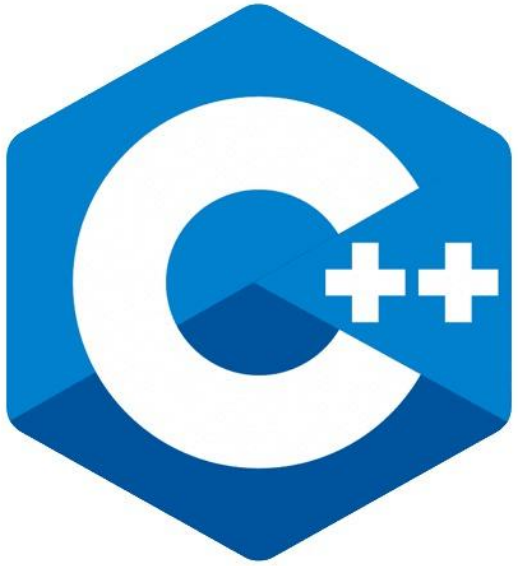


Module 1: Introduction

Instructor: Khouv Tannhuot
Phone: 087 42 47 36 (Telegram)

What is code?

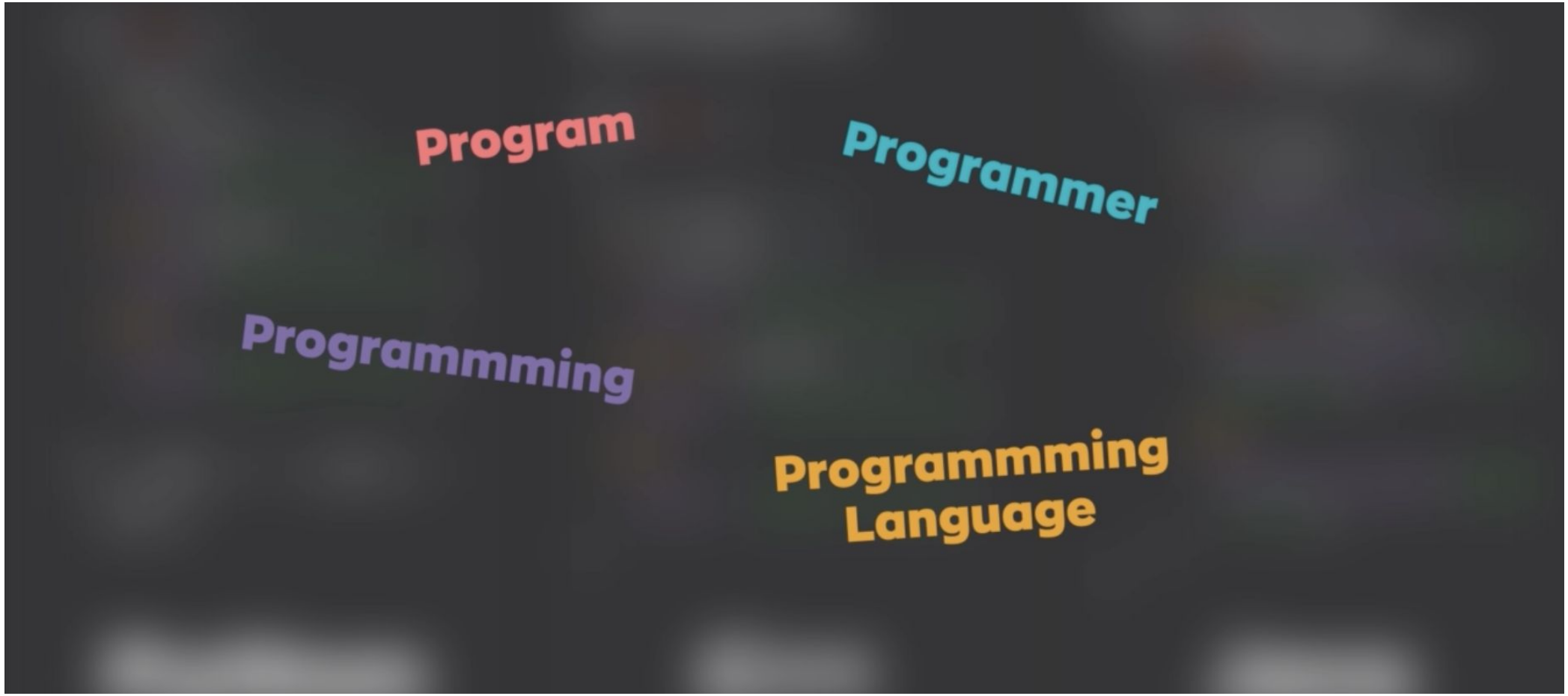
- Code is instructions a computer can understand. We can write code and tell the computer to run it, and then the computer will do whatever the code we wrote tells it to.
- For example, we can tell the computer to show some text on the screen; to display some graphics; or to perform calculations.
- Everything you see in front of you—be it on your phone, tablet, or computer—is the result of code. Somebody, somewhere, wrote the code that allows you to read this (and I don't mean the text itself, I wrote that!).

How do you write code?

- Code is written in a programming language. They are similar to human languages, but much simpler. We can't just write whatever we want though, in order for the computer to understand it, our code must follow what is accepted by the programming language we choose.

- As an example, some programming languages make extensive use of the semicolon (;). Other programming languages do not use the semicolon. How we write our code depends on which programming language we use.

