Lecture 1

Introduction

Module 1 (Contact Hours: 12)

Introduction to OOPS Concepts,

Handling Arrays,

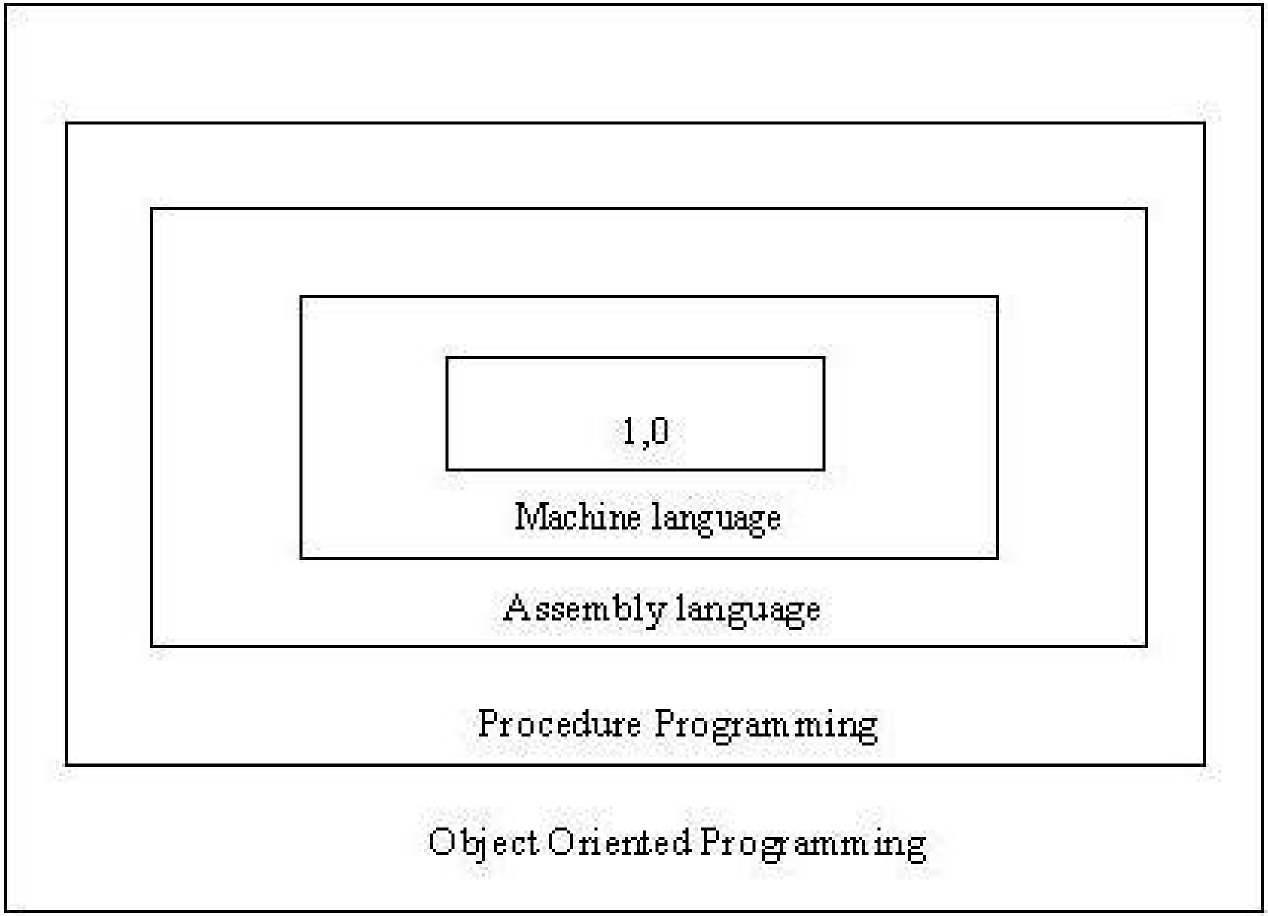
Tower of Hanoi,
Binary Search,

Time Complexity,
Asymptotic Analysis, Big –Oh notation,
Solving Recurrence relations,

Lists and Implementations: - Linked lists, Recursive functions on lists, Deletion, insertion, reversing, joining.

