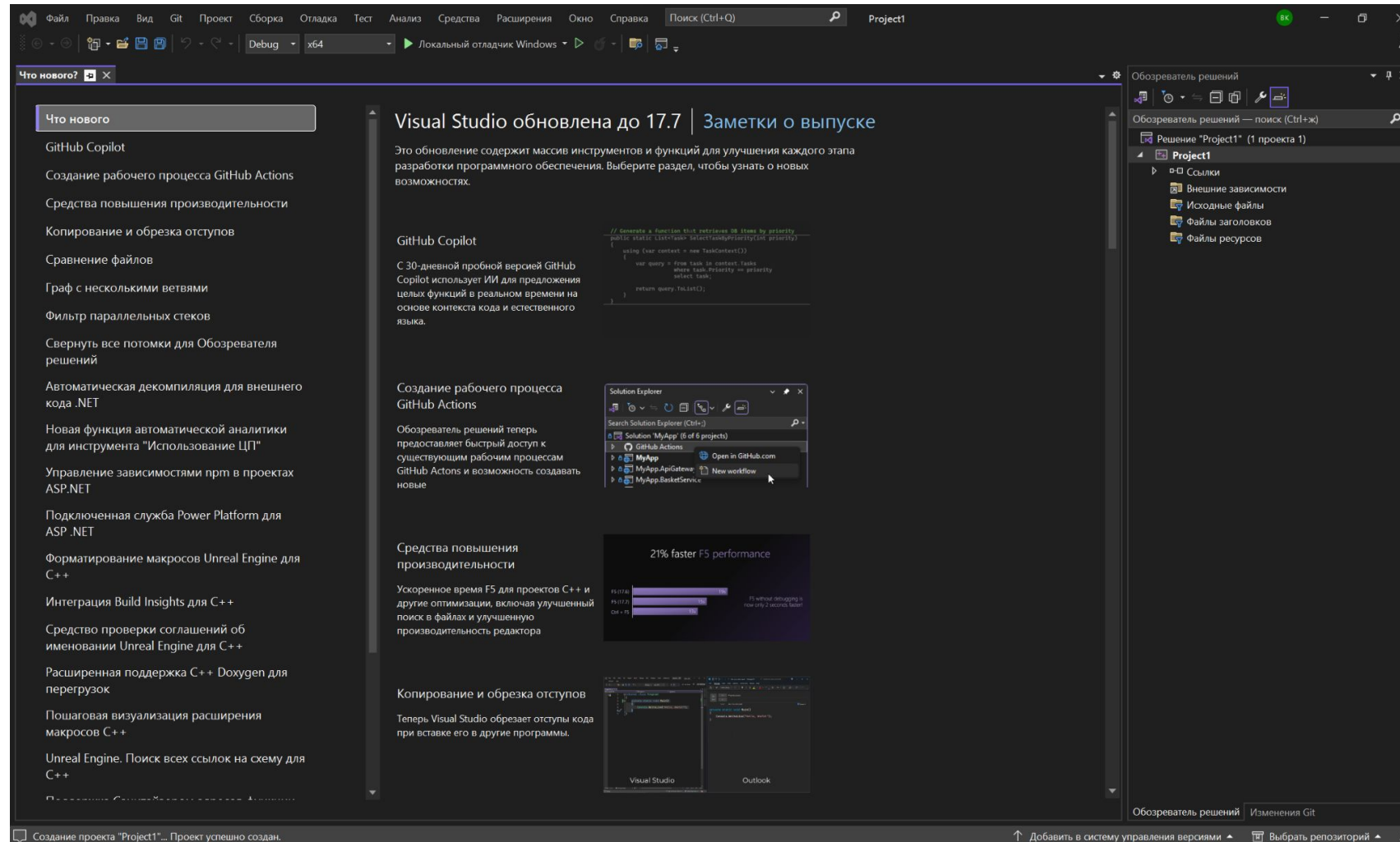
Project1

☐ Поместить решение и проект в одном каталоге