Common Terms in Programming



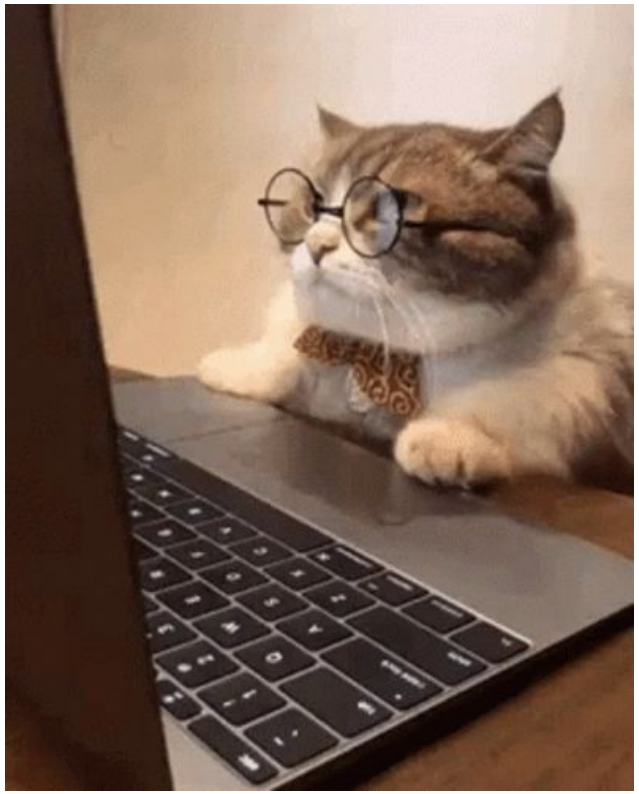
Program

```
int main(int argc, char const *argv[])
{
    int x, y;
    cin >> x >> y;
    cout << "Sum: " << x + y;

    return 0;
}
```

