Ahram Canadian University

Fundamental of programming 1

Dr. Marian Wagdy

Lecture 1

General Rules

Grades

- Midterm 25
- Quiz 5
- Attendance and participation 10
- Sheets 10
- Project or practical exam 10

Chapter 1

- Computer and programming
- Computer Organization
- Computer languages
- Program Development Life Cycle (PDLC)
- Introduction to C++

Computer and programming

- Every computer system is composed of multiple pieces of **hardware** and **software**.

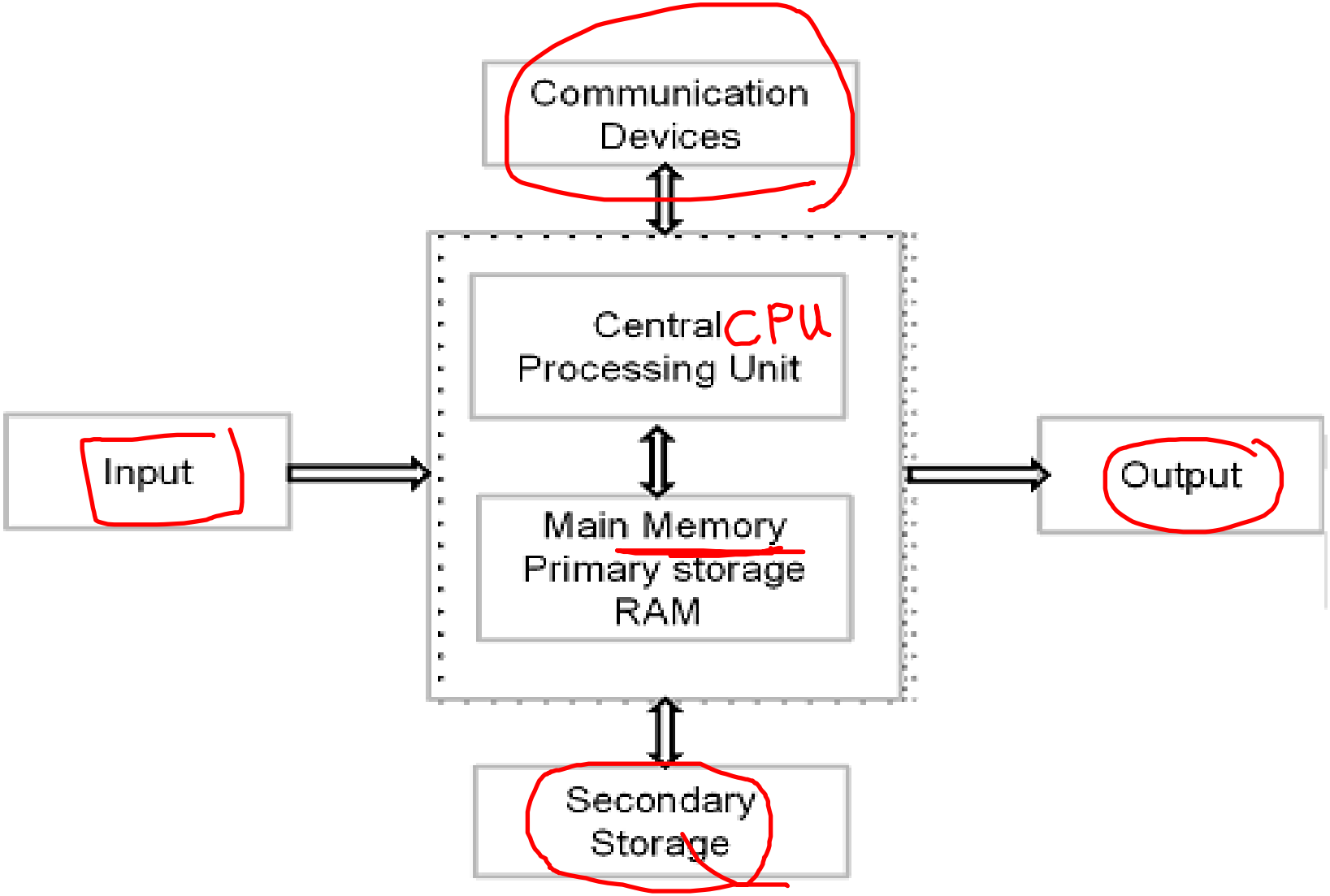
- **Hardware** is the equipment, or the physical devices.

