C++ Foundations I: Introduction. Java vs C++

Commencement of Session

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

Outline

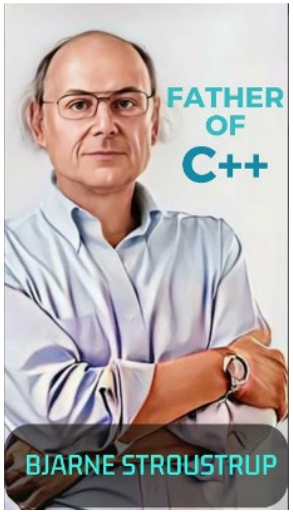
- What is programming?
- What is C++?
- Programming paradigms.
- Java vs C++.
- Machine language and the JVM.
- Compilers and linkers.

What is **programming**?

- Writing **instructions** for a **computer** with the purpose of getting the computer to perform required tasks.
  - Programming is treated as encompassing 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.
- A fair bit of Java/C++ syntax is similar.

What is C++?

- One of the world's most popular programming languages
- High-level programming language
- Freedom of control over system resources and memory
- Portable with many applications

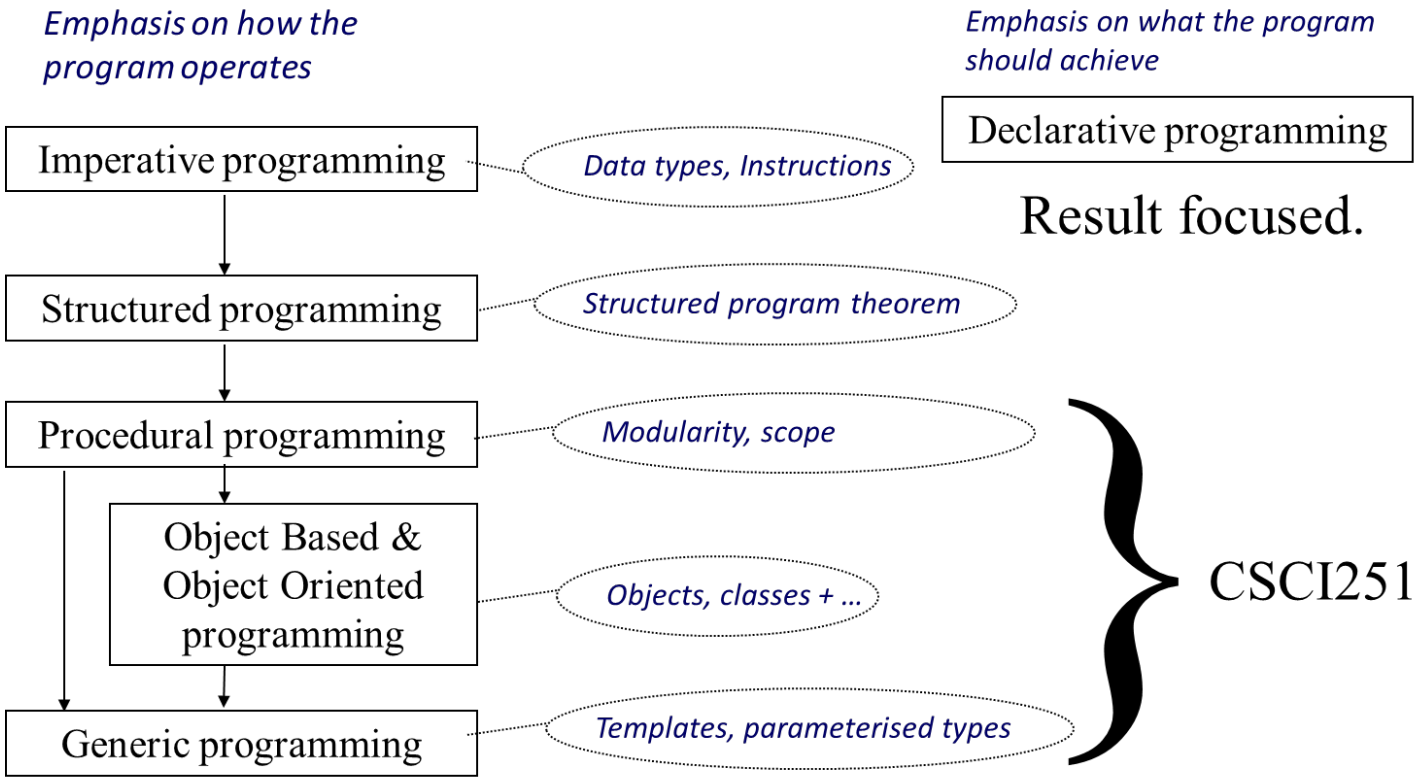


Why I created C++?

Fundamental concepts

- **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



C++ vs Java

- Java is object oriented.
  - It's difficult to avoid objects in Java, everything has to be in a class
- C++ can be object oriented too but …
  - You don't have to have classes → procedural programming.
  - programming can be object-based.
- Both Java and C++ can be generic

C++ vs Java

- Platform dependent vs Platform independent
- The Java Virtual Machine is a platform dependent application that runs on hardware/OS.
- Java is used for implementing client-server web based applications, among other things.

C++ vs Java

- C++ allows more direct control of the hardware resources, including using memory pointers
  - IOT devices
- C++ is mostly used for desktop applications and system-level programming.
  - Systems programming produces software that serves the system
  - application programming serves the user
- Also for game coding and building large complex applications.