set of instructions

Programmer



person who writes programs

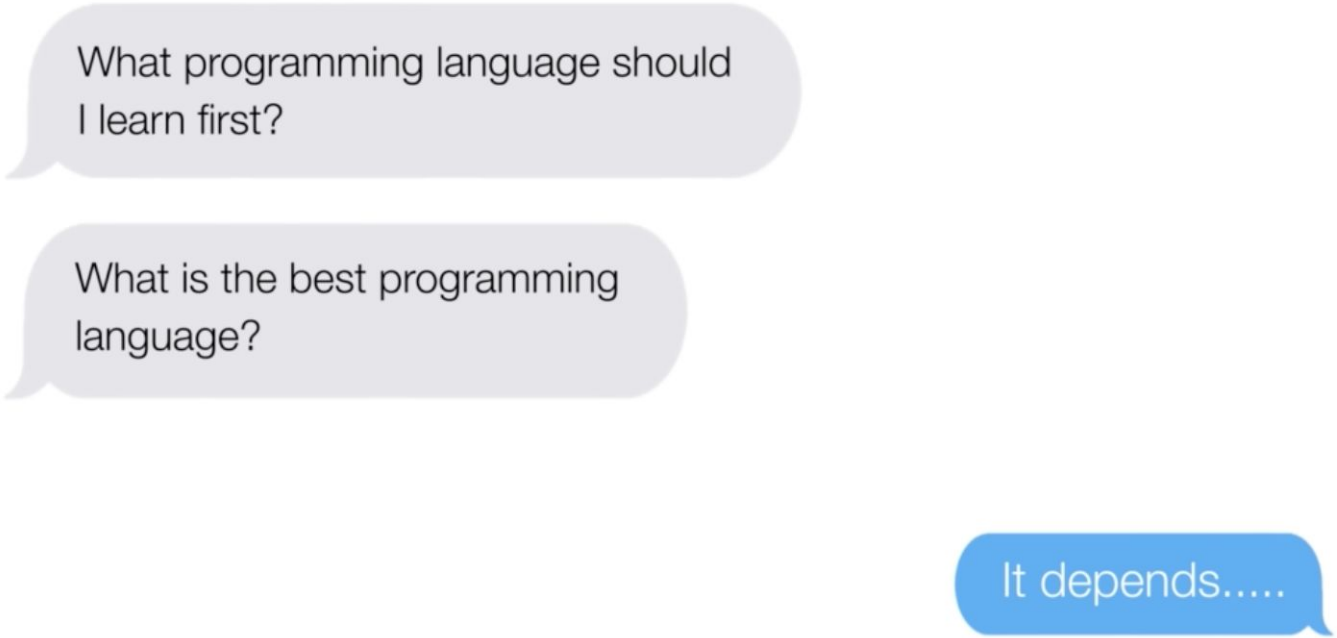
Programming

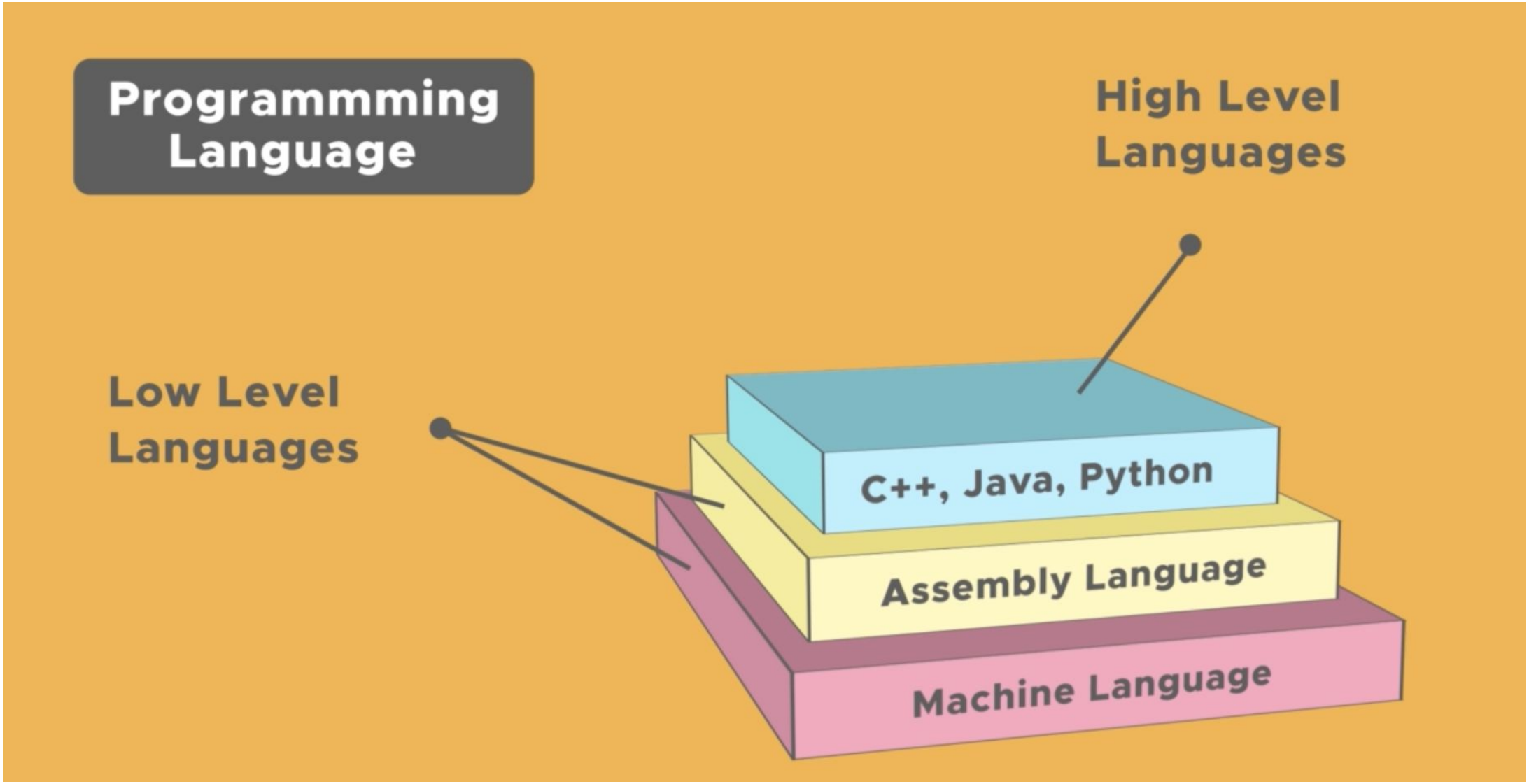
process of writing programs

Programming Language

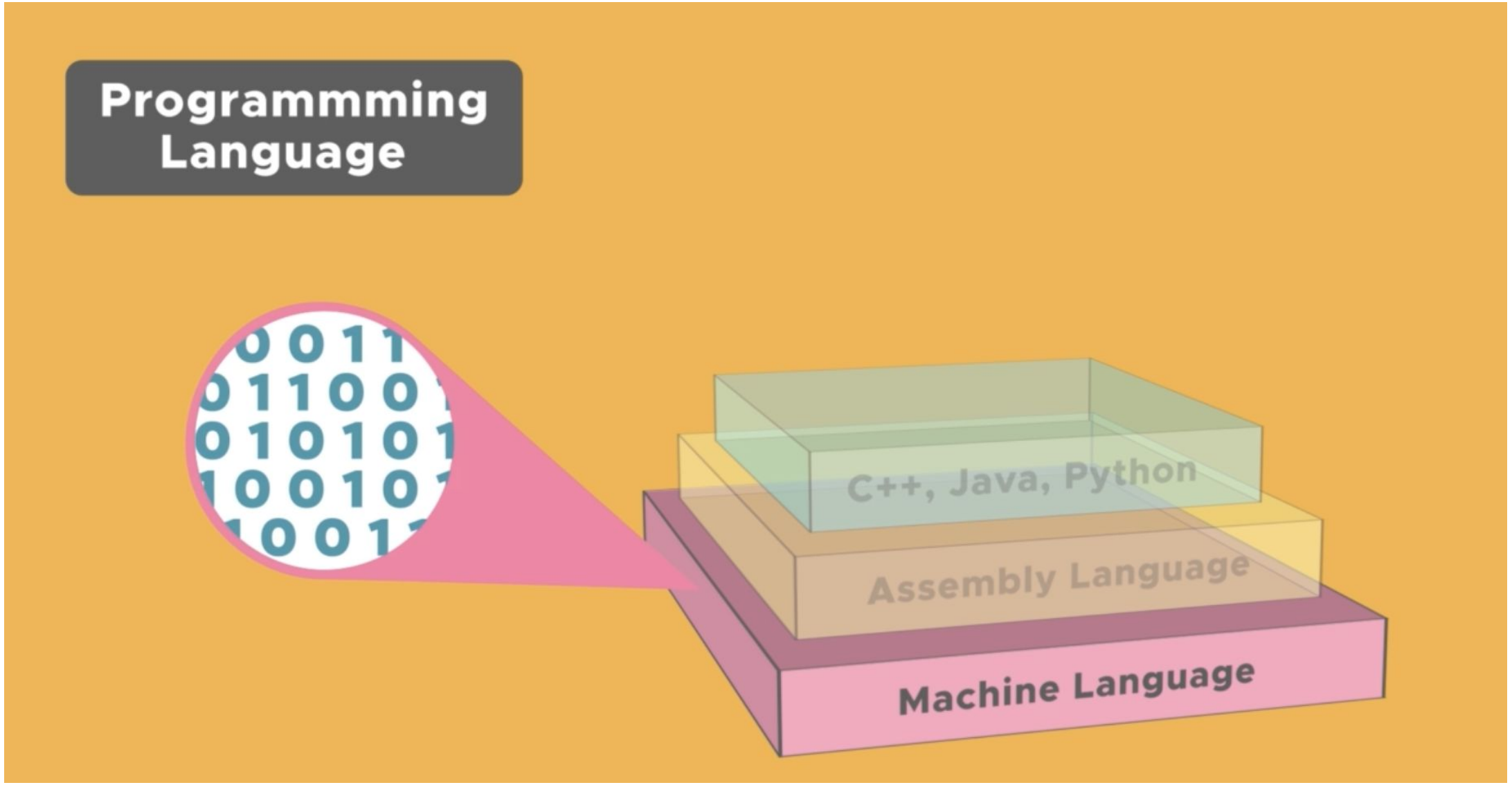
language used to write programs

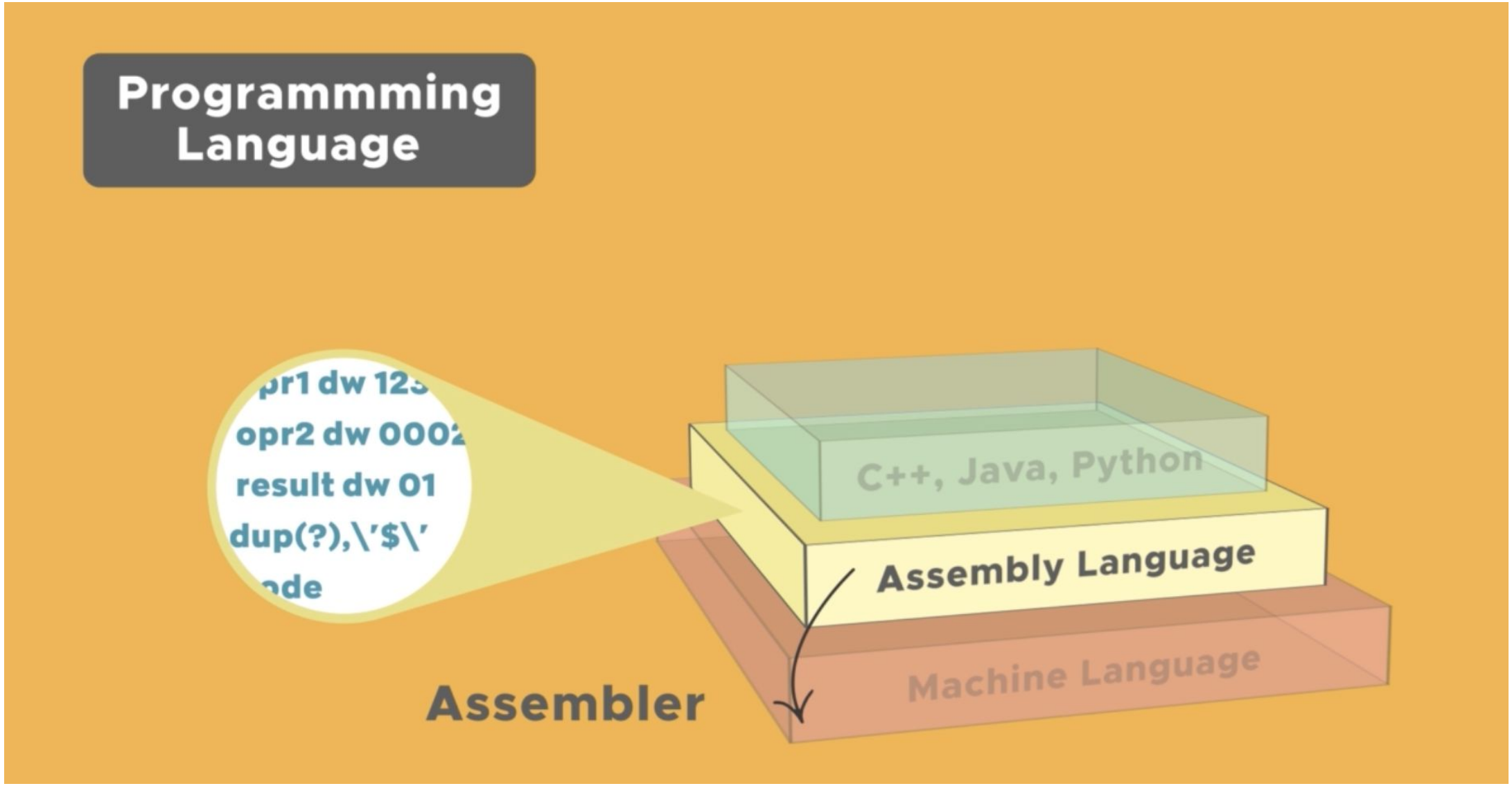
Programming Language

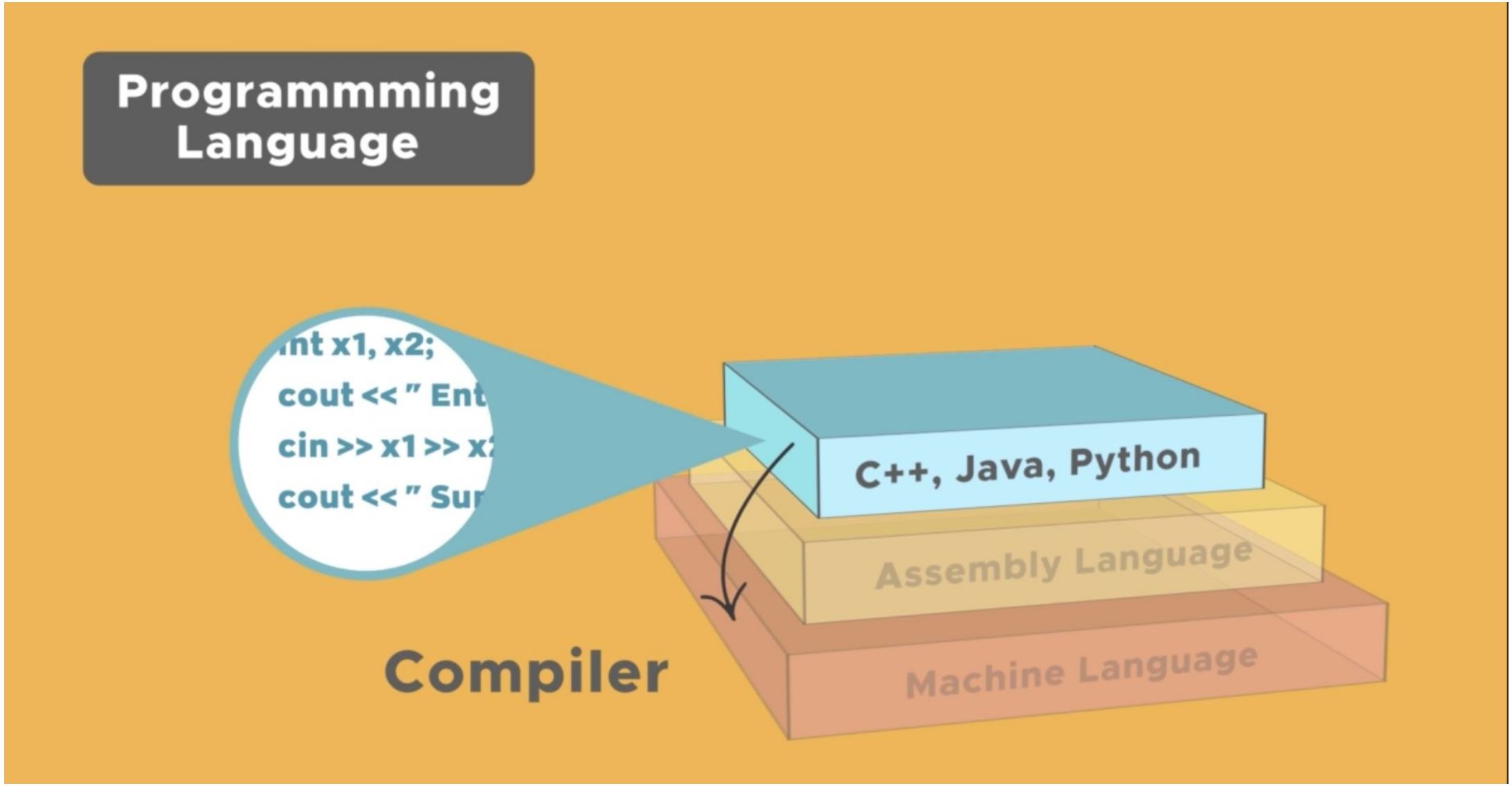



Programmming Language
