SLIIT ACADEMY

Higher Diploma in Information Technology
Year 1, Semester 1

Introduction to Programming(C++)

Lecture 02 : Basic Elements of C++

Intended Learning Outcomes

On the Completion of this lecture, student will be able to learn ,

LO1:Understand the how to develop and build a C++ program

LO2:Understand the basic elements of C++ program

LO3: Discuss the how to work with input and output operations in C++

LO4:Understand the define variables , constants and working with different datatypes.

SLIIT ACADEMY

Introduction to C++ Programming Language

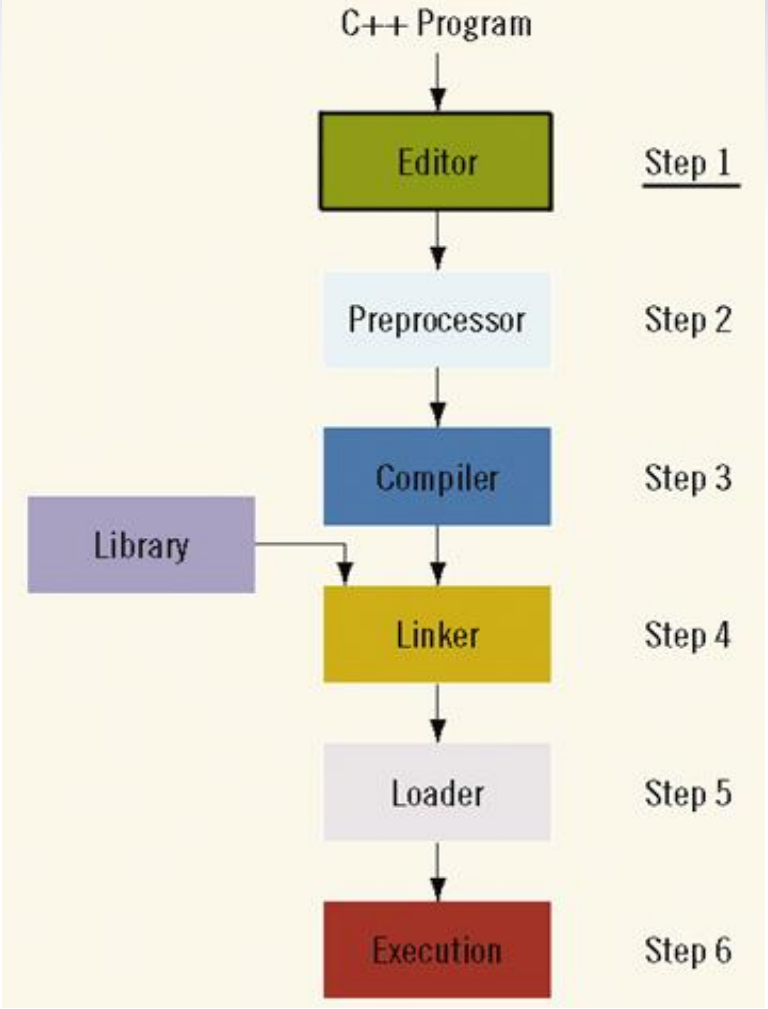
- C++ is a general-purpose programming language and widely used nowadays for competitive programming.
- It has imperative, object-oriented and generic programming features.
- C++ is portable and can be used to develop applications that can be adapted to multiple platforms.
- C++ runs on lots of platforms like Windows, Linux, Unix, Mac etc.
- C++ was developed as an enhancement of the C language.

SLIIT ACADEMY

How to Build a C++ Program

Step 1:

- Use text editor to create a C++ program.
- This program is called **source code or source program.**

