# Lesson 2

01

How Computers Count

02

Anatomy of a Program

03

Storing Your Data

04

Code in Action

05

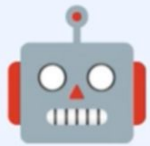
Your C++ Journey Begins

# 1

# How Computers Count

## The Language of Machines

Humans Use: Decimal (Base 10) Digits 0 through 9.



Computers Use: Binary (Base 2) Digits 0 and 1 only.

| System | Base | Representation | C++ Prefix |
| --- | --- | --- | --- |
| Decimal | 10 | 255 | `none` |
| Binary | 2 | `11111111` | `0b` |
| Hexadecimal | 16 | `FF` | `0x` |

# 2

# Anatomy of a Program

## The Basic Blueprint

# "Hello World" Anatomy

# Core Syntax Rules

- C++ is case sensitive (`main` is not the same as `Main`).
- Every statement MUST end with a semicolon (`;`).
- Comments are ignored by the compiler (`//` or `/* */`).

# 3

# Storing Your Data

## The Building Blocks

# Variable

A named location in memory used to store data.
Memory is measured in bytes, and a byte is 8 bits.

| Type | What it Stores | Memory Used |
| --- | --- | --- |
| `int` | Whole numbers | 4 bytes |
| `double` | Decimal numbers | 8 bytes |
| `char` | Single characters | 1 byte |
| `bool` | `true` or `false` | 1 byte |

# Rules for Identifiers

- Must begin with an alphabet or _
- Can contain alphabets, numbers, or _
- C++ is case-sensitive (`score` != `Score`)
- Best practice: Use meaningful names!

The `unsigned` keyword can't hold negatives and causes hard-to-find bugs by wrapping around to a large positive number.

# 4

# Code in Action

## Putting It All Together

# Arithmetic Operators

- `+` Addition
- `-` Subtraction
- `*` Multiplication
- `/` Division (Note: `5 / 2` is `2` for integers)
- `%` Modulus (gives the remainder)

`int`: Used for storing whole numbers like your age or a game score. (e.g., 75, 1000)

`double`/`float`: Used for decimal numbers like a price or your GPA. (e.g., 9.99, 3.14)

In programming, what is the result of `1 + 2 * 3`?

Use `cout` to display output to the screen and `cin` to get input from the user.

# 5

# Your C++ Journey Begins

## What's Next?

# This Week's Takeaways

- Computers think in binary, but we often use hexadecimal as a shorthand.
- All C++ programs start execution in the `main()` function.
- Always initialize your variables to prevent bugs.
- Every statement MUST end with a semicolon (`;`).

You have the building blocks. What simple problem will you solve?

How does a computer remember anything?

How do computers remember your age, GPA, or score?

Think of computer memory as a giant warehouse, filled with millions of empty, anonymous boxes.

1

# Naming the Data

## What is a Variable?

> A variable is the **name** of a location in memory where we can **store** some value.

# What is Memory?

- A collection of Bytes (8 bits)
- A Bit is the smallest unit (0 or 1)
- Each location has a long, complex address
- We need an easier way to find our data!

In C++, `int score;` tells the computer to reserve a spot in memory and label it 'score'.

# 1

# Computer's Memory

## A Giant Warehouse

2

# Choosing The Right Type

## What Kind of Data?

# Data Type

A name we give to identify the type of data we use.

| Data Type | Use Case | Memory Used |
| --- | --- | --- |
| `int` | Whole numbers | 4 bytes |
| `double` | Numbers with decimals | 8 bytes |
| `char` | A single character | 1 byte |
| `bool` | True or False values | 1 byte |

Statically Typed (C++): You must declare the variable's type BEFORE you use it.

Dynamically Typed (Python): The type is determined automatically when you assign a value.

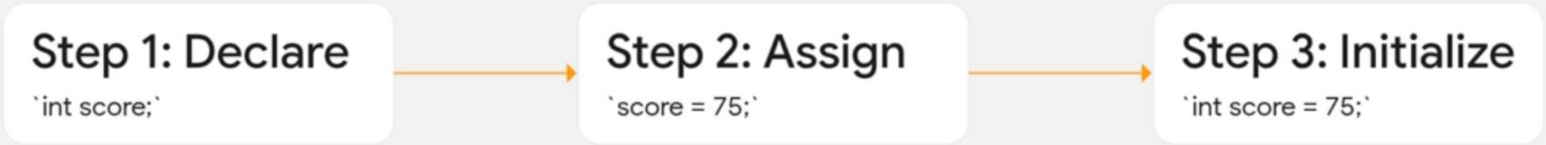
3

# Making an Assignment

## Storing Your Data

The equals sign (=) is the assignment operator. It's an action command to store data.

# The Full Process



# 4

# The Foundation of Code

## Why This Matters

This isn't just a beginner's exercise. Every complex program is built on this simple foundation.

# This Foundation Powers:

- AAA Video Games (Unreal Engine)
- Professional Software (Adobe After Effects)
- Massive Databases (MySQL, MongoDB)
- Web Browsers & Operating Systems
- Embedded Systems (Your smart toaster!)

# Key Takeaways

- Variables are the *names* for memory locations.
- Data Types are the *rules* for what can be stored.
- Assignment is the *action* of storing the data.

Now you can store data... but how do you make it DO something?