  For example, keyboards, mice, speakers,

- **Software** is computer instructions that tell the computer instructions that tell the hardware what to do.

- **Software** is program, which are instruction sets written by programmers.
  - For example, businesses use word-processing and accounting programs and games and games

Computer Organization



Computer Organization

- Five main components
  - Input devices
    - Allows communication to the computer
  - Output devices
    - Allows communication to the user
  - Processor (CPU)
  - Main memory
    - Memory locations containing the running program
  - Secondary memory
    - Permanent record of data often on a disk

What is a programming language?

- A **programming language** is an artificial language that a computer understands.

- The language is made up of series of **statements** that fit together to form **instructions**.

- These instructions tell a computer what to do.

Computer languages

Computer languages may be divided into **three general types:**

1. High-level languages
2. Machine languages
3. Assembly languages

High-level Languages

- Common programming languages include …

C C++ Java Pascal Visual Basic FORTRAN Perl
PHP Lisp Scheme Ada C# Python

- These high – level languages
  - Resemble (look like) human languages
  - Are designed to be easy to read and write
  - Use more complicated instructions than the CPU can follow
  - Must be translated to zeros and ones for the CPU to execute a program

Low-level Languages
Assembly language

- An assembly language command such as

  ADD X Y Z

  might mean add the values found at x and y in memory, and store the result in location z.

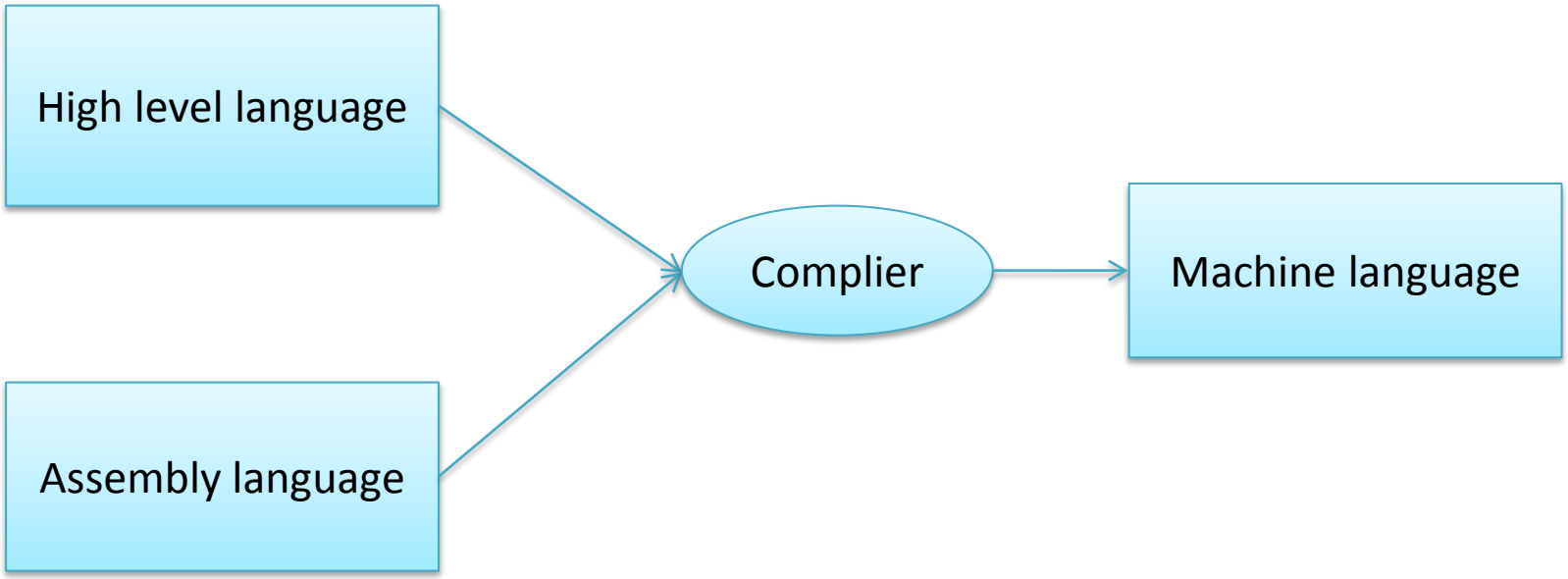
Machine language

- Assembly language must be translated to machine language (zeros and ones)