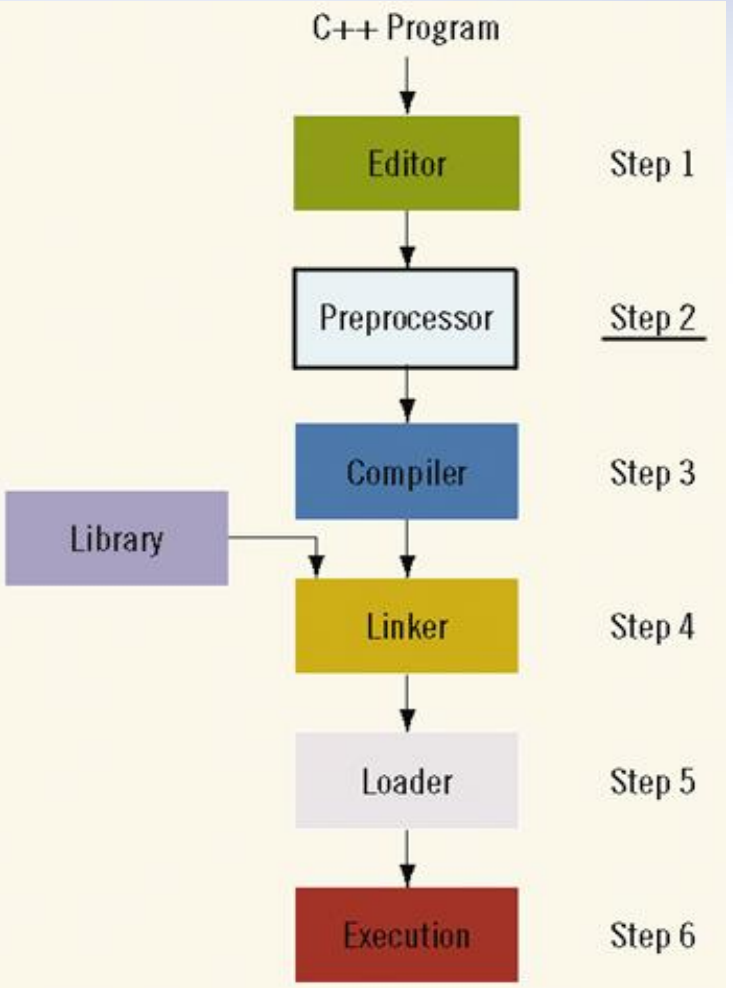


SLIIT ACADEMY

How to Build a C++ Program

Step 2:

- In a C++ program, statements that begin with the symbol # are called **preprocessor directives.**
- These statements are processed by a program called **preprocessor.**
- The preprocessors which give instructions to the compiler to preprocess the information before actual compilation starts.

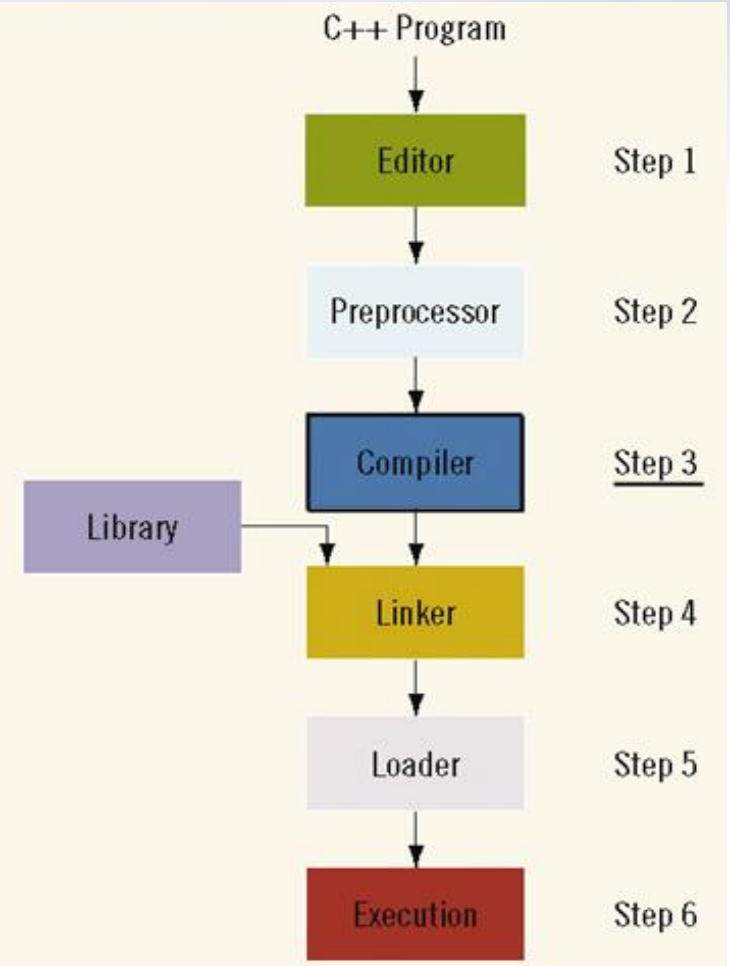


SLIIT ACADEMY

How to Build a C++ Program

Step 3:

- Compiler is used to:
  - Verifies that the program obeys the rules of the programming language and checks the source program for syntax errors.
  - Translate the program into equivalent machine language (object program).

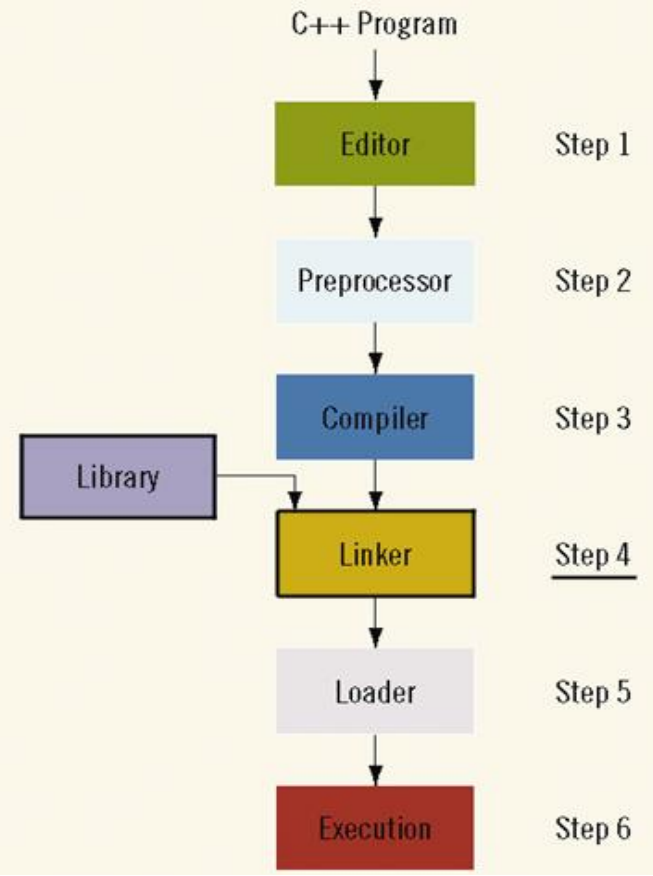


SLIIT ACADEMY

How to Build a C++ Program

Step 4:

- Programs in high level languages are developed using a **software development kit (SDK).**
- SDK contains programs that are useful in creating your program such as mathematical functions.
- The prewritten code resides in a library.



