Contents

[Introduction to Variables 1](#_Toc56957069)

[Step 1: Declare a Variable 2](#_Toc56957070)

[Step 2: Initialize a Variable 2](#_Toc56957071)

[Combining Step 1 and Step 2 2](#_Toc56957072)

[Arithmetic Operators 3](#_Toc56957073)

[Chaining 4](#_Toc56957074)

[User Input 5](#_Toc56957075)

[Instructions 5](#_Toc56957076)

[Challenge: Temperature (Part 1) 5](#_Toc56957077)

[Challenge: Temperature (Part 2) 6](#_Toc56957078)

[Summary 6](#_Toc56957079)

**VARIABLES**

## Introduction to Variables

The "Hello World!" program simply writes to the screen. It does not read anything, calculate anything, or allow for user input. That’s no fun!

Real programs tend to produce results based on some input that the user of the program gives, rather than just outputting the same thing every time.

To read something from the keyboard, we first need somewhere in the computer’s memory to *store data*. That is where variables come in.

A **variable** is simply a name that represents a particular piece of your computer’s memory that has been set aside for you to store, retrieve, and use data.

In this lesson, we will learn about some of the basic data types:

* int: integer numbers
* double: floating-point numbers
* char: individual characters
* string: a sequence of characters
* bool: true/false values

Every variable has a **type**, which represents the kind of information you can store inside of it. It tells your compiler how much memory to set aside for the variable, and it defines what you can do with the variable.

## Step 1: Declare a Variable

*"Every variable in C++ must be declared before it can be used!"*

Suppose we are building a game and we want to keep track of a player’s score that goes from 0 to 10. We need a variable!

Before we can use a variable, we must *declare*, or create, it. To declare a variable, we need to provide two things:

* A type for the variable.
* A name for the variable.

So to declare an integer variable called score, we need to write:

int score;

* The int is the type of the variable.
* The score is the name of the variable.
* The ; is how we end a statement.

In C++, variable names consist only of upper/lower case letters, digits, and/or underscores.

**Note:** C++ is known as a *strongly typed* language. If you try to give an integer value a decimal number, you are going to get unexpected results, warnings, or errors.

## Step 2: Initialize a Variable

After we declare a variable, we can give it a value!

Suppose that we have declared an int variable called score, to set it to 0, we can simply write:

score = 0;

* The score is the name of the variable.
* The = indicates assignment.
* The 0 is the value you want to store inside the variable.

**Note:** In C++, a single equal sign = does not really mean “equal”. It means “assign”. In the code above, we are assigning the score variable a value of 0.

## Combining Step 1 and Step 2

We can both declare and assign a value to a variable in a single initialization statement.

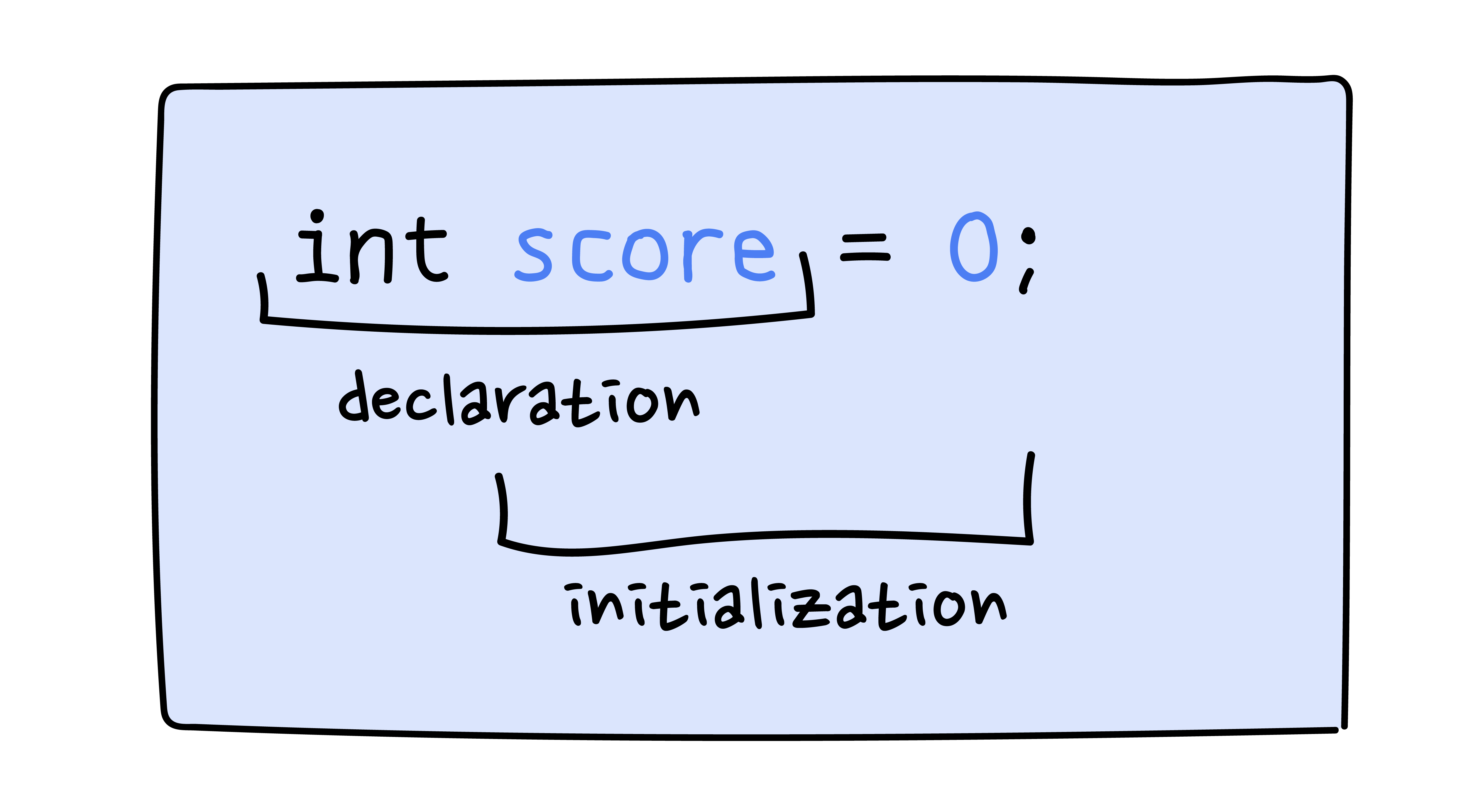
Suppose we have these two lines:

