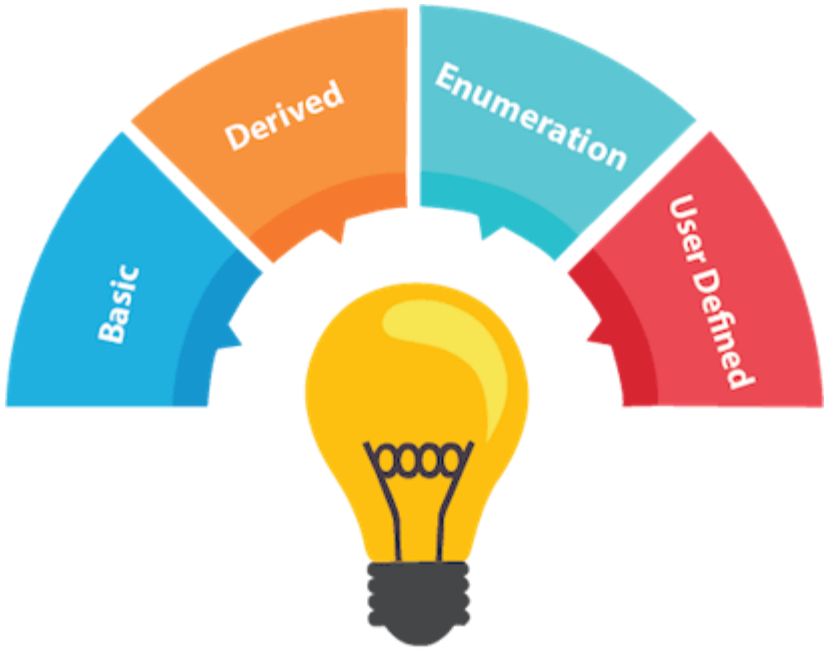
Day 2 : Variable & Datatypes in C++

+

Quick Recap of Day 1



Data Types in C++

Today’s Discussion

Quick recap

Headers

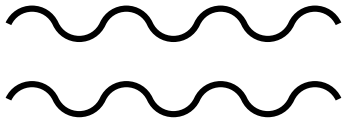
main() function

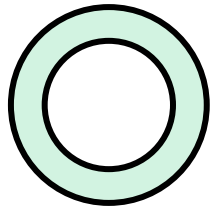
Comments

Intro to variables

Intro to datatypes

Headers





C++ code begins with the inclusion of header files. There are many header files available in the C++ programming language, which will be discussed while moving ahead with the course.

So, what are these header files?

The names of program elements such as variables, functions, classes, and so on must be declared before they can be used.

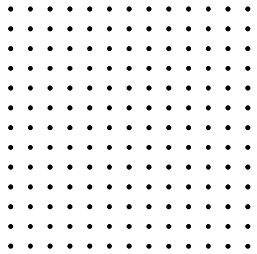
For example, you can’t just write x=100

without first declaring variable x as: int x = 100;

The declaration tells the compiler whether the element is an int, a double, a float, a function, or a class.

Similarly, header files allow us to put declarations in one location and then import them wherever we need them. This saves a lot of typing in multi-file programs. To declare a header file, we use #include directive in every .cpp file. This #include is used to ensure that they are not inserted multiple times into a single .cpp file.

Now, moving forward to the code:



#include <iostream>

using namespace std;

iostream stands for Input/Output stream, meaning this header file is necessary to take input through the user or print output to the screen.

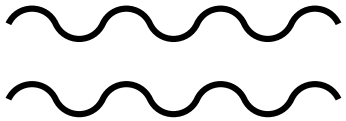
This header file contains the definitions for the functions:

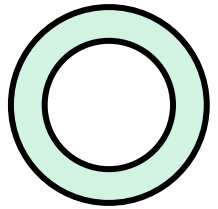
- cin: used to take input
- cout: used to print output namespace defines which input/output form is to be used.

You will understand these better as you progress in the course.

Note: semicolon (;) is used for terminating a C++ statement. i.e., different statements in a C++ program are separated by a semicolon.

main() function





Look at the following piece of code:

```
int main()
{
Statement 1;
Statement 2; ...
}
```

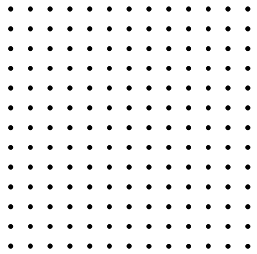
You can see the highlighted portion above. Let's discuss each part stepwise. Starting with the line: int main()

- int: This is the return type of the function. You will get this thing clear once you reach the Functions topic.
- main(): This is the portion of any C++ code inside which all the commands are written and executed.

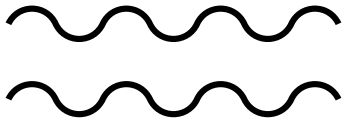
○ This is the line at which the program will begin executing. This statement is similar to the start block of flowcharts. ○ As you will move further in the course, you will get a clear glimpse of this function. Till then, just note that you will have to write all the programs inside this block.

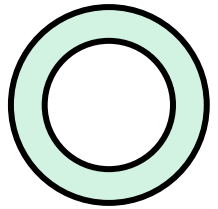
- {}: all the code written inside the curly braces is said to be in one block, also known as a particular function scope.