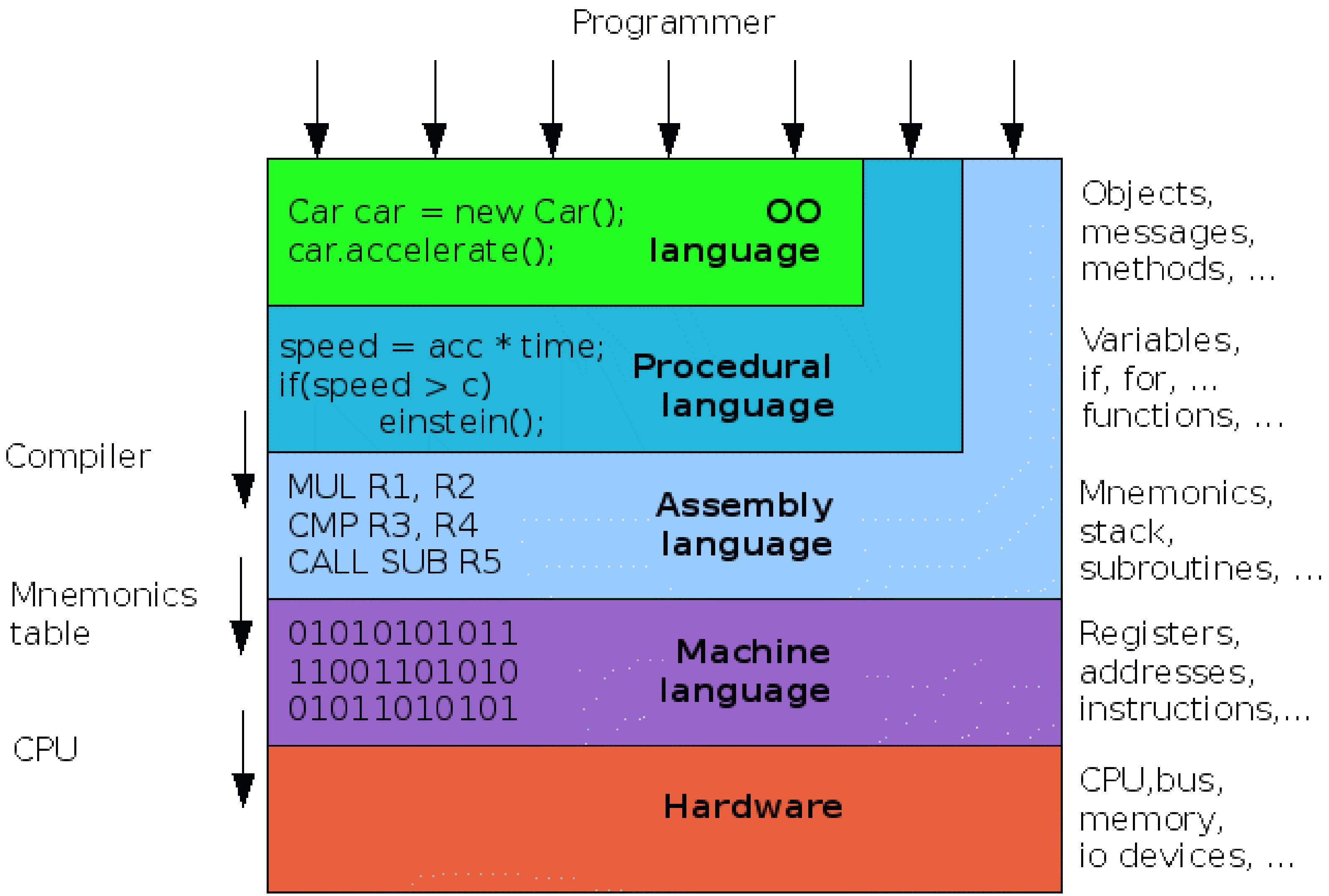
Basic Concepts of C++

Basic Concepts of C++



Layers of Computer Software

Basic Concepts of C++



Basic Concepts of C++

- **Object Oriented Programming** is method of programming where a system is considered as a collection of objects that interact together to accomplish certain tasks. Objects are entities that encapsulate data and procedures that operate on the data.
- In **OOPS** first a concept known as **"**Object Oriented Analysis **(OOA)"** is used to specify the objects in term of real world requirements, their behavior and interactions required. The next concept would be the **"**Object Oriented Design **(OOD)"** that converts these real-time requirements as a hierarchy of objects in terms of software development requirement. Finally OOPS is used to implement the requirements using the C++ programming language.

- The main purpose of object oriented programming is to simplify the design, programming and most importantly debugging a program. So to modify a particular data, it is easy to identify which function to use. To add additional features it is easy to identify where to add functions and its related data.