Linker combines the object code with the program from libraries.

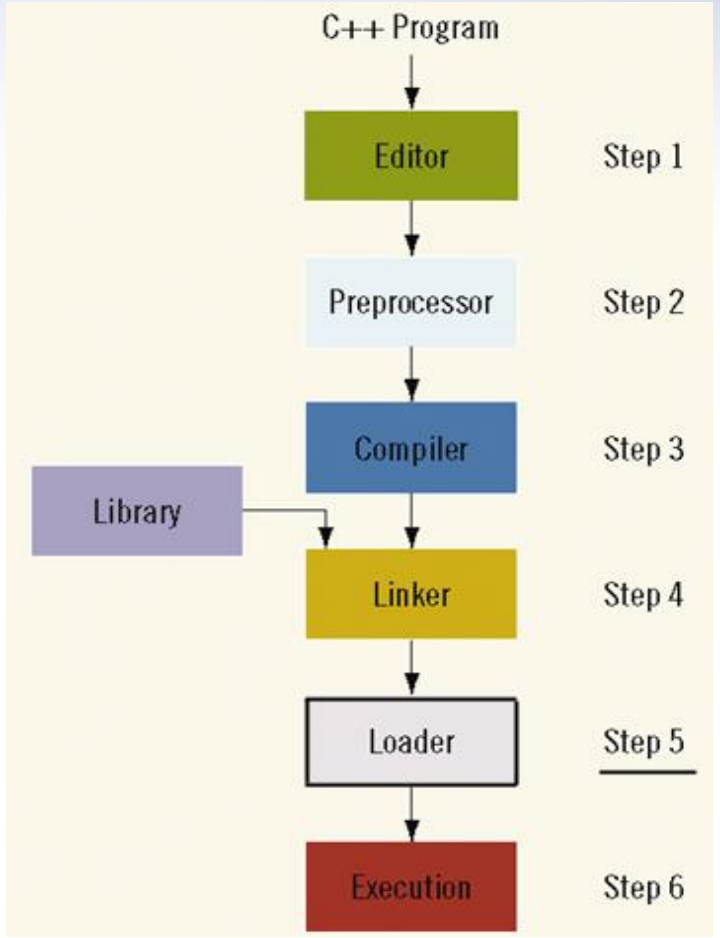
SLIIT ACADEMY

How to Build a C++ Program

Step 5:

You must load the executable program into main memory for execution.

Loader: a program that loads an executable program into main memory.

