CSCI251/CSCI851 Autumn-2020 Advanced Programming **(S2a)**

C++ Foundations I:

Introduction to Programming

Java vs C++

Outline

- What is programming?
- Programming paradigms.
- Java vs C++.
- Machine language and the JVM.
- Compilers and linkers.

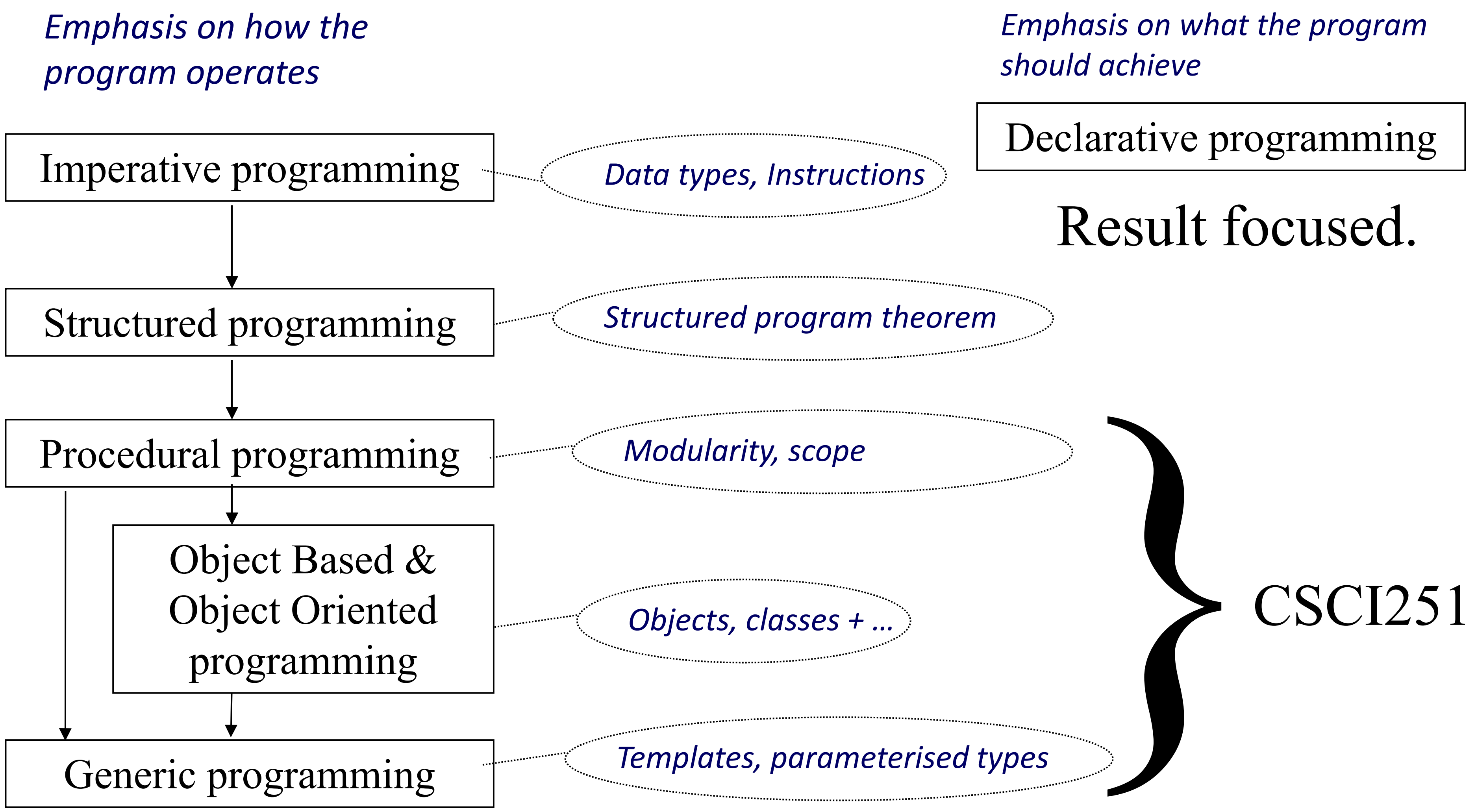
What is **programming**?