Basic Concepts of C++

- Object Oriented Programming Language

- Basic Concepts:
    - Classes
    - Objects
    - Data Abstraction
    - Encapsulation
    - Inheritance
    - Polymorphism
    - Dynamic Binding
    - Message Passing

** Will be discussed later in detail

Applications of OOPs Languages

- Real-time systems
- Simulation and Modeling
- Object-oriented Databases
- Hypertext and Hypermedia systems
- AI and expert systems
- Neural Networks and parallel programming
- Decision Support and office automation systems
- CIM/CAM/CAD systems

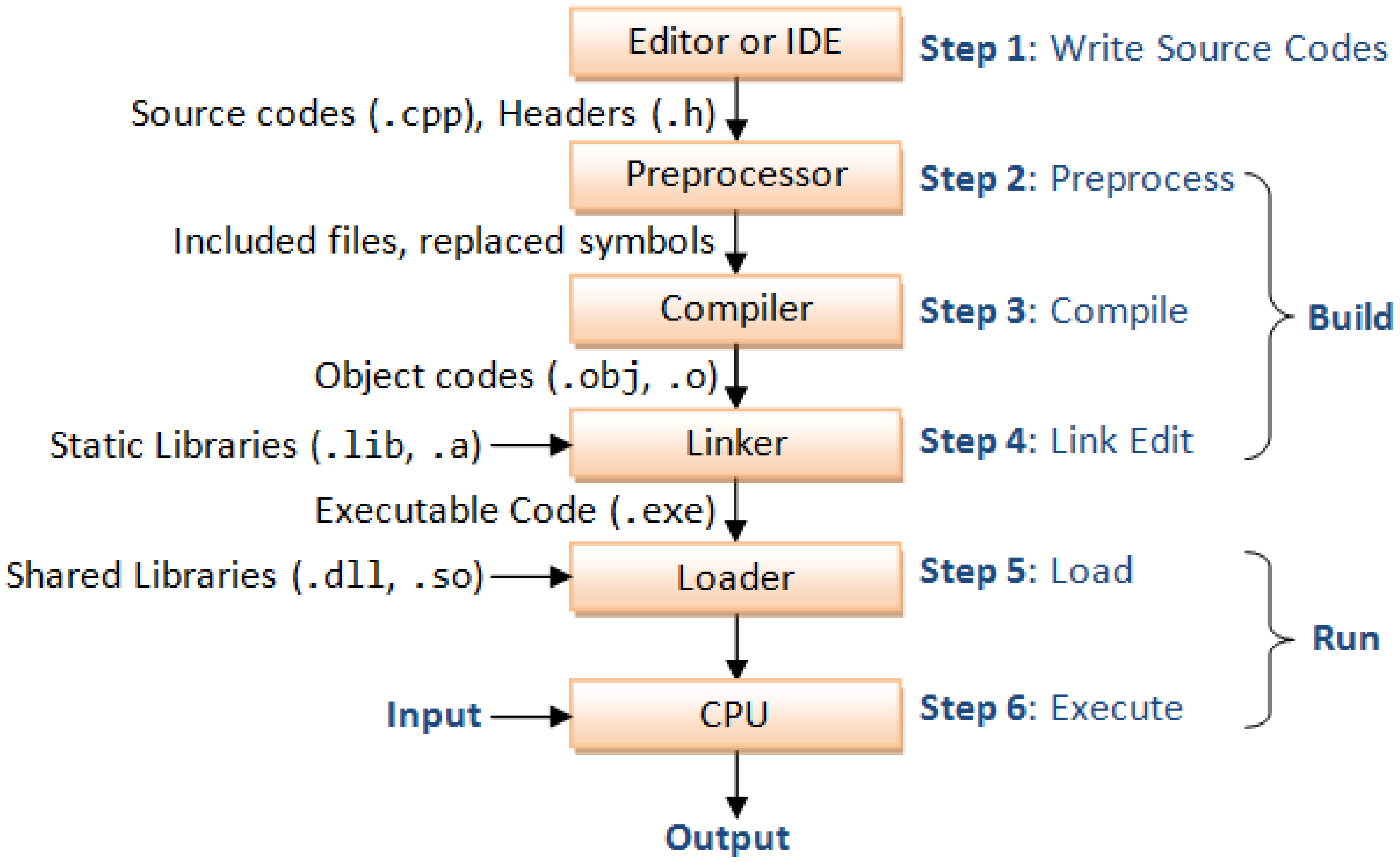
Evolution of C++

- **C (1972)**
- **Bjarne Stroustrup** adds features of the language Simula (an object-oriented language designed for carrying out simulations) to C resulting in …
- **C++ (1983)**
- **ANSI Standard C++ (1998)**
- **ANSI Standard C++ (2011)**

- **The present C++**
  - A general-purpose language that is in widespread use for systems and embedded
  - The most commonly used language for developing system software such as databases and operating systems

How a Program is Executed?