High Level Languages
Low Level Languages
C++, Java, Python
Assembly Language
Machine Language

STEP IT ACADEMY

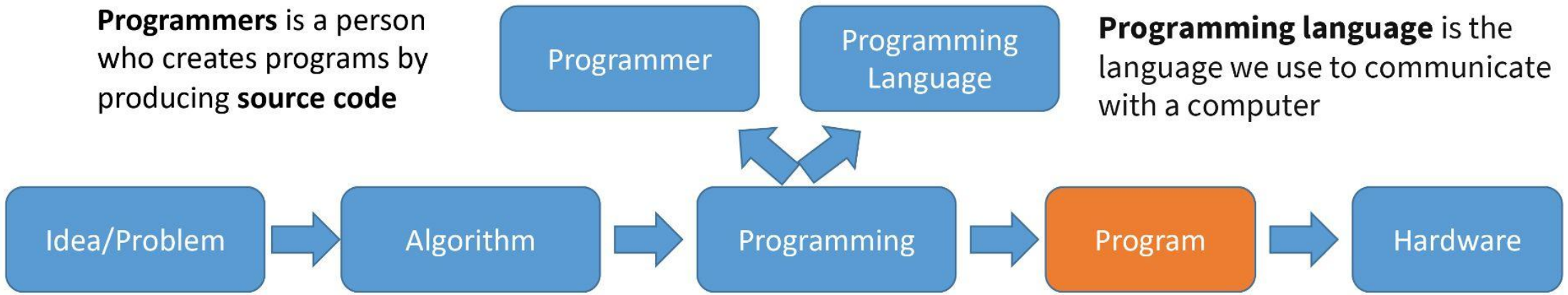



Programmming Language
pr1 dw 12
opr2 dw 0002
result dw 01
dup(?),\'$\'
ode
C++, Java, Python
Assembly Language
Machine Language
Assembler


Programmming Language
int x1, x2;
cout << " Ent
cin >> x1 >> x
cout << " Sur
C++, Java, Python
Assembly Language
