Modern C++

An effective short way

By

Mustapha Ossama Abdelhalim

2024

Contents

[1 Starter and Installation 3](#_Toc169556485)

[1.1 For windows 3](#_Toc169556486)

[2 Basics 5](#_Toc169556487)

[2.1 Introduction 5](#_Toc169556488)

[2.2 Hello World 6](#_Toc169556489)

[2.3 Types, Variables, and Arithmetic 7](#_Toc169556490)

[2.3.1 Types 7](#_Toc169556491)

# Starter and Installation

Modern C++ starts with C11, this book introduce C11 and later on, the moving to C17 section

## For windows

1. Go to [winlibs.com](https://winlibs.com/)
2. Determine which list you will choose from UCRT runtime if you are using windows 10 or 11, or choose MSVCRT runtime if you are using older versions of windows.
3. If you will use the gcc for application that runs only on windows choose MCF threads, if you are using application that runs on windows and later maybe used on Linux distribution; choose POSIX threads

I will choose Win64 in UCRT runtime in POSIX thread section as I have windows 10 x64 and have 7zip installed see Figure 1 gcc releases

Figure 1 gcc releases

See this video for more details [LINK](https://www.youtube.com/watch?v=COZw6XetvR0)

A screenshot of a computer

Description automatically generatedAfter downloading and extracting, move the mingw to c directory and get the bin path in environment variable and make sure to delete the old gcc form environment variables if exists. See Figure 2 adding bin folder path to environment variables

Figure 2 adding bin folder path to environment variables

A screen shot of a computer error

Description automatically generatedType in cmd gcc –version and you should see that gcc installed see Figure 3 verifying gcc installation4

Figure 3 verifying gcc installation

# Basics

In this chapter, the Basics of C++ will be introduced as a refresher, the following topics will be introduce:

* **First program**
  + Compilation Hello World
* **Variables and Data Types**
  + Primitive types: int, char, float, double, bool
  + Derived types: arrays, pointers, references
  + User-defined types: structs, enums, classes
* **Operators and Expressions**
  + Arithmetic operators: +, -, \*, /, %
  + Relational operators: ==, !=, >, <, >=, <=
  + Logical operators: &&, ||, !
  + Bitwise operators: &, |, ^, ~, <<, >>
  + Assignment operators: =, +=, -=, \*=, /=, %=, &=, |=, ^=, <<=, >>=
  + Increment and decrement operators: ++, --
  + Conditional operator: ?:
* **Control Structures**
  + Conditional statements: if, if-else, nested if, switch-case
  + Looping statements: for, while, do-while
  + Jump statements: break, continue, goto, return

## Introduction

A **programming language** is set of instruction to perform a task, that’s it

In this book we will use notepad++ (even the simple preinstalled notepad will work fine) and compile our program in command prompt CMD, also its completely fine to use any integrated development environment (IDE), but make sure that you are using C11 gcc version.

C++ language has two types of files headers files(.h files) and source files (.cpp files), to compile the program and make it executable for windows (aka converted to .exe files to run on windows). you will use the following command in cmd

g++ -std=c++11 name.cpp -o name.exe

let’s break it down

* **g++** is the gcc command to perform compilation
* **-std-c++11** is flag to specify the version of c11
* **name.cpp** is our source file
* **-o** is the flag for output the .exe file
* **name.exe** is the name of output

## Hello World

Lets compile our first program !

#include<iostream>

int main**(){**

std**::**cout**<<**"Hello World"**;**

**return** 0**;**

**}**

* #include<iostream>

is library that permit us to output data and take input from user

* int main(){

return 0;}

Is the entry point for our program, all programs and applications should have that function (later functions will be expressed)

* std::cout<<”hello world”;

is the command to output hello world on the screen

1. make a file named Hello.cpp for example
2. type the code above
3. open cmd in the same directory as the file Hello.cpp
4. type: g++ -std=c++11 Hello.cpp -o Hello.exe
5. to run the program type: Hello.exe

the output should be as follows in Figure 4 first programA screenshot of a computer

Description automatically generated

Figure 4 first program

## Variables and data types

C++ has types to declare each variable, each variable should have a keyword to define if it integer (like 10, 99, and120) or decimal aka float like (10.2, 0.2, and 22.8) or character (like ‘a’ , ‘b’ and ‘c’) , this declaration specify:

* + How the variable stored in memory and takes how much of program memory
  + How operation change that variable

The types in C++ is as follows in Figure 5 Types in C++ :

Figure 5 Types in C++

Primary (primitive) data types are compiler dependent that means that the data types could be stored in different sizes for different compilers, in gcc compiler:

Type the following to examine the sizes of different datatypes, for example int (integer saved in 4 bytes in gcc).