Flowchart



Editor or IDE
Step 1: Write Source Codes
Source codes (.cpp), Headers (.h)
Preprocessor
Step 2: Preprocess
Included files, replaced symbols
Compiler
Step 3: Compile
Build
Object codes (.obj, .o)
Static Libraries (.lib, .a)
Linker
Step 4: Link Edit
Executable Code (.exe)
Shared Libraries (.dll, .so)
Loader
Step 5: Load
Run
Input
CPU
Step 6: Execute
Output

1. Editor

- C++ program is written in an Editor.
- Saved as a file with extension .cpp.

2. Preprocessor

- Preprocessing performs (usually simple) operations on the source file(s) prior to compilation.

- **Typical preprocessing operations include:**

  (a) **Expanding macros** (shorthand notations for longer constructs). For example, in C,

  #define abc(x,y) (3*x+y*(2+x))

  In program n = abc(a,b) becomes

  n = (3*a+b*(2+a))

- (b) **Inserting named files**. For example, in C++,

  # include <iostream>

  is replaced by the contents of the file iostream.h

3. Compiler

- Compiler is a program that can read a program in one language — the ***source language*** — and translate it into an equivalent program in another language — the ***target language or machine language.***

- An important role of the compiler is to report any errors in the source program that it detects during the translation process.

4. Linker

- A linker **combines object code** (machine code that has not yet been linked) produced from compiling and assembling many source programs, as well as standard library functions **and resources supplied by the operating system**.

5. Loader

- Compilers, assemblers and linkers usually produce code whose memory references are made relative to an undetermined starting location that can be anywhere in memory.

  (relocatable machine code)

- The loader then puts together all of the executable object files into memory for execution.

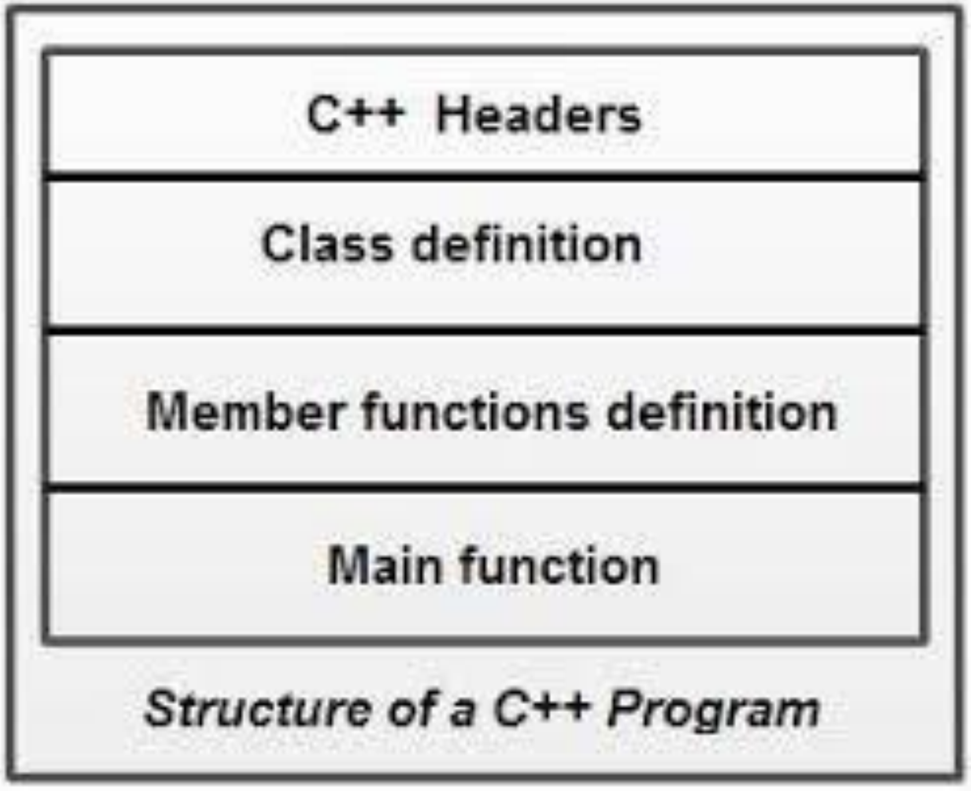
- A loader calculates appropriate absolute addresses for these memory locations and amends the code to use these addresses.

6. Execute

CPU executes the program one instruction at time.