Machine Language
Compiler

The Life Cycle of Computer Program

Programmers is a person who creates programs by producing **source code**



Programming language is the language we use to communicate with a computer

Idea/Problem is what we are intending to solve or achieve

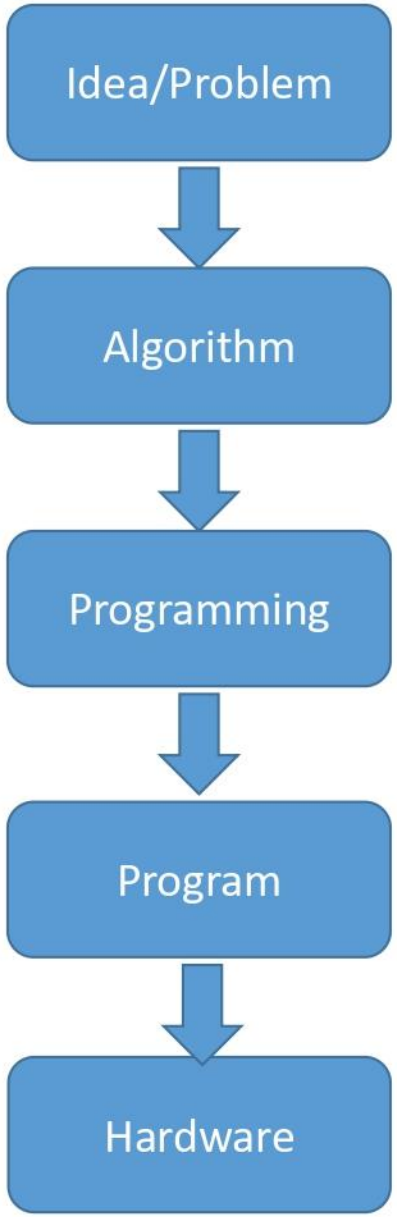
Algorithm is the well defined procedure to solve a particular problem or achieve specific task.

Programming is the process of creating a program.

Program is a set of instructions that tell computer to perform specific task.

Computer is a machine which can perform many computational tasks.

Here is a sample of the life cycle of C++ Programming

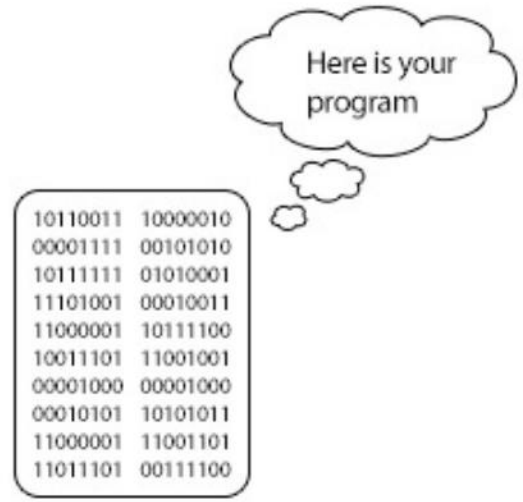


"I want to write a program that will allow users to enter two numbers, then adding the two numbers."