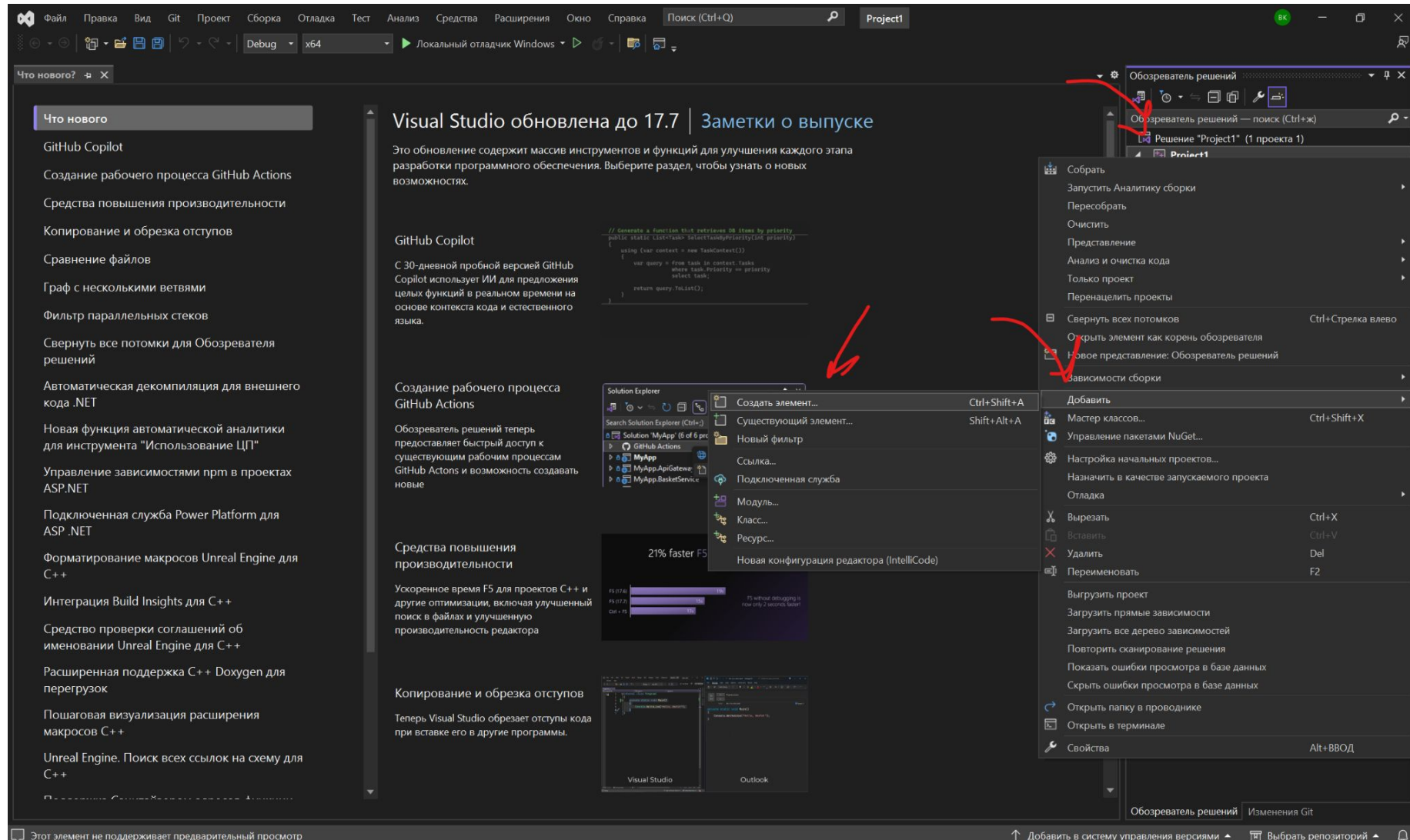
Проект будет создан в "C:\work\Project1\Project1\"