A simple C++ Program

General Structure of C++ Program



C++ Headers

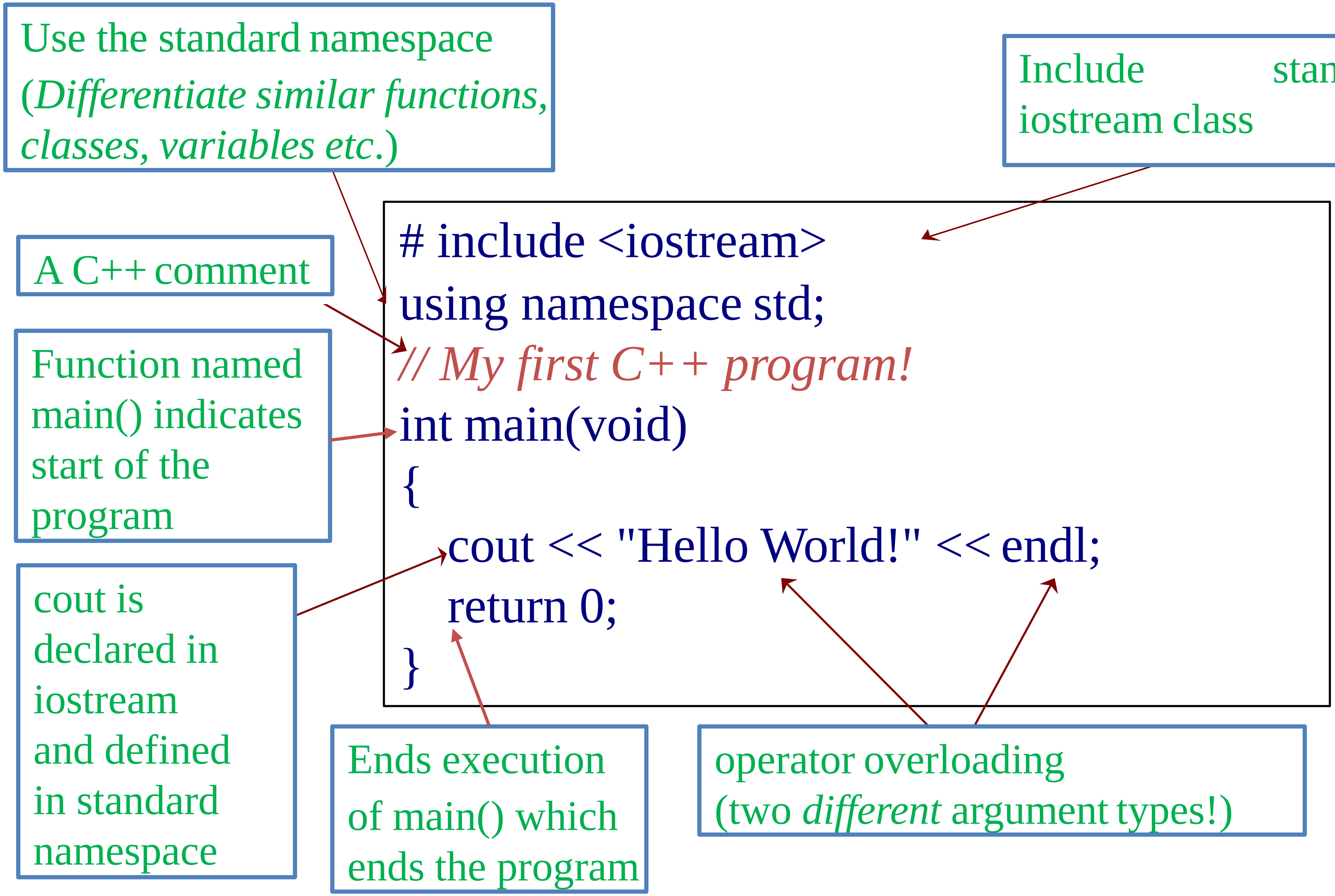
Class definition

Member functions definition

Main function

Structure of a C++ Program

“Hello World” in C++



```
# include <iostream>
using namespace std;
// My first C++ program!
int main(void)
{
   cout << "Hello World!" << endl;
   return 0;
}
```

Use the standard namespace (*Differentiate similar functions, classes, variables etc.*)

Include standard iostream class

A C++ comment

Function named main() indicates start of the program

cout is declared in iostream and defined in standard namespace

Ends execution of main() which ends the program

operator overloading (two *different* argument types!)

Common Error !!

Common error:

Omitting a semicolon as a Python Programmer

Oh No!

```
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello, World!" << endl
    return 0;
}
```

Thank You