  0110 1001 1010 1011

- The CPU can follow machine language

Compiler



A compiler is a program that translates a high-level language program and low-level language into a machine-language that the computer can directly understand and execute.

What is the difference between:

1. **Algorithm**
2. **Program**
3. **Pseudocode**
4. **Flowchart**

- Algorithm
  - Algorithms are typically written in natural language or plain English.
- Program
  - An algorithm expressed in a language the computer can understand.
- Pseudocode
  - Pseudocode is written in a format resembling that of a high-level programming language.
- Flowcharts
  - Flowcharts are the graphical representation of logic.

Example

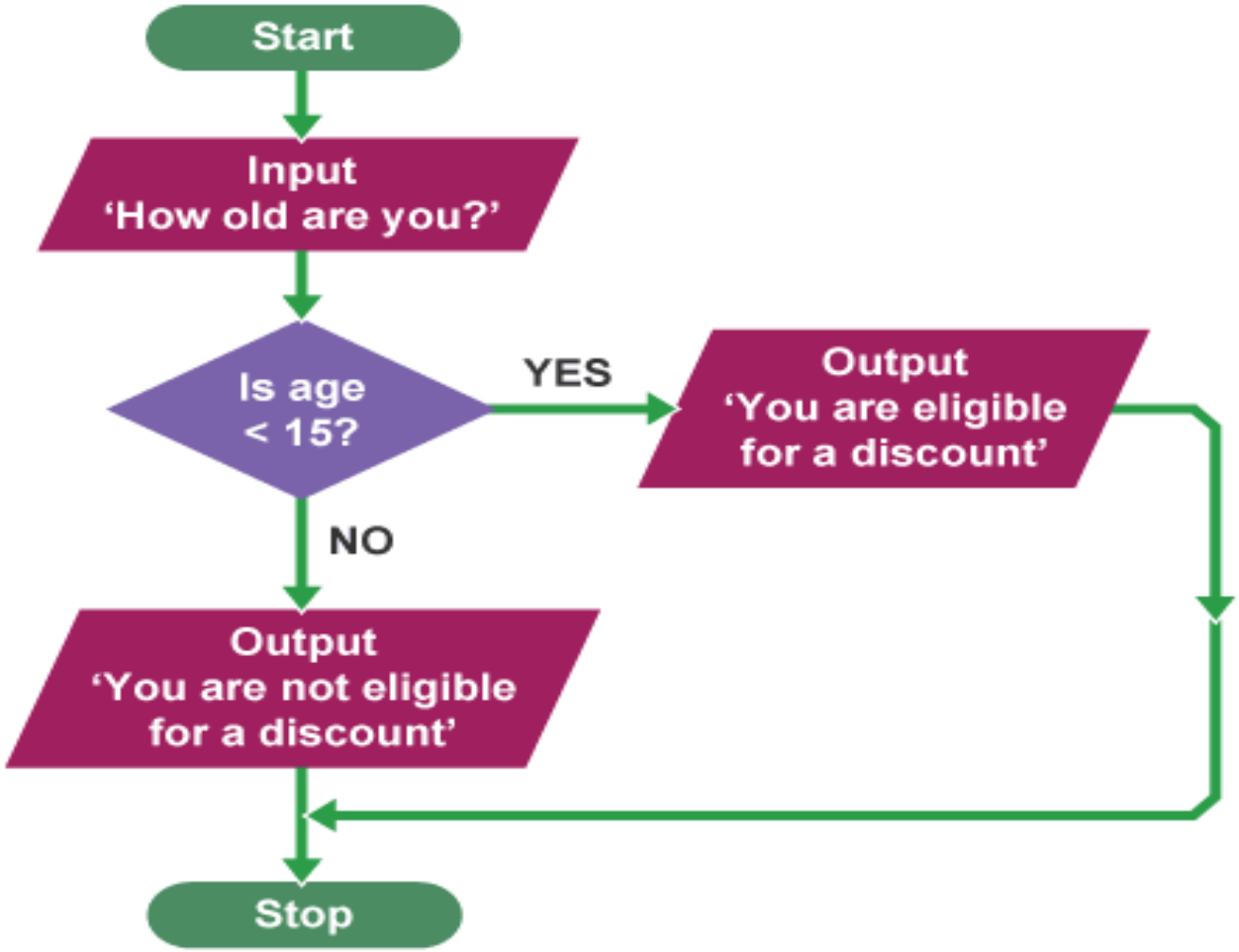
- Consider this simple problem. A cinema is offering discount tickets to anyone who is under 15. Decomposing this problem, gives this **algorithm**:

1. Find out how old the person is
2. If the person is younger than 15 then say “You are eligible for a discount ticket.”
3. Otherwise, say “You are not eligible for a discount ticket.”

- In pseudocode, the algorithm would look like this:

```
OUTPUT "How old are you?"
INPUT User inputs their age
STORE the user's input in the age variable
IF age < 15 THEN
    OUTPUT "You are eligible for a discount."
ELSE
    OUTPUT "You are not eligible for a discount."
```

In a **flowchart**, this algorithm would look like this:



- Creating the program in Python
- A Python (3.x) program to meet this algorithm would be:

```
age = int(input("How old are you?"))
if age < 15:
    print("You are eligible for a discount.")
else:
    print("You are not eligible for a discount.")
```

Program Development Life Cycle (PDLC)

1. Analyze the problem
2. Design the program
3. Code the program
4. Testing and Debugging
5. Maintenance

1- Analyze the problem

The computer user must figure out the problem, then decide how to resolve the problem - choose a program.

2- Design the program

- A flow chart is important to use during this step of the PDLC.
- This is a visual diagram of the flow containing the program.
- This step will help you break down the problem.

3- Code the program

- This is using the language of programming to write the lines of code.
- The code is called the listing or the source code.
- The computer user will run an object code for this step.

4- Testing and Debugging

- **Bug**
  - A mistake in a program
- **Debugging**
  - Eliminating mistakes in programs

- Debug or testing the program. The computer user must debug. This is the process of finding the "bugs" on the computer. The bugs are important to find because this is known as errors in a program.

- Formalize the solution. One must run the program to make sure there are no syntax and logic errors. Syntax are grammatical errors and logic errors are incorrect results.