Назад Создать

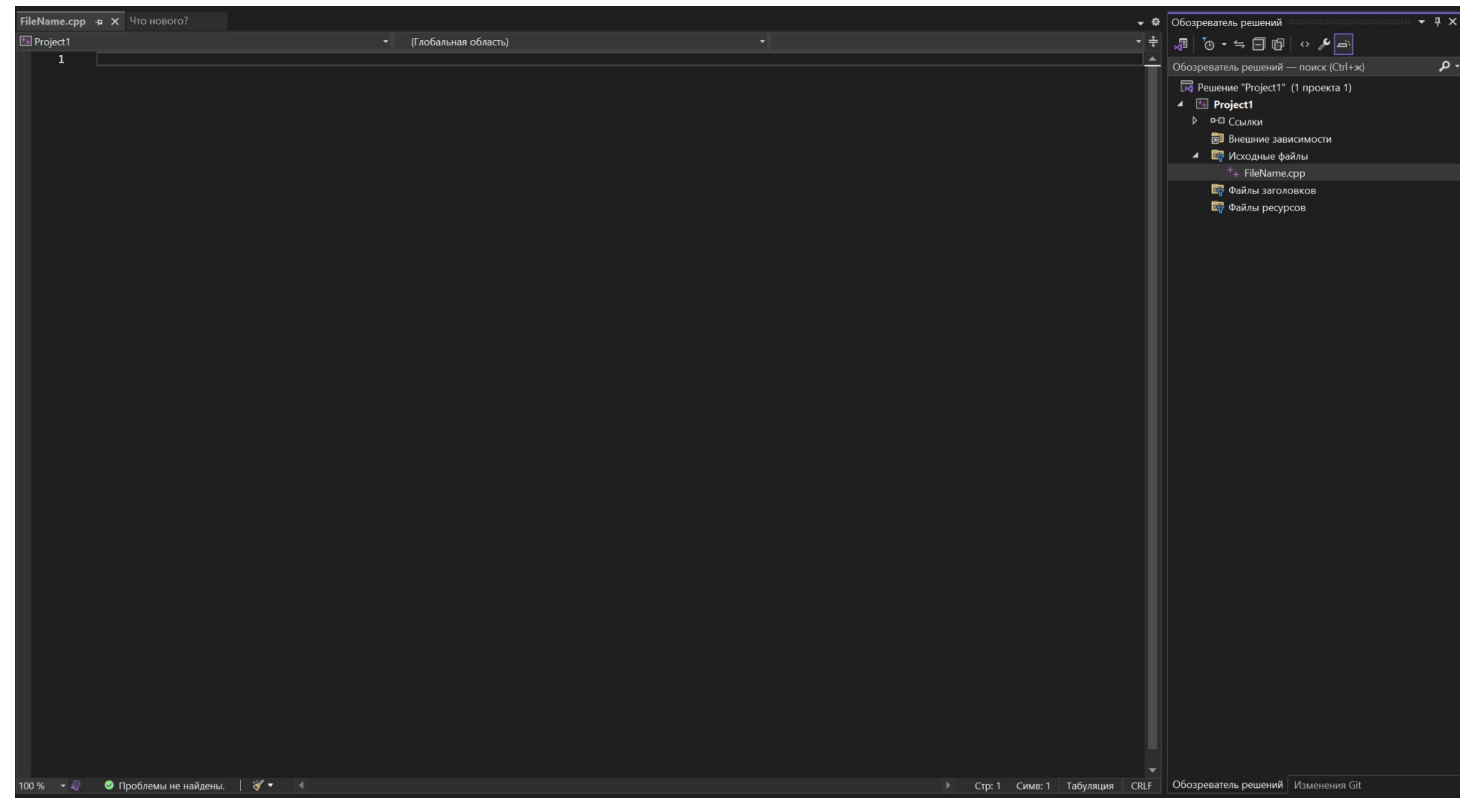
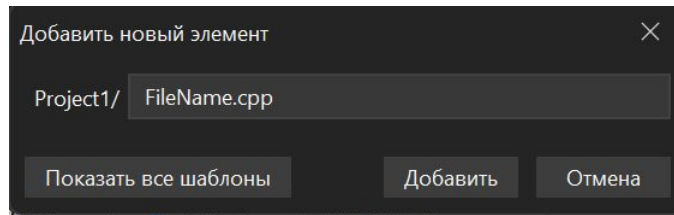
# Проект создан