STEP 1 : Allocate memory for storing three numbers
STEP 2 : Store first number in computer memory
STEP 3 : Store second number in computer memory
STEP 4 : Add these two numbers together and store the result of the addition in a third memory location
STEP 5 : Print the result

add.cpp: cpp source code file

```
#include <iostream>
using namespace std;
int main() {
        int x; int y; int sum;
        x = 25;
        y = 10;
        sum = x + y;
        cout << "The sum is ";
        cout << sum;
        return 0;
}
```



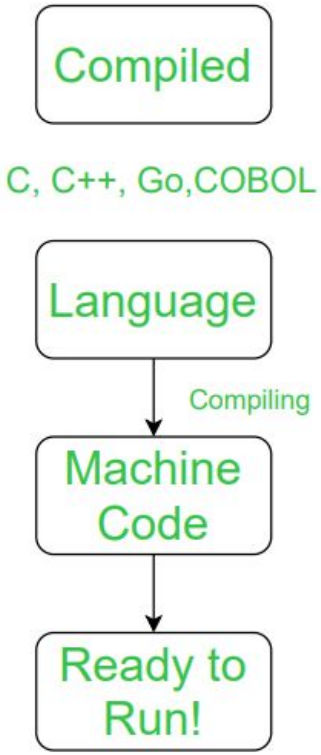
add.exe: cpp executable file

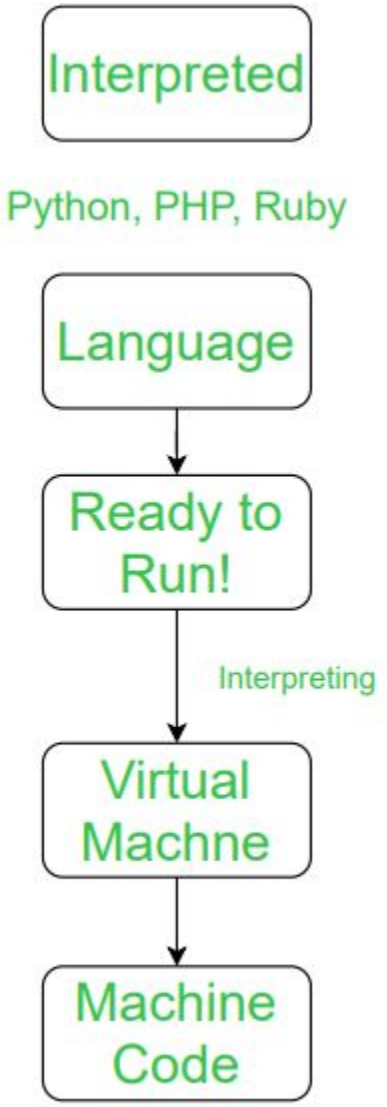
What is C++?

- General Purpose Programming Language
- Middle Level Programming Language
- Compiled Programming Language
- can write many types of programming
  - Procedural Programming
  - OOP (Object Oriented Programming)
  - Generic Programming

Difference between Compiled and Interpreted Language

S.NO.	Compiled Language	Interpreted Language
1	Compiled language follows at least two levels to get from source code to execution.	Interpreted language follows one step to get from source code to execution.
2	A compiled language is converted into machine code so that the processor can execute it.	An interpreted language is a language in which the implementations execute instructions directly without earlier compiling a program into machine language.
4	The compiled programs run faster than interpreted programs.	The interpreted programs run slower than the compiled program.
5	In a compiled language, the code can be executed by the CPU.	In Interpreted languages, the program cannot be compiled, it is interpreted.
6	This language delivers better performance.	This language delivers slower performance.





A bit history of C++

- C++ (pronounced see plus plus) was developed by Bjarne Stroustrup at Bell Labs as an extension to C, starting in 1979.
- C++ adds many new features to the C language
- Three major updates to the C++ language (C++11, C++14, and C++17, ratified in 2011, 2014, and 2017 accordingly) have been made since then, each adding additional functionality

Why C++?

- Efficient language
- Reliable and fast
- Better understanding of Procedural and Object Oriented Programming Approach
- Easy to learn other programming language Java or C#

```python
def main():
  counter = 4
  if (counter > 2):
    print("More than two!")
  elif (counter < 2):
    print("Less than two!")
  else:
    print("Equal to two!")

if __name__ == '__main__':
  main()
```

Python

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

void main ()
{
  int counter = 4;
  if (counter > 2)
    cout << "More than two!";
  else if (counter < 2)
    cout << "Less than two!";
  else
    cout << "Equal to two!";
}
```

C++

```java
public class Driver {
  public static void
      main(String[] args)
    {
      int counter = 4;
      if (counter > 2)
        System.out.print("More
        than two");
      else if (counter < 2)
        System.out.print("Less
        than two");
      else
        System.out.print("Equal
        to two");
    }
}
```

Java

Where we use C++?

Here are a few common types of applications that most likely would be written in C++:

- Video games
- Real-time systems (e.g. for transportation, manufacturing, etc...)
- High-performance financial applications (e.g. high frequency trading)
- GUI Applications and simulations
- Productivity / office applications
- Embedded software
- Audio and video processing

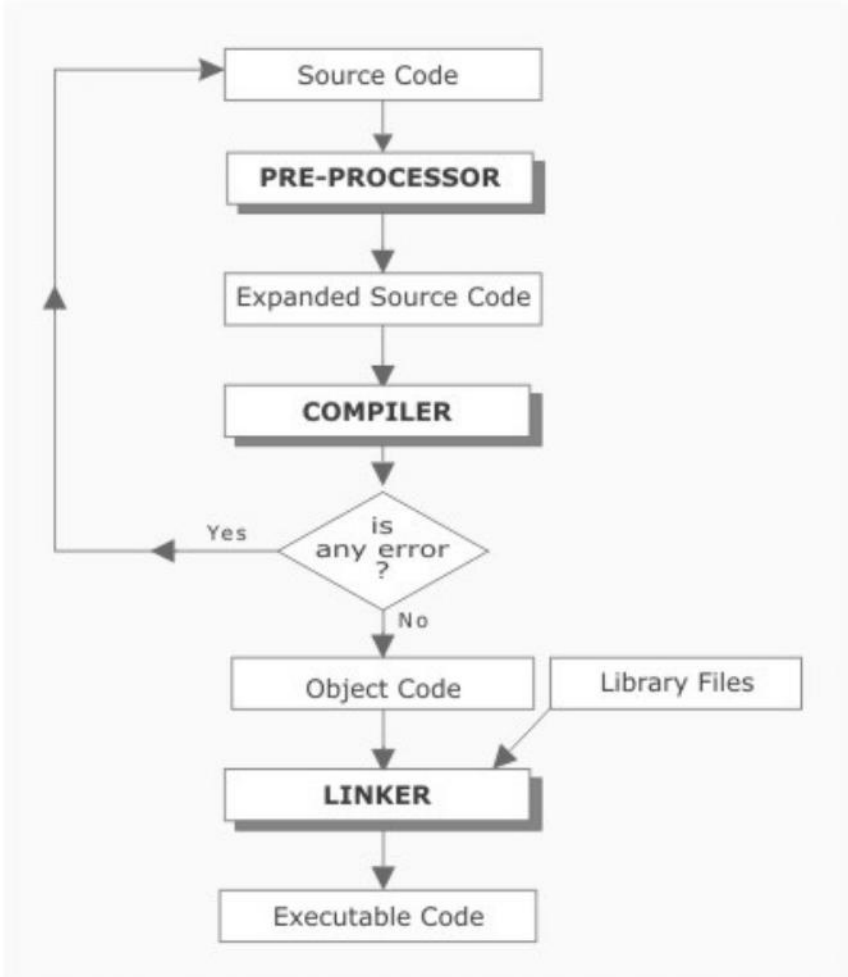
Introduction to C++ Development

Hello.cpp