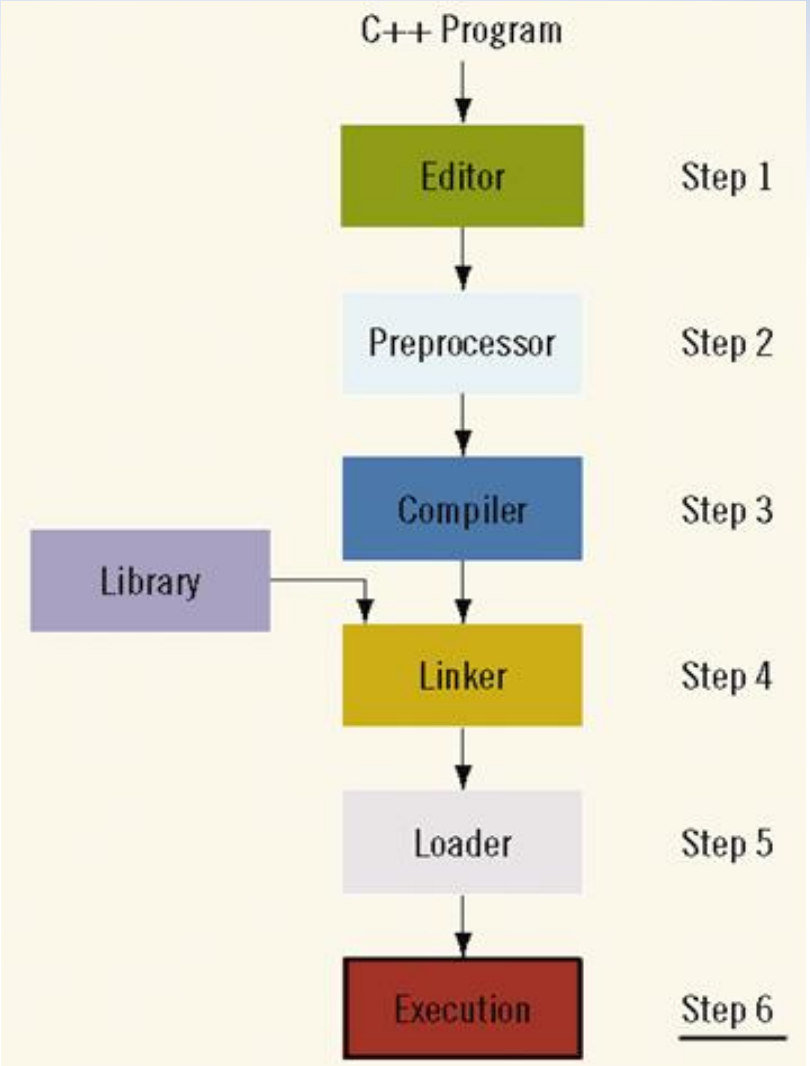


SLIIT ACADEMY

How to Build a C++ Program

Step 6:

The final step is to execute the program.



SLIIT ACADEMY

Creating a C++ Program

- A C++ program is a collection of functions, **one of which is the function main.**
- The first line of the function main is called the heading of the function:

```
int main()
```

- The statements enclosed between the curly braces ({ and }) form the **body of the function.**
- Contains two types of statements:
  - Declaration statements
  - Executable statements

SLIIT ACADEMY

Creating a C++ Program

- C++ program has two parts:
  - Preprocessor directives
  - The program
- Preprocessor directives and program statements constitute C++ source code **(.cpp).**
- Compiler generates object code **(.obj).**
- Executable code is produced and saved in a file with the file extension **.exe**

SLIIT ACADEMY

C++ Program Elements

- Comments
- Compiler directive #include
- using namespace
- main function definition
- Declarations
- Executable statements

SLIIT ACADEMY

Structure of C++ Program

```
#include <iostream>
using namespace std;
int main()
{ // prints "Hello, World!":
   cout << "Hello, World!\n";
   return 0;
}
```

Preprocessor Directives

Function named main() indicates start of the program

Comment

Statement

Ends execution of main() which ends the program

SLIIT ACADEMY

Comments

- Comments are for the reader, not the compiler. Because the comments are completely ignored by the C++ compiler.
- Comments make a program easier to understand.

Type	Explanation
Single line	`// This is a C++ program. It prints the sentence:` `// Welcome to C++ Programming.`

Multiple line	`/*`
	`  You can include comments that can occupy several lines.`
	`*/`

SLIIT ACADEMY

Compiler Directive #include

- Compiler directives are used to instruct compiler on what you want in your program.
- Preprocessor directives are commands supplied to the preprocessor.
- Many functions and symbols needed to run a C++ program are provided as collection of libraries.
- C++ provides many useful libraries (Separate files of code)
- Every library has a name and is referred to by a header file.
- Libraries are loaded using #include directive.

Compiler Directive #include

Syntax:

```
#include <filename>  //std or system header file
#include "filename"  //user defined header file
```

Example:

```
#include <iostream> //loads the iostream library containing
input/output routines
                    //(i.e. for cin and cout objects)
```

Note most compilers require that the # symbol must be at the start of the line.

using namespace

One can limit the scope of identifiers used in the program to some predefined namespace ***region*** by adding the line into the program.

```
using namespace region;
```

Example:

```
using namespace std;
```

`std` (short for standard) is a namespace region where C++ standard library is defined.

SLIIT ACADEMY

`main` Function Definition

- Every program must have a function named main.
- A C++ program begins its execution with function main.
- The main function body should end with the line **return 0;**
- which returns 0 to the operating system after the function finishes execution.

SLIIT ACADEMY

main Function Definition

Syntax

```
int main()
{
   function body //declaration statements
                 //and executable statements
}
```

Example

```
int main()
{
   cout << "Enjoy C++" << endl;
   return 0;
}
```

SLIIT ACADEMY

Declaration Statements

The declaration statements tell the compiler what data are needed in the function.

```
#include <iostream>
using namespace std;
int main()
{ // defines constants; has no output:
  const char BEEP = '\b';
  const int MAXINT = 2147483647;
  const int N = MAXINT/2;
  const float KM_PER_MI = 1.60934;
  const double PI = 3.14159265358979323846;
}
```

SLIIT ACADEMY

Input/Output Operations

- A stream is a sequence of characters associated with an input device, an output device, or a disk file.
- Class iostream defines objects:
  - cin as the stream associated with the standard input device (keyboard)
  - cout as the stream associated with the standard output device (screen)

SLIIT ACADEMY

namespace and Using cin and cout in a Program

- cin and cout are declared in the header file iostream, but within std namespace
- To use cin and cout in a program, use the following two statements:

```
#include <iostream>
using namespace std;
```

- Class iostream also defines the input operator >> and the output operator <<.

Input (Read) Statement

- cin is the standard C++ input stream (complements cout)
- cin statement reads in a value from the user & stores it in a variable..
- the stream extraction operator >> is used to read values from the stream can read multiple values from the stream using multiple instances of >>
- cin can be used with integers, floating-point numbers, characters, booleans and strings.
- cin ignores leading spaces in the input stream.