# Создание файла с кодом



# Создание файла с кодом



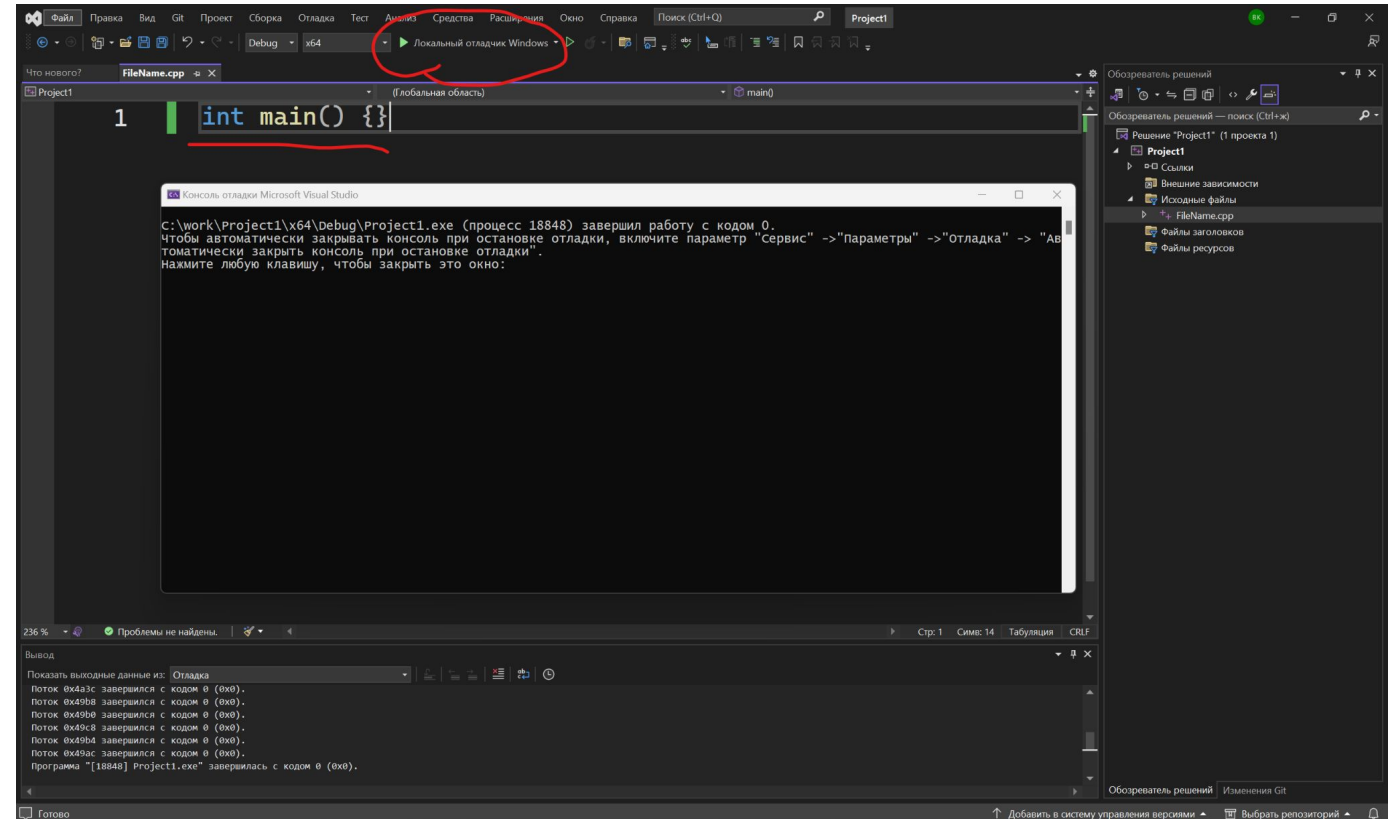
# Минимальная программа на C++

"**int**": Это тип данных, обозначающий целое число. После того как программа завершит работу она вернет числовой код (как в сообщении на скриншоте по умолчанию 0, значит всё успешно)

Функция "**main**" - это место, где программа начинает выполнение. Она всегда должна быть в программе, чтобы компьютер знал, с чего начинать.

"**()**": Тут в будущем мы сможем указать параметры для запуска, но даже если сейчас их нет, скобки нужно ставить обязательно

"**{}**": Эти фигурные скобки - это место, где вы пишете инструкции или команды для компьютера. В данном случае, скобки пустые, так что внутри них нет никаких инструкций. Программа ничего не делает, она просто заканчивается.