- Writing **instructions** for a **computer** with the purpose of getting the computer to perform required tasks.
  - I prefer to think of programming as encompassing a lot of the design too, rather than just being the writing the code part.
- The instructions are written in a programming **language** which has a specified **syntax**.
- In particular there is syntax associated with:
  - **Input** and **output**
  - **Variables and data types**
  - **Control structures**
  - etc.

Fundamental concepts: I/O, variables, control structures…

- Input is getting data from outside …
- Output is sending data to outside …
- For a given value of ***Outside***: Outside a program, outside a function, outside the computer …
- Variables are places within the program where we store data.
- Control structures relate to where we use the results of tests to determine where we go next.

Programming paradigms



Imperative:
Statements change the program state and describe how things happen.

- Don’t get too carried away with the separation though.
- You can write procedural programs that call black-box functions or modules.
  - So you know want you want to be done but don’t want to know the details → which is heading towards being declarative, but only at a certain level.

C++ vs Java: Paradigms

- Java is object oriented.
  - It's difficult to avoid objects in Java, everything has to be in a class … but some of the primitive data types are not objects so it's not a pure object oriented language.
  - Java does have some generic programming.
- C++ can be object oriented too but …
  - You don't have to have classes → procedural programming.
  - Even when objects are around you don't need to use all the object oriented programming features, so programming can be object-based.
  - C++ can also be generic, providing extra abstraction.

- Java is used for implementing client-server web based applications, among other things.
- It’s often described as being platform independent, …
- … but it’s perhaps better to think of it as being a dedicated platform that you are writing for, the Java Virtual Machine (JVM).
- The Java Virtual Machine is a platform dependent application that runs on hardware/OS.
  - In many cases, but not all, the JVM is written in C or C++.

- C++ is mostly used for desktop applications and systems programming.
    - Systems programming produces software that serves the system, vs application programming which serves the user.
  - Bjarne Stroustrup writes ("Foundations of C++": ETAPS 2012) … http://www.stroustrup.com/

  "The aim of C++ is to help classical systems programming tasks."
  - It's also used for embedded systems, resource constrained systems, and large systems.
- The evolution of C++ standards has driven it towards platform independence, but it has operating system dependent functionality.

Machine language and the JVM

- Computers can understand only a machine language, which is a sequence of binary instructions (0’s and 1’s).
- Generally, programs are not written in machine language, although it is possible.
- There are “simpler” languages to program in:
  - Assembly language.
  - High-level languages, such as C or C++.
- While C++ program runs as executable native machine code; so directly on the hardware/OS, a Java program runs in a **Java Virtual Machine (JVM)**.

- Java is good for standardisation, safety, and web programming, but is often slow and less powerful which is relevant for top end game coding and building large complex applications.
  - Performance isn't such a problem if you have small bursts of activity driven by user inputs, users are slow…
- The JVM manages hardware/OS resources.
- C++ allows more direct control of the hardware resources, including using memory pointers.
  - This means you have to be more careful to avoid resource mishandling.
  - We will look at programming defensively from quite early on.

C++ vs Java: OO differences

- There are naming differences:
  - Java has fields and methods, C++ has data members and member functions.
    - I'll refer to member functions as methods sometimes anyway.
- And different functionality:
  - C++ supports multiple inheritance.
  - C++ supports operator overloading, in addition to function overloading.
  - C++ supports C-functionality, including structs and unions.
- We are going to move away from object oriented programming initially.