SLIIT ACADEMY

Input (Read) Statement

Syntax

```
cin >> variable_Name;
```

Example

```
cin >> age; // reads integer value into age variable
cin >> salary >> bonus; // reads real values (salary first, then bonus)
```

As a result of the cin statement, the value entered by the user is stored in the variable any future reference to that variable will access that value.

```
cout << "You entered " << age << " as your age." << endl;
```

SLIIT ACADEMY

Output Statements

- Output statements are used to display prompting messages or computational results.
- cout is the standard C++ output stream, used for writing messages.
- The stream insertion operator << is used to place messages on the stream.

Syntax

```
cout << data_elements;
```

Example

```
cout << "Enter two integers: ";
cout << "m =  " << m << ", n = " << n << endl;
```

endl (end-line constant) causes the display to advance to the next line.

SLIIT ACADEMY

Output Statements

	Statement	Output
1	`cout << 29 / 4 << endl;`	`7`
2	`cout << "Hello there." << endl;`	`Hello there.`
3	`cout << 12 << endl;`	`12`
4	`cout << "4 + 7" << endl;`	`4 + 7`
5	`cout << 4 + 7 << endl;`	`11`
6	`cout << 'A' << endl;`	`A`
7	`cout << "4 + 7 = " << 4 + 7 << endl;`	`4 + 7 = 11`
8	`cout << 2 + 3 * 5 << endl;`	`17`
9	`cout << "Hello \nthere." << endl;`	`Hello` `there.`

SLIIT ACADEMY

Output Statements

- The new line character is '\n' .May appear anywhere in the string.

```
cout << "Hello there.";
cout << "My name is James.";

Output:
   Hello there. My name is James.
```

```
cout << "Hello there.\n";
cout << "My name is James.";

Output :
    Hello there.
    My name is James.
```

	Escape Sequence	Description
\n	Newline	Cursor moves to the beginning of the next line
\t	Tab	Cursor moves to the next tab stop
\b	Backspace	Cursor moves one space to the left
\r	Return	Cursor moves to the beginning of the current line (not the next line)
\\	Backslash	Backslash is printed
\'	Single quotation	Single quotation mark is printed
\"	Double quotation	Double quotation mark is printed

SLIIT ACADEMY

Executable Statements

The executable statements cause some action to take place when a program is executed.

Example :

```
cout << "Enter two integers: ";
cin >> m >> n;
m = m + n;
cout << "m =  " << m << ", n = " << n << endl;
return 0;
```

SLIIT ACADEMY

return Statement

- return transfers control from a function back to the activator of the function (or the calling function).

Syntax

```
return expression;
```

Example

```
return 0;
return base*height/2.0;
```

- In the main function definition, return 0 transfers the control back to the operation system. With the return value of 0 which indicates to the operating system that there is no error in program execution.
- return can be without any expression if the return value is not needed.
- The *return* statement must come last inside *main*.

SLIIT ACADEMY

The Shortest C++ Program

```
int main()
{
   return 0;
}
```

No input and output - just starts and exits.

Handling Input

- Data must be loaded into main memory before it can be manipulated.
- Storing data in memory is a two-step process:
  - **Instruct computer to allocate memory**
  - **Include statements to put data into memory**

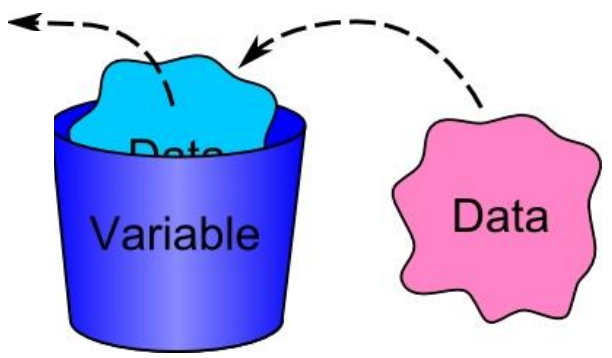
SLIIT ACADEMY

Variable & Variable Declaration

- Variable is a symbol that represents a storage location in the computer's memory.
- The information that is stored in that location is called the value of the variable.
- Variable is a memory location whose content may change during execution.
- The syntax to declare a named constant is:

```
dataType identifier, identifier, . . .;
```

```
double amountDue;
int counter;
char ch;
int x, y;
string name;
```



SLIIT ACADEMY

Methods of Variable Declaration

Consider two ways of declaring variables:

Method 1
```
int feet,inch;
double x,y;
```

Method 2
```
int a,b; double x,y;
```

Both are correct; however, the second is hard to read

SLIIT ACADEMY

Assignment Statement

- The assignment statement takes the form:

```
variable = expression;
```

- Expression is evaluated and its value is assigned to the variable on the left side
- In C++, = is called the assignment operator.

```
int num1, num2;
double sale;
char first;
string str;
```

```
num1 = 4;
num2 = 4 * 5 - 11;
sale = 0.02 * 1000;
first = 'D';
str = "It is a sunny day.";
```