Again, these things will be clear when you will study functions. For now, just understand that this is the format in which we are going to write our basic C++ code. As you will move forward with the course, you will get a clear and better understanding.



C++ Comments





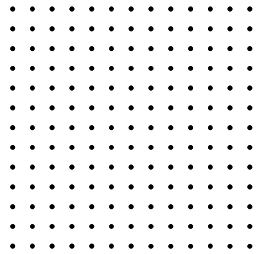
Comments can be used to explain C++ code, and to make it more readable. It can also be used to prevent execution when testing alternative code. Comments can be singled-lined or multi-lined.

Single-line Comments

Single-line comments start with two forward slashes (//).
Any text between // and the end of the line is ignored by the compiler (will not be executed).
This example uses a single-line comment before a line of code:

```
// This is a comment
cout << “Coddict C++ ";

cout << "Hii , Juniors !"; // Welcome to Coding
```



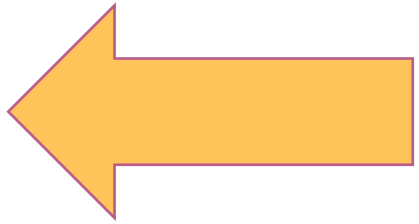
C++ Comments

C++ Multi-line Comments

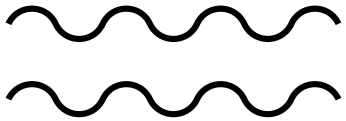
Multi-line comments start with /* and ends with */.
Any text between /* and */ will be ignored by the compiler:

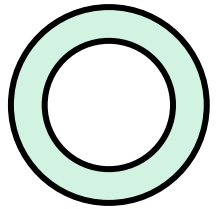
```
/* To explore the best things of this u need
Have curiosity !*/
cout << “I m learning programming ";

/*Hello Siri! Play Eminem
Latest songs*/
```



C++ Variables

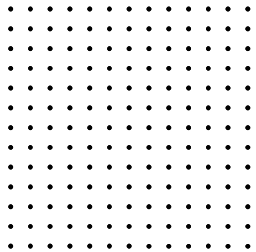




Variables are containers for storing data values.

In C++, there are different **types** of variables (defined with different keywords), for example:

- `int` - stores integers (whole numbers), without decimals, such as 123 or -123
- `double` - stores floating point numbers, with decimals, such as 19.99 or -19.99
- `char` - stores single characters, such as 'a' or 'B'. Char values are surrounded by single quotes
- `string` - stores text, such as "Hello World". String values are surrounded by double quotes
- `bool` - stores values with two states: true or false



Declaring (Creating) Variables

To create a variable, specify the type and assign it a value:

Syntax

```
type variableName = value;
```

Where *type* is one of C++ types (such as `int`), and *variableName* is the name of the variable (such as **x** or **myName**).

The **equal sign** is used to assign values to the variable.

To create a variable that should store a number, look at the following example:

```
int myNum = 15;
cout << myNum;
```

Declaring (Creating) Variables

In this way, we can only create a variable in the memory location. Currently, it doesn't have any value. We can assign the value in this variable by using two ways:

- By using variable initialization.

- By taking input Here, we can discuss only the first way, i.e., variable initialization.

Example: int x = 20;

Example:
```
int x;
cin>>x;
cout<<x;
```

Rules for defining variables in C++

- You can’t begin with a number.
  Ex- 9a can't be a variable, but a9 can be a variable.

- Spaces and special characters except for underscore(_) are not allowed.

- C++ keywords (reserved words) must not be used as a variable name.

- C++ is case-sensitive, meaning a variable with the name ‘A’ is different from a variable with the name ‘a’. (Difference in the upper-case and lower-case holds true).

Keywords in C++

- C++ keywords (reserved words) must not be used as a variable name.

asm	dynamic_cast	new	template
auto	else	operator	this
bool	enum	private	throw
break	extern	protected	true
case	false	public	try
catch	float	register	typedef
char	for	reinterpret_cast	typeid
class	friend	return	union
const	goto	short	unsigned
const_cast	if	signed	using
continue	inline	sizeof	virtual
default	int	static	void
delete	long	static_cast	volatile
do	mutable	struct	wchar_t
double	namespace	switch	while

Datatypes in C++

All variables use data-type during declaration to restrict the type of data to be stored. Therefore, we can say that data types tell the variables the type of data they can store. Pre-defined data types available in C++ are:

- int: Integer value
- unsigned int: Can store only positive integers.
- float, double: Decimal number
- char: Character values (including special characters)
- unsigned char: Character values
- bool: Boolean values (true or false)
- long: Contains integer values but with the larger size
- unsigned long: Contains large positive integers or 0
- short: Contains integer values but with smaller size

Table for datatype and its size in C++:
(This can vary from compiler to compiler and system to system depending on the version you are using)

Datatypes in C++

Examples:

int price = 5000;

f loat interestRate = 5.99f;

char myLetter = 'D';

bool isPossible = true;

string myText = "";

Data Type	Size	Description
boolean	1 byte	Stores true or false values
char	1 byte	Stores a single character/letter/number, or ASCII values
int	2 or 4 bytes	Stores whole numbers, without decimals
float	4 bytes	Stores fractional numbers, containing one or more decimals. Sufficient for storing 7 decimal digits
double	8 bytes	Stores fractional numbers, containing one or more decimals. Sufficient for storing 15 decimal digits

THANK YOU!