Program Errors

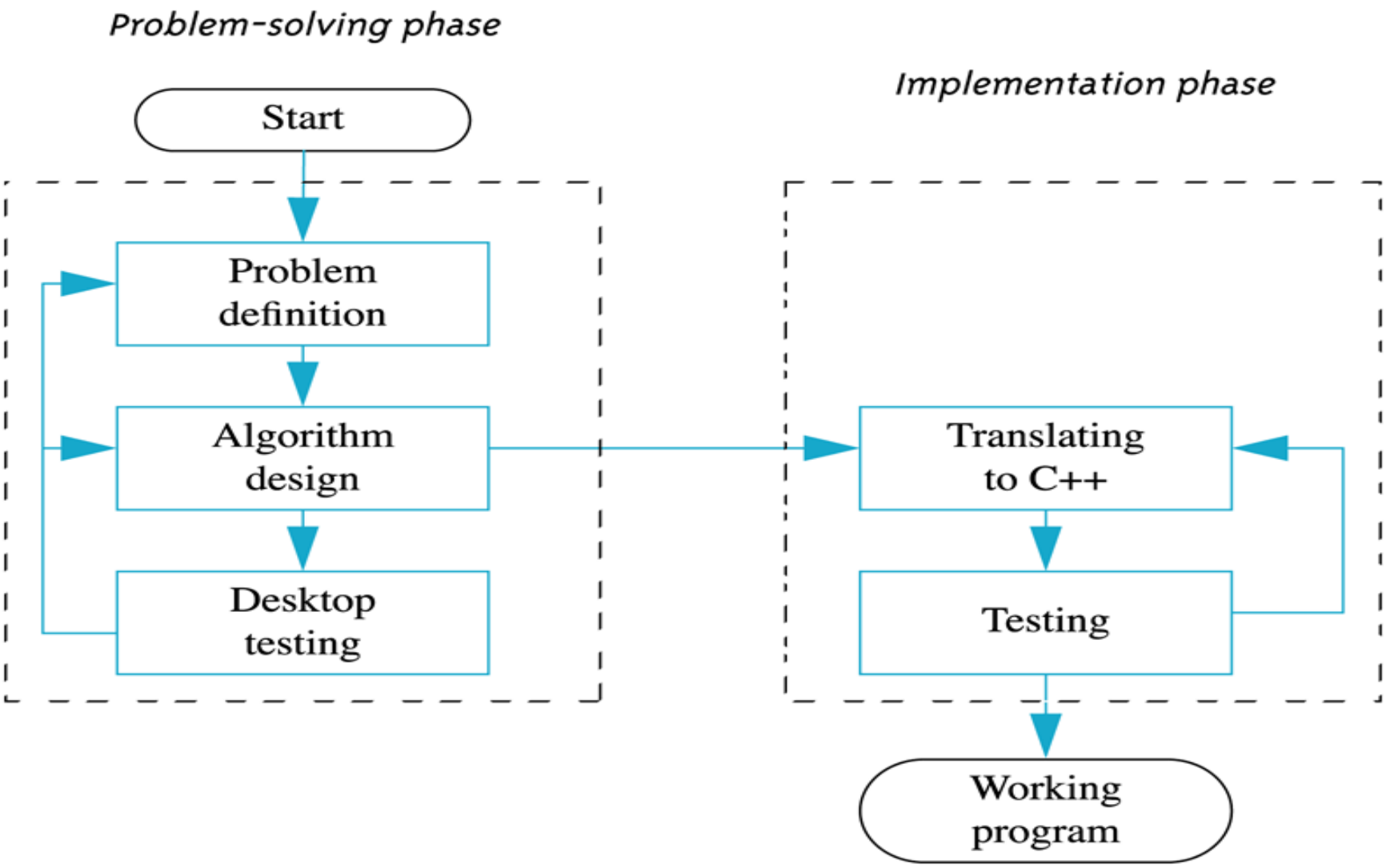
1. Syntax errors
   - Violation of the grammar rules of the language
   - Discovered by the compiler
     - Error messages may not always show correct location of errors
2. Run-time errors
   - Error conditions detected by the computer at run-time
3. Logic errors
   - Errors in the program's algorithm
   - Most difficult to diagnose
   - Computer does not recognize an error

5- Maintenance

- This step is the final step of gathering everything together.

- Internal documentation is involved in this step because it explains the reasoning one might of made a change in the program or how to write a program

Program Design Process



Introduction to C++

- Where did C++ come from?
  - Derived from the C language
  - C was derived from the B language
  - B was derived from the BCPL language
- Why the '++'?
  - ++ is an operator in C++ and results in a cute pun

C++ History

- C developed by Dennis Ritchie at AT&T Bell Labs in the 1970s.
  - Used to maintain UNIX systems
  - Many commercial applications written in c
- C++ developed by Bjarne Stroustrup at AT&T Bell Labs in the 1980s.
  - Overcame several shortcomings of C
  - Incorporated object oriented programming
  - C remains a subset of C++