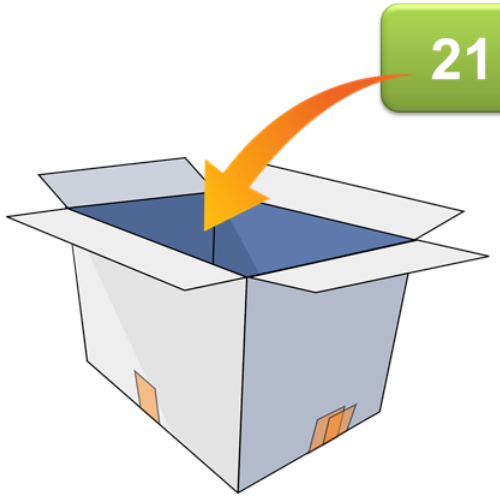
```
1.  num1 = 18;
2.  num1 = num1 + 27;
3.  num2 = num1;
4.  num3 = num2 / 5;
5.  num3 = num3 / 4;
```

SLIIT ACADEMY

Saving and Using the value of an expression

- To save the value of an expression:
  - **Declare a variable of the appropriate data type**
  - **Assign the value of the expression to the variable that was declared:**

Use the assignment statement



We can think that variable is one type of Container where we can store some element

```
int age = 21 ;
```

```
int = which type of element we can store
age = name of the container box

21 = type of element
```

Declaring & Initializing Variables

- Variables can be initialized when declared:

```
int first=13, second=10;
char ch=' ';
double x=12.6;
```

- All variables must be initialized before they are used
  - But not necessarily during declaration

SLIIT ACADEMY

Variable Initialization(Putting Data into Variables)

- There are two ways to initialize a variable:

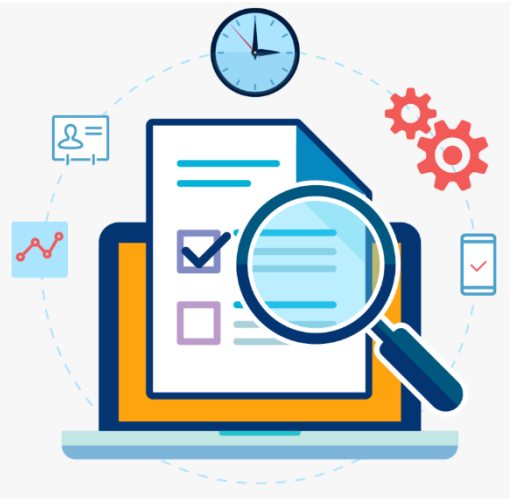
```
int feet;
```

Method 01 : By using the C++ assignment statement

```
feet = 35;
```

Method 02: By using a input(read) statement

```
cin >> feet;
```



SLIIT ACADEMY

Variable Initialization

In most cases it is wise to initialize variables where they are declared.

```
#include <iostream>
using namespace std;


int main()
{ // prints "m = ?? and n = 44":
  int m;  // BAD: m is not initialized
  int n=44;
  cout << "m = " << m << " and n = " << n << endl;
```

SLIIT ACADEMY

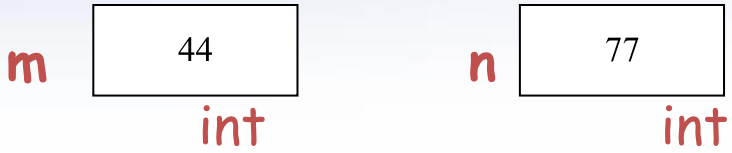
Using Integer Variable

```
#include <iostream>
using namespace std;


int main()
{ // prints "m = 44 and n = 77":
  int m, n;
  m = 44;
      // assigns the value 44 to the variable m
  cout << "m = " << m;
  n = m + 33;
      // assigns the value 77 to the variable n
  cout << " and n = " << n << endl;
}
```

View the Variable

- We can view the variable m and n like this



- m is like the address on a mailbox.
- Its value 44 is like the contents of a mailbox and
- Its type int means that the variable holds only integer values.

SLIIT ACADEMY

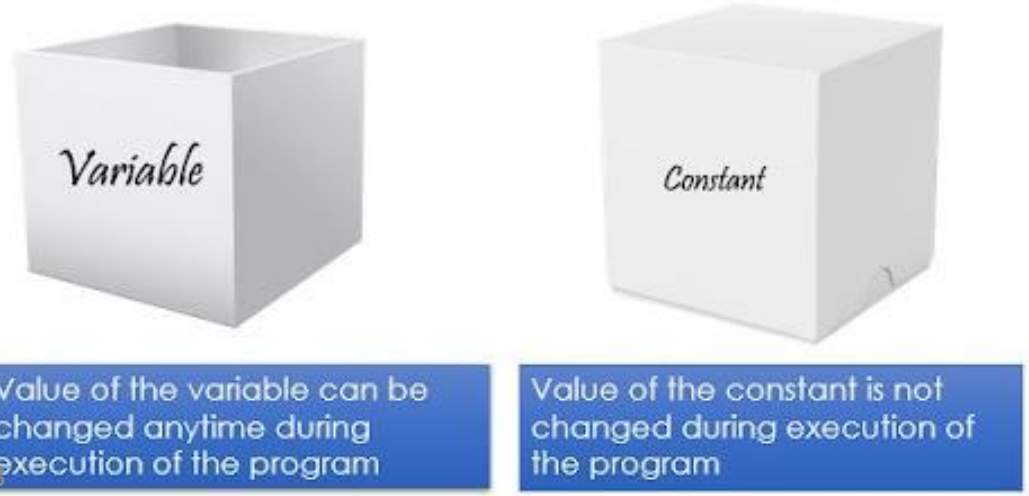
C++ Constants

- A constant is a data item with a name and a value that remain the same during the execution of the program.
- The syntax to declare a named constant is:

```
const dataType identifier = value;
```

- In C++, **`const`** is a reserved word.

```
const double CONVERSION = 2.54;
const int NO_OF_STUDENTS = 20;
const char BLANK = ' ';
const double PAY_RATE = 15.75;
```



SLIIT ACADEMY

Reserved Words and Identifiers

Reserved words (or keywords) have specific meanings that cannot be used for other purposes.