Warning: Hello World …

```
alert('Hello, world!');
```

Javascript

```
public class HelloWorld {
     public static void main(String[] args) {
          System.out.println("Hello, world!");
     }
}
```

Java

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

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

C++

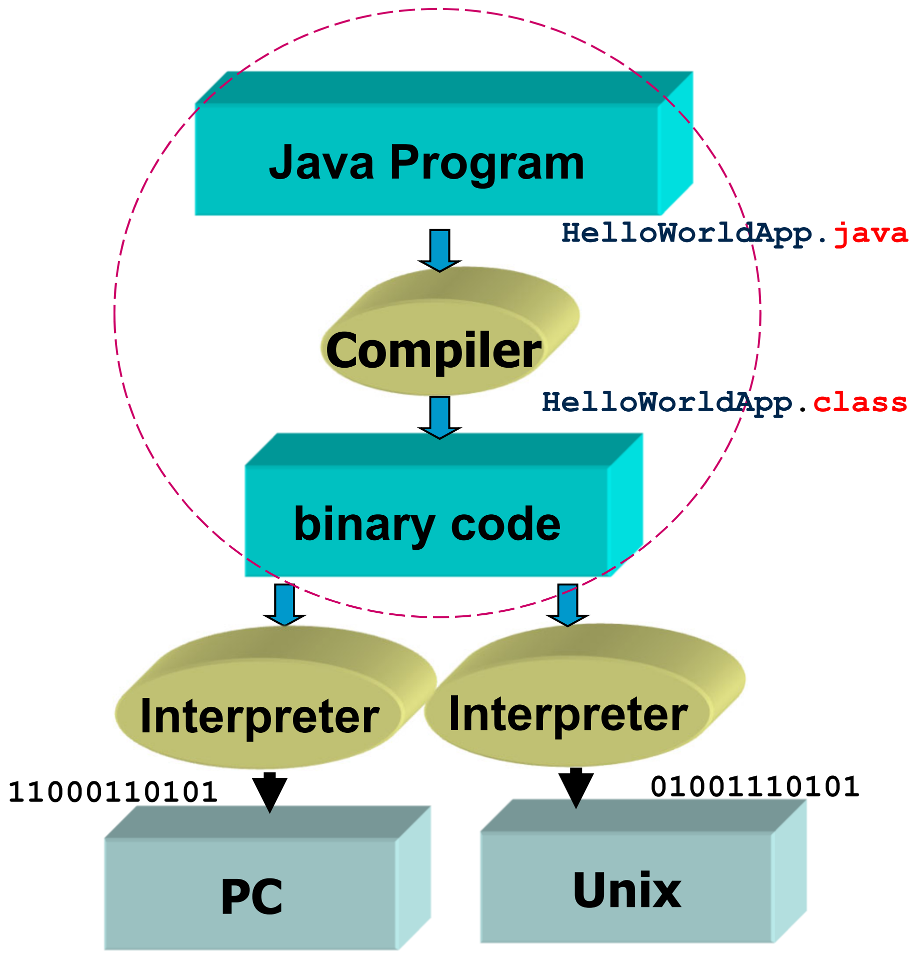
Both have a lot of syntax ☹

Compilers and linkers

- Before the computer can run any program it needs to be converted from the programming language into machine instructions.
- This is the job of
  - A compiler, and/or
  - An interpreter …
- … in combination with a linker.