#include<iostream>

**using** **namespace** std**;**

int main**(){**

cout**<<**"char has: "**<<sizeof(**char**)<<**endl**;**

cout**<<**"wide char has: "**<<sizeof(**wchar\_t**)<<**endl**;**

cout**<<**"int has: "**<<sizeof(**int**)<<**endl**;**

cout**<<**"float has: "**<<sizeof(**float**)<<**endl**;**

cout**<<**"double has: "**<<sizeof(**double**)<<**endl**;**

cout**<<**"long has: "**<<sizeof(**long**)<<**endl**;**

cout**<<**"long double has: "**<<sizeof(**long double**)<<**endl**;**

**return 0;**

**}**

A computer screen with white text

Description automatically generatedThe output should be in gcc compiler (maybe different for other compilers) see Figure 6:

Figure 6 datatypes sizes in gcc compiler

WHY we use different types of primitive (primary) variables?

To answer this question lets examine the following table

|  |  |  |  |
| --- | --- | --- | --- |
|  | details | Memory allocation (in GCC) | Syntax |
| char | Store characters (‘a’,’b’,etc ) and integers from -128 to 127 | 1 | char x = ‘a’; |
| wchar\_t | Store much more characters than char | 2 | wchar\_t x = L’あ’ |
| int | Store integer numbers till 2^31 positive integers and 2^31 negative integers | 4 | int x = 15; |
| float | Store decimal numbers | 4 | float x = 15.12; |

Also you have some modifiers like long/short and signed and unsigned

* Short: shorten integer to be usually stored in 2 bytes instead of 4 bytes which means that the value of short int will from 2^15 positives and 2^15 negatives not 2^31 positive integers and 2^31 negative integers.
* Long: will long the integers to be usually 12 bytes instead of 4 bytes which enlarge the range of that variable
* unsigned: signed (char or int or even short int) will store all bytes in positive for example, unsigned char has range of 0-255 while signed char (or char) has -128 to 127 (2^7 positives and 2^7 negatives)

back to our question, why we have different primitive data types?  
simply if I have variable that store integer variable of human age, I want only a variable that store positive integers of range 0 yrs old -150 yrs old, so char will be chosen or even short int (aka short) no need to take 4 bytes of integer as no human ever lived 2billion years !! so it waste of memory to choose int.

remember ! char variable store integers like 15 and characters like ‘a’ not only characters

what happen if:

1. what happen if: signed short int (aka short) which have range of -32768 to 32767, store number like 32770?

ans: the variable will overflow (aka return to zero and start to count gain the reminder) which mean that 32770is higher than the capability of unsigned short (32767) by 3 so the value will be 3 like in Figure 7 Variables overflow, note: same thing to unsigned short variable the start 0 and max is 65635 so if the number exceeds; it will start counting the reminder from 0.

Remember: when you exceed the variable range; overflow will happen



Figure 7 Variables overflow

1. what happen if: storing float number like 15.02 in integer variable like

int x = 15.02 ?

Ans: the float point (.02) will be truncated i.e. s is 15 only

SO always remember which primitive data types to choose !!;

Exercises on primitive (primary) Data types:

Exercises 1: introduction

Write C++ code to introduce someone, the introduction must include:

* + Name (string): like “Ahmed” , to declare string datatype called string like:

string name;

cin>>name;

* + Age (unsigned short) like 28
  + Salary (unsigned short) like 15000
  + GPA (float) like 3.5
  + NOTE: the data should be as input from user: to get input from user use cin>>var;

Answer:

#include <iostream>

**using** **namespace** std**;**

int main**()** **{**

string name**;**

unsigned short age**,**salary**;**

float gpa**;**

cout**<<**"enter your name"**<<**endl**;**

cin**>>**name**;**

cout**<<**"enter your age and salary "**<<**endl**;**

cin**>>**age**>>**salary**;**

cout**<<**"enter your gpa"**<<**endl**;**

cin**>>**gpa**;**

cout**<<**"Introduction\nMy name is:"**<<**name**<<**endl**;**

cout**<<**"I am "**<<**age**<<**"years old "**<<**"my salary is: "**<<**salary**<<**endl**;**

cout**<<**"my GPA is: "**<<**gpa**;**

**return** 0**;**

**}**

NOTE: \n between “ “ is as same as endl after cout which means start from new line (i.e start printing at the beginning of the new following line)

NOTE: using namesapace std; is used to write cout and cin without typing std::cout and std::cin

Exercise 2: bankClient

Write C++ program to show:

* Client name: string
* ID: int
* Deposite money: float

Answer in the github repository