Reserved Words	Meaning
const	Declaration of a constant data item
float	Declaration of floating-point data item
include	A compiler directive
int	Declaration of integer
namespace	Region where program elements are defined
return	Causes a return from a function to the unit that activates it
using	Indicates that a program is using elements from a particular namespace

SLIIT ACADEMY

List of all C++ Reserved Words

and	default	inline	pret_cast	typename
and_eq	delete	int	return	union
asm	do	long	short	unsigned
auto	double	mutable	signed	using
bitand	dynamic_cast	namespace	sizeof	virtual
bitor	else	new	static	void
bool	enum	not	static_cast	volatile
break	explicit	not_eq	struct	wchar_t
case	export	operator	switch	while
catch	extern	or	template	xor
char	false	or_eq	this	xor_eq
class	float	private	throw	
compl	for	protected	true	
const	friend	public	try	
const_cast	goto	register	typedef	
continue	if	reinter	typeid	

SLIIT ACADEMY

Identifiers

Identifiers are names of data elements and objects manipulated by a program.

Identifiers	Meaning
cin	C++ name for standard input stream
cout	C++ name for standard output stream
m,n	Data element used for storing integers
x,y,z	Data element used for storing doubles
C1,C2,C3,C4	Data element used for storing characters
std	C++ name for the standard namespace

SLIIT ACADEMY

Rules for Selecting an Identifier

- Always begins with a letter or an underscore symbol (_)
- Consists of letters, digits, or underscores
- Cannot be a reserved word
- Some compilers limit the number of characters in an identifier to 32.
- C++ is case sensitive. **Uppercase and lowercase letters are viewed as different identifiers.**

Examples of **Invalid identifiers**:

1Letter, float, const, Two*Four, Joe's, two-dimensional

SLIIT ACADEMY

Program Style: Choosing Identifier Names

- Pick a meaningful name for an identifier.
- All capital letters are usually used for constants – KM_PER_HOUR, DOLLARS_PER_HOUR
- Avoid selecting two identifier names that are different only in their use of uppercase and lowercase letter.
- A mistyped identifiers can cause an *undefined identifier* error.
- Examples of valid identifier names without using underscore letter:

  wattHour, displayGrade, avgScore.

- Examples of valid identifiers names with the use of underscore letters:

  max_score, total_students

SLIIT ACADEMY

Practice Question 01

Provide reasons indicating why the following identifiers are invalid.

a) 2Me

b) Float

c) True

d) 2&k

e) Sub total

f) main%

g) One + two

Whitespaces

- Every C++ program contains whitespaces

  Include blanks, tabs, and newline characters

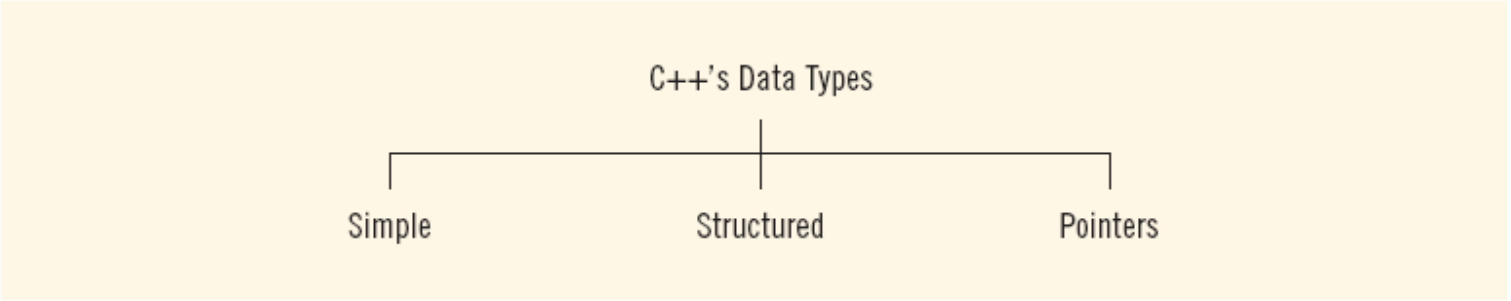
- Used to separate special symbols, reserved words, and identifiers
- The C++ compiler generally ignores whitespace, with a few minor exceptions.

```
cout<<"Hello";
cout << "Hello";
cout                    <<          "Hello"
   ;
cout
<<
"Hello";
```