# Программа Hello World!

```
#include <iostream>
```

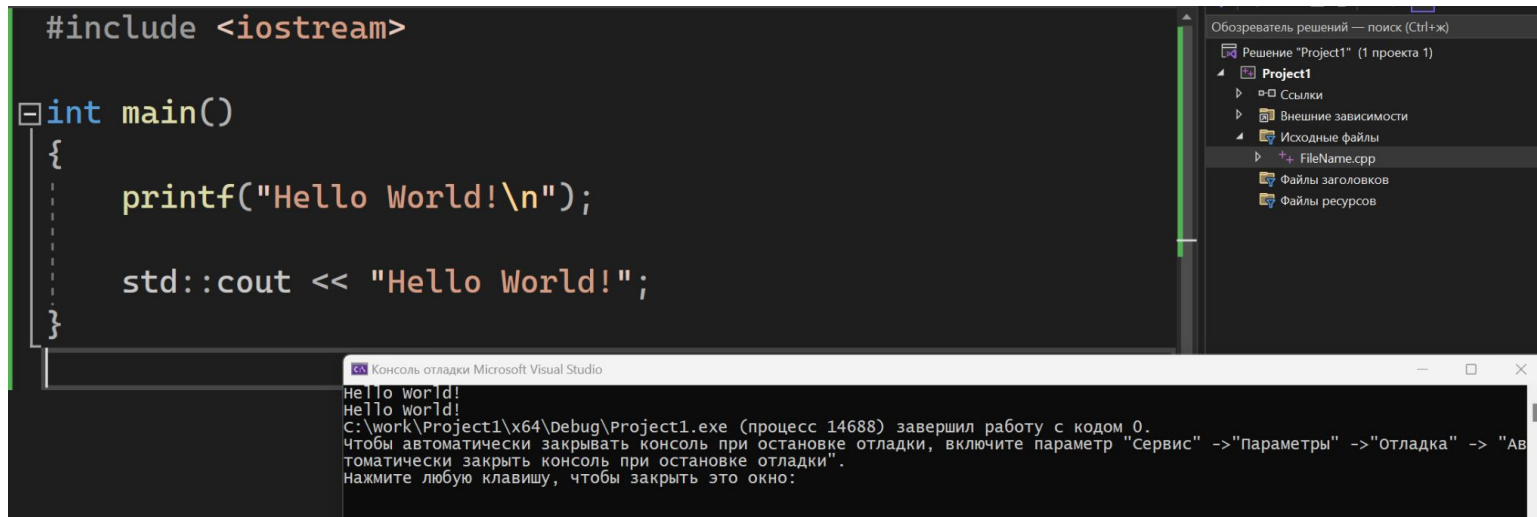
```
int main()
```

```
{
```

```
    printf("Hello World!\n");
```

```
    std::cout << "Hello World!";
```

```
}
```



The screenshot displays the Microsoft Visual Studio IDE. The main editor window shows the C++ code for a Hello World program, which includes the `<iostream>` header and uses both `printf` and `std::cout` to output "Hello World!". The code is as follows:

```
#include <iostream>

int main()
{
    printf("Hello World!\n");

    std::cout << "Hello World!";
}
```

On the right side, the "Solution Explorer" (Обозреватель решений) is visible, showing the project structure for "Project1". It includes folders for "Ссылки" (References), "Внешние зависимости" (External Dependencies), "Исходные файлы" (Source Files), and "Файлы заголовков" (Header Files). The "Исходные файлы" folder contains the file `FileName.cpp`.

At the bottom, the "Debug Console" (Консоль отладки Microsoft Visual Studio) shows the output of the program. It displays "Hello world!" twice, followed by a message indicating that the program has finished execution with code 0. The console also provides instructions on how to automatically close the console when debugging is stopped by enabling the "Service" parameter in the "Parameters" -> "Debug" -> "Automatic close console on debug stop" settings.



# Ресурсы для дальнейшего изучения

**"C++ Primer" by Stanley B. Lippman, Josée Lajoie, and Barbara E. Moo** Эта книга является одной из наилучших для начинающих. Она предоставляет хорошее введение в C++ и содержит много практических примеров.

**"Accelerated C++" by Andrew Koenig and Barbara E. Moo** Эта книга ориентирована на быстрое и интенсивное обучение C++. Она подходит для тех, кто хочет освоить язык программирования быстро.

**"C++ Primer Plus" by Stephen Prata** Эта книга отлично подходит для начинающих, так как она включает много примеров и упражнений, а также подробно объясняет ключевые концепции.

**"Beginning C++ Through Game Programming" by Michael Dawson** Если вы интересуетесь разработкой игр, эта книга предлагает увлекательный способ изучения C++ через создание игр.

**"C++ For Dummies" by Stephen R. Davis** Книга "For Dummies" хорошо известна своим простым и доступным стилем написания, что делает ее отличным выбором для новичков.

**"A Tour of C++" by Bjarne Stroustrup** Эта книга написана самим создателем C++. Она может быть полезна для тех, кто хочет понять язык изнутри. (Для новичков будет сложнее в понимании)

**"Programming: Principles and Practice Using C++" by Bjarne Stroustrup** Еще одна книга от Бьярне Страуструпа, которая рассчитана на тех, кто хочет изучить C++ как первый язык программирования.