```
//this is demo
#include<iostream>

Int main(){
    std::cout<<”Hello”;
    return 0;
}
```

```
10110011 10000010
00001111 00101010
10111111 01010001
11101001 00010011
11000001 10111100
10011101 11001001
00001000 00001000
00010101 10101011
11000001 11001101
11011101 00111100
```



Code of iostream file

```
int main(){
    std::cout<<”Hello”;
    return 0;
}
```

Basic Creation and Execution of a C++ program

- Create source code with a text editor, store to disk
  - Source code is just a plain text file, usually given a filename extension to identify the programming language (like .c for C, or .cpp for C++)
- Preprocessor -- Part of compiler process, performs any pre-processing tasks on source code
- Compilation -- syntax checking, creation of object code
  - Object code is the machine code translation of the source code
- Linking -- Final stage of the creation of an executable program. Linking of object code files together with any necessary libraries (also already compiled)
- Execution of program
  - Program loaded into memory, usually RAM
  - CPU executes code instructions

C++ Development ToolSet and IDE

Toolset

- TextEditor
- Pre Processor
- Compiler
- Tester/Debugger
- Linker

IDE: Integrated Development Environments

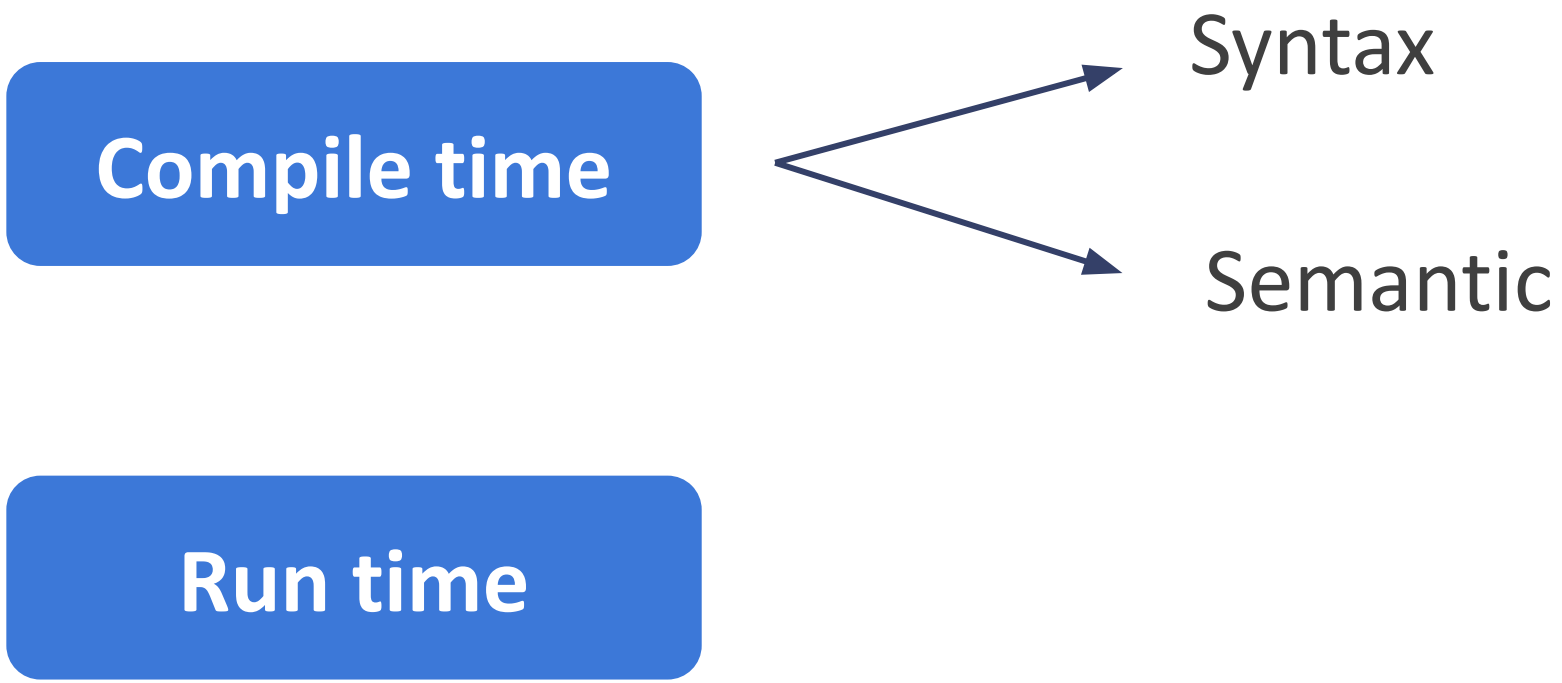
- An Integrated Development Environment (IDE) is a software package that includes all necessary tools for creating a program. Typcially includes:
  - Text editor
  - Compiler
  - Linker
  - Debugger
  - ability to launch and run applications from within IDE environment
- Not every compiler is an IDE, but pretty much any compiler software includes: preprocessor, compiler, linker
- Examples of C++ IDEs
  - Microsoft Visual C++ 6.0
  - Metrowerks CodeWarrior
  - Borland Builder

IDE

Programming is about Problem Solving

- Algorithm - a finite sequence of steps to perform a specific task
  - To solve a problem, you have to come up with the necessary step-by-step process before you can code it
  - This is often the trickiest part of programming
- Testing - algorithms must also be tested!
  - Does it do what is required?
  - Does it handle all possible situations?

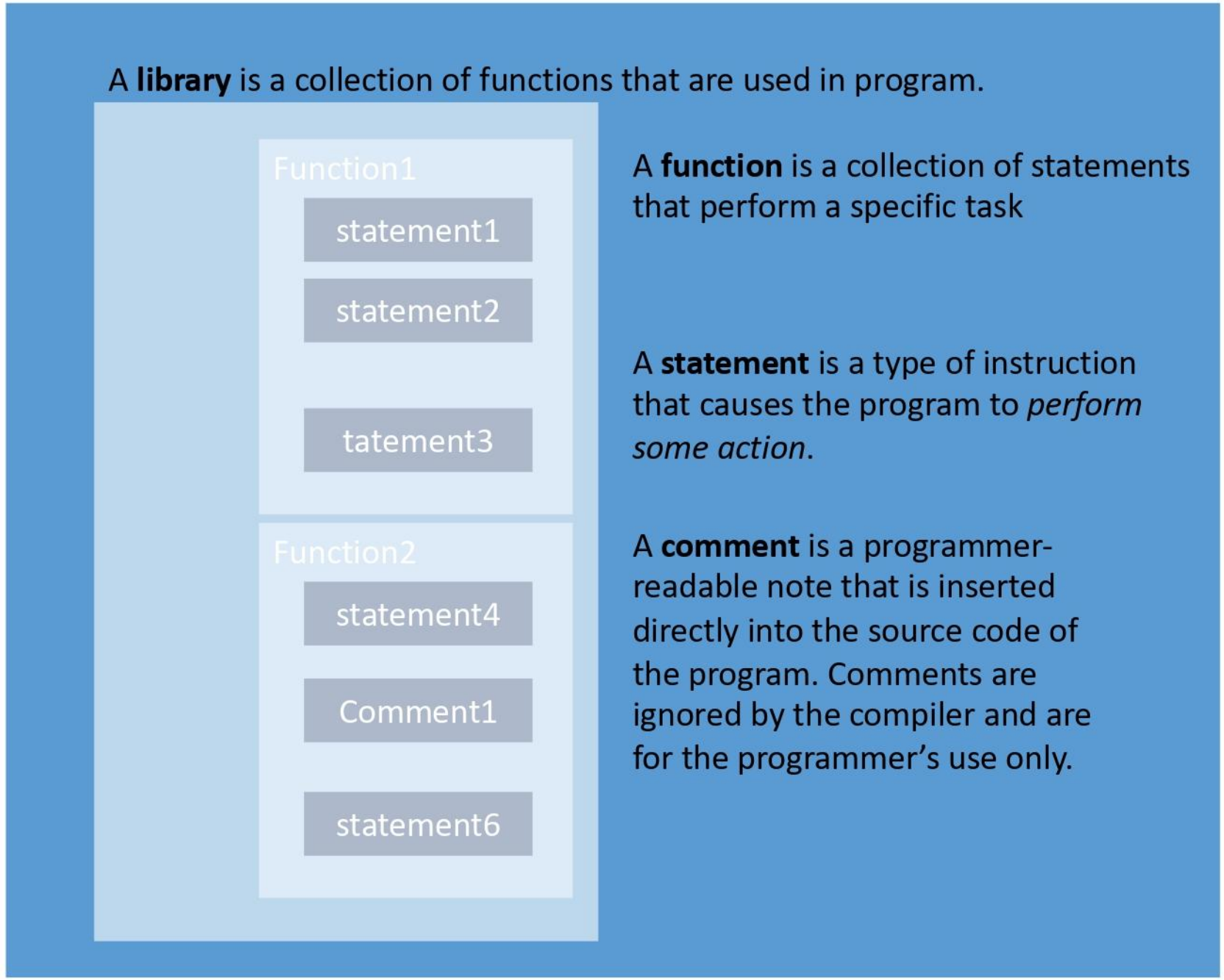
Type of Errors



Syntax

- Syntax is the formal grammar of the language, which specifies a well- formed statement the compiler will recognize
- in C++, the syntax of variable initialisation is:
  - data_type variable_name = value_expression;
- Example:
  - int number = 18;

A computer program is a set of instructions that tell the computer what to do



Structure of a C++ Program

- Statement -- smallest complete executable unit of a program
  - Declaration statement
  - Execution statement
  - Compound statement -- any set of statements enclosed in set braces { } (often called a block)
  - Simple C++ statements end with a semicolon. (A block does not typically need a semicolon after it, except in special circumstances).
- Sequence of statements, typically grouped into functions
  - function: a subprogram. a section of a program performing a specific task
  - Every function body is defined inside a block (see below)

Structure of a C++ Program

- For a C++ executable, exactly one function called main()
- Can consist of multiple files and typically use libraries
- Library
  - usually pre-compiled code available to the programmer to perform common tasks
  - Compilers come with many libraries. Some are standard for all compilers, and some may be system specific
  - Use the #include directive to make a library part of a program (satisfies declare-before-use rule)

Building and Running a C++ Program

- Pre-processing
  - The #include directive is an example of a preprocessor directive (anything starting with #).
  - #include<iostream> tells the preprocessor to copy the standard I/O stream library header file into the program
- Compiling
  - Syntax checking, translation of source code into object code (i.s. machine language). Not yet an executable program.
- Linking
  - Puts together any object code files that make up a program, as well as attaching pre-compiled library implementation code (like the standard I/O library implementation)
  - Endresultisafinaltarget--like executable program
- Run it!