SLIIT ACADEMY

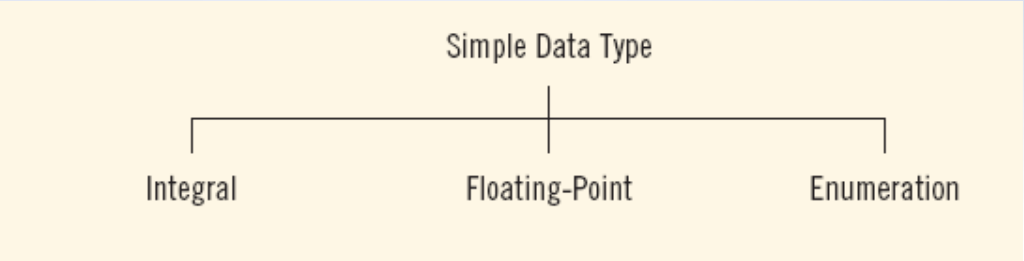
Data Types

- Data type: set of values together with a set of operations.
- C++ data types fall into three categories:



Primitive Data Types

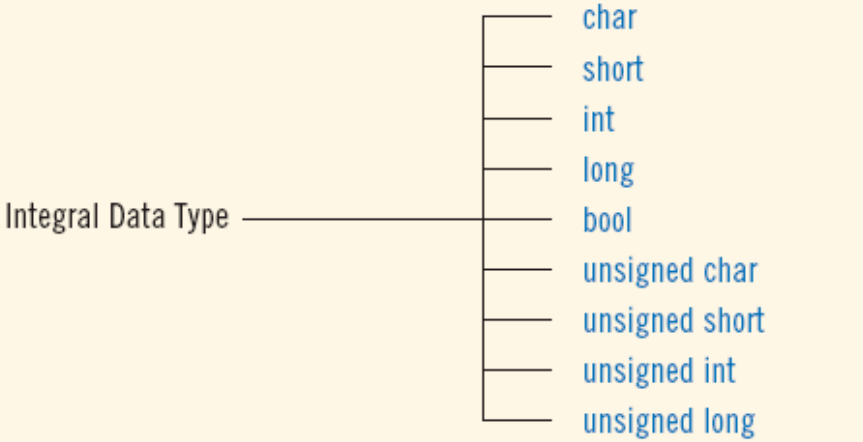
- Three categories of simple data
  - Integral: integers (numbers without a decimal)
  - Floating-point: decimal numbers
  - Enumeration type: user-defined data type



SLIIT ACADEMY

Primitive Data Types

- Integral data types are further classified into nine categories:



Data Type	Values	Storage (in bytes)
int	-2147483648 to 2147483647	4
bool	true and false	1
char	-128 to 127	1

Different compilers may allow different ranges of values.

SLIIT Academy Pvt Ltd. © 2023

Primitive Data Types

Integer Data Type

- Examples:
- -6728,0,78,+763
- Positive integers do not need a + sign.
- No commas are used within an integer.
- Commas are used for separating items in a list.

bool Data Type

- bool type
  - Two values: true and false
  - Manipulate logical (Boolean) expressions
- true and false are called logical values.
- bool, true, and false are reserved words.

SLIIT ACADEMY

Primitive Data Types

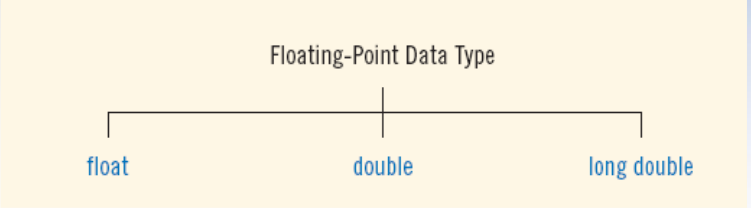
char Data Type

- The smallest integral data type
- Used for characters: letters, digits, and special symbols.
- Each character is enclosed in single quotes

```
'A','a','0','*','+','$','&'
```

- A blank space is a character and is written `' '`, with a space left between the single quotes

Floating-Point Data Types



	Float	Double
Range	3.4E+38 to 3.4E+38 (four bytes)	1.7E+308 to 1.7E+308 (eight bytes).
Maximum number of significant digits	6 or 7	15
Precision: maximum number of significant digits	single precision	double precision

SLIIT ACADEMY

C++ Strings

- A *string* type has been added for storing text.
- must include a library file that contains the string definition.

```
#include <string>
```

- Once this has been done, can declare and use variables of type string.

```
string firstName;
cout << "What is your name? ";
cin >> firstName;
cout << "Nice to meet you " << firstName << "." << endl;
```

SLIIT ACADEMY

String example

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string firstName, lastName;
    cout << "Enter your name (first then last): ";
    cin >> firstName >> lastName;

    cout << "Nice to meet you, " << firstName << " "<< lastName <<
      ".  May I just call you " << firstName << "?" << endl;

    return 0;
}
```

strings are delimited by whitespace (i.e., arbitrary sequences of spaces, tabs, and new-lines)

Summary

- Introduction to C++ Programming
- Creating C++ Program
- Structure of C++ Program
- The Basics of a C++ Program
- The Input/Output Operators
- Variable , variable declaration and Variable initialization
- Reserved Words and Identifiers
- Primitive Data types

SLIIT ACADEMY