// Declare a variable  
int score;  
   
// Initialize a variable  
score = 0;

We can actually combine these two lines into a single line of code:

int score = 0;

This means we are declaring an integer called score and setting it equal to 0.



**Note:** We only need to declare a variable one time! And it is highly suggested to initialize a variable before using it later.

## Arithmetic Operators

Computers are incredible at doing calculations. Now that we have declared variables, let’s use them with *arithmetic operators* to calculate things!

Here are some arithmetic operators:

* + addition
* - subtraction
* \* multiplication
* / division
* % modulo (divides and gives the remainder)

For example:

int score = 0;  
// score is 0  
   
score = 4 + 2;  
// it is now 6  
   
score = 4 - 2;  
// it is now 2  
   
score = 4 \* 2;  
// it is now 8  
   
score = 4 / 2;  
// and now 2  
   
score = 5 % 2;  
// and now 1

**Note:** The order of operations can be specified using parentheses. For example, the use of parentheses in score = 5 \* (4 + 3) sets score equal to 5 \* 7 rather than 20 + 3.

## Chaining

Now that we have outputted a variable and have also outputted things using multiple couts. Let’s take a closer look at cout again.

If we have the code below:

int age = 28;  
   
std::cout << "Hello, I am ";  
std::cout << age;  
std::cout << " years old\n";

It will output:

Hello, I am 28 years old

Notice how we use quotes around the characters in "Hello, I am " but not in age.

* We use quotes when we want a literal string.
* We don’t use quotes when we refer to the value of something with a name (like a variable).

So now, is it possible to write the cout statements within a single line?

Yep! You can use multiple << operators to *chain* the things you want to output.

For the same code above you can also do:

int age = 28;  
   
std::cout << "Hello, I am " << age << " years old\n";

This is called **chaining**.

## User Input

Like we mentioned in the introduction, another way to assign a value to a variable is through user input. A lot of times, we want the user of the program to enter information for the program.

We have cout for output, and there is something called cin that’s used for input!

std::cout << "Enter your password: ";  
std::cin >> password;

The name cin refers to the standard input stream (pronounced “see-in”, for **c**haracter **in**put). The second operand of the >> operator (“get from”) specifies where that input goes.

To see how it works, we have to try it with a program.

### Instructions

**1.**

Add the following code:

std::cin >> tip;

So that the user of the program can enter something with their keyboard and what they enter gets saved in the int variable named tip.

## Challenge: Temperature (Part 1)

Now that we’ve learned about the basics of variables and cin, let’s write a program!

The mad scientist Kelvin has mastered predicting the weather in his mountain-side meteorology lab.

Recently, Kelvin began publishing his weather forecasts on his website, however, there’s a problem: All of his forecasts describe the temperature in Fahrenheit degrees.

Let’s convert a temperature from Fahrenheit (F) to Celsius (C).

The formula is the following:

C = (F - 32) / 1.8*C*=(*F*−32)/1.8

## Challenge: Temperature (Part 2)

Let’s go back to the **temperature.cpp** that we wrote. This time, instead of giving tempf a value of the current temperature in New York:

tempf = 83;

Let’s ask the user what the temperature is using cin!

## Summary

* A variable represents a particular piece of your computer’s memory that has been set aside for you to use to store, retrieve, and manipulate data.
* C++ basic data types include:
  + int: integers
  + double: floating-point numbers
  + char: individual characters
  + string: sequence of characters
  + bool: true/false
* Single equal sign = indicates assignment, not equality in the mathematical sense.
* cin is how to receive input from the user.