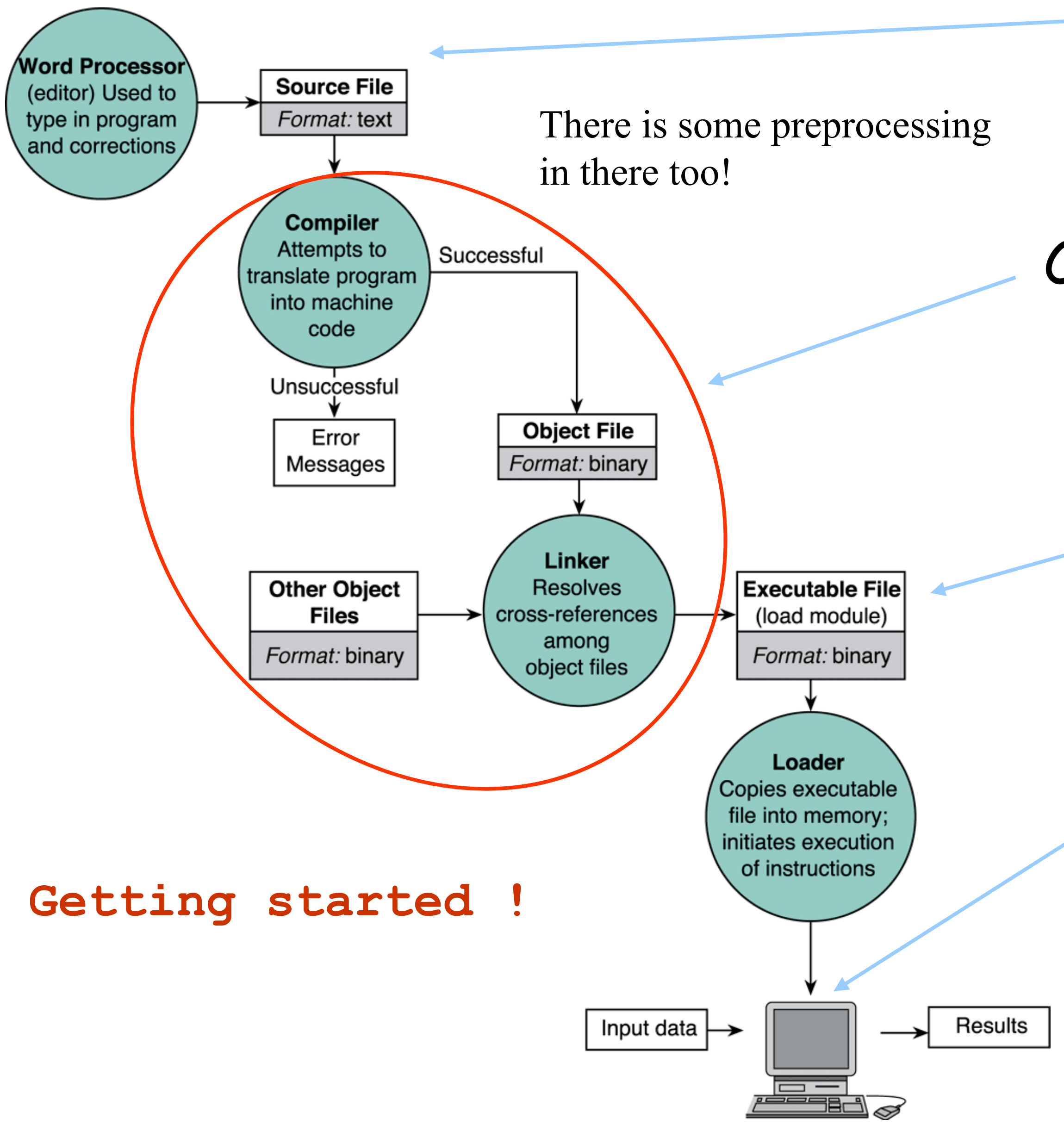
Java

- Roughly …
- The compiler take a .java file and produces a .class file, so bytecode.
- The JVM interprets that bytecode to turn it into native machine code.
- There are compilers that produce native machine code for some specified platform.



From CSIT111

C++



Word Processor (editor) Used to type in program and corrections
Source File
Format: text
Compiler Attempts to translate program into machine code
Successful
Unsuccessful
Error Messages
Object File
Format: binary
Other Object Files
Format: binary
Linker Resolves cross-references among object files
Executable File (load module)
Format: binary
Loader Copies executable file into memory; initiates execution of instructions
Input data
Results

There is some preprocessing in there too!

hello.cpp

CC -o hello hello.cpp

hello

Hello World !

Getting started !

What’s next?

- Basics of C++ syntax and some preliminary procedural programming.
- Enough that you can do some decent work for the first lab exercises.