A Sample C++ Program

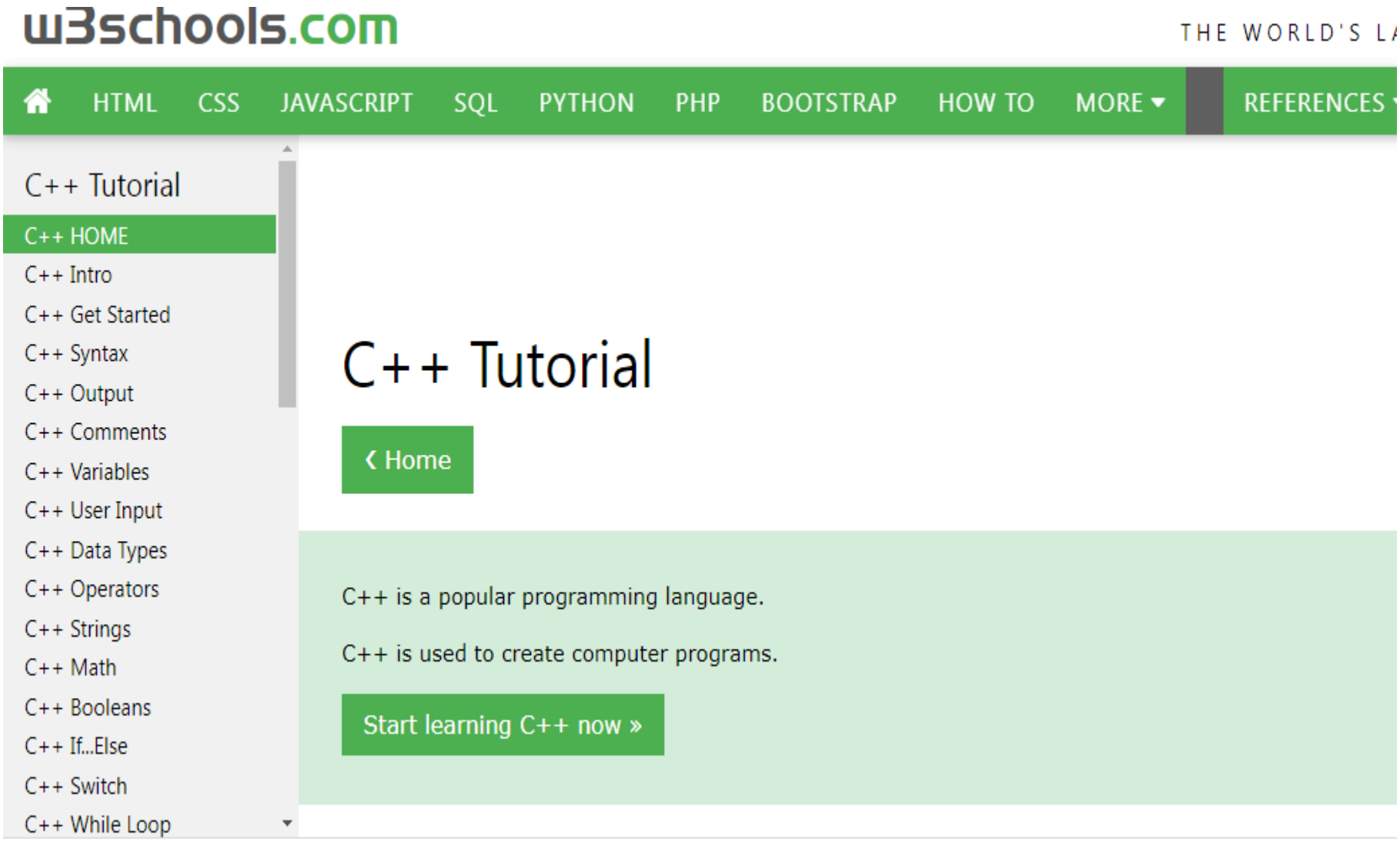
- A simple C++ program begins this way

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

int main()
{
```

- And ends this way

```
        return 0;
}
```


w3schools.com
THE WORLD'S LA
HTML
CSS
JAVASCRIPT
SQL
PYTHON
PHP
BOOTSTRAP
HOW TO
MORE
REFERENCES
C++ Tutorial
C++ HOME
C++ Intro
C++ Get Started
C++ Syntax
C++ Output
C++ Comments
C++ Variables
C++ User Input
C++ Data Types
C++ Operators
C++ Strings
C++ Math
C++ Booleans
C++ If...Else
C++ Switch
C++ While Loop
C++ Tutorial
❮ Home
C++ is a popular programming language.
C++ is used to create computer programs.
Start learning C++ now »

Thanks