C++ vs Java

```
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;
}
```

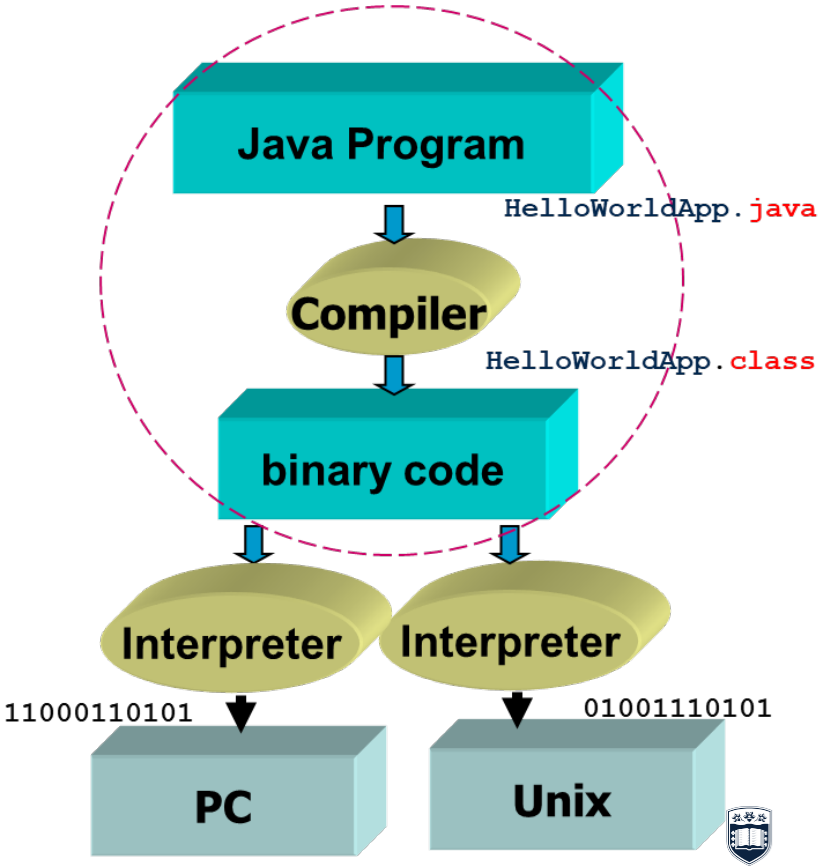
C++

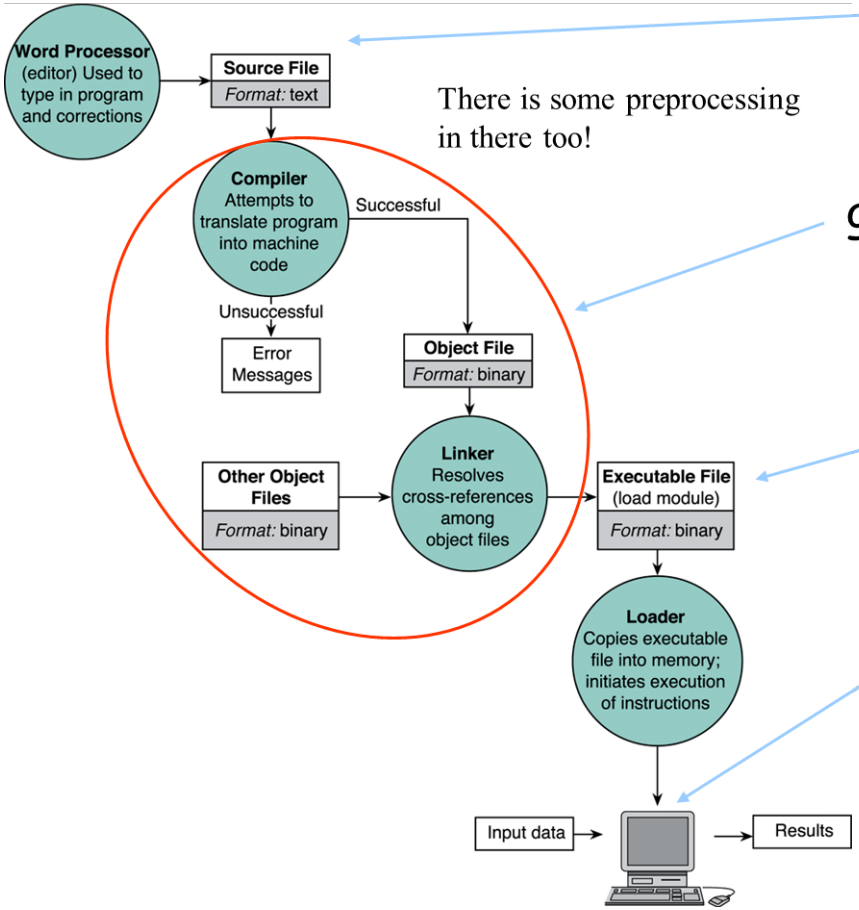
Compilers and linkers

- 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

- 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.





hello.cpp

g++ -o hello hello.cpp

hello

Hello World !

C++

- Compilation and running: (on Ubuntu)
- We saw some examples of code and how to compile them.

```
$ g++ hello.cpp –o hello
$ ./hello

Or
$ g++ -c hello.cpp –o hello.o
$ g++ hello.o –o hello
```

C++14 and C++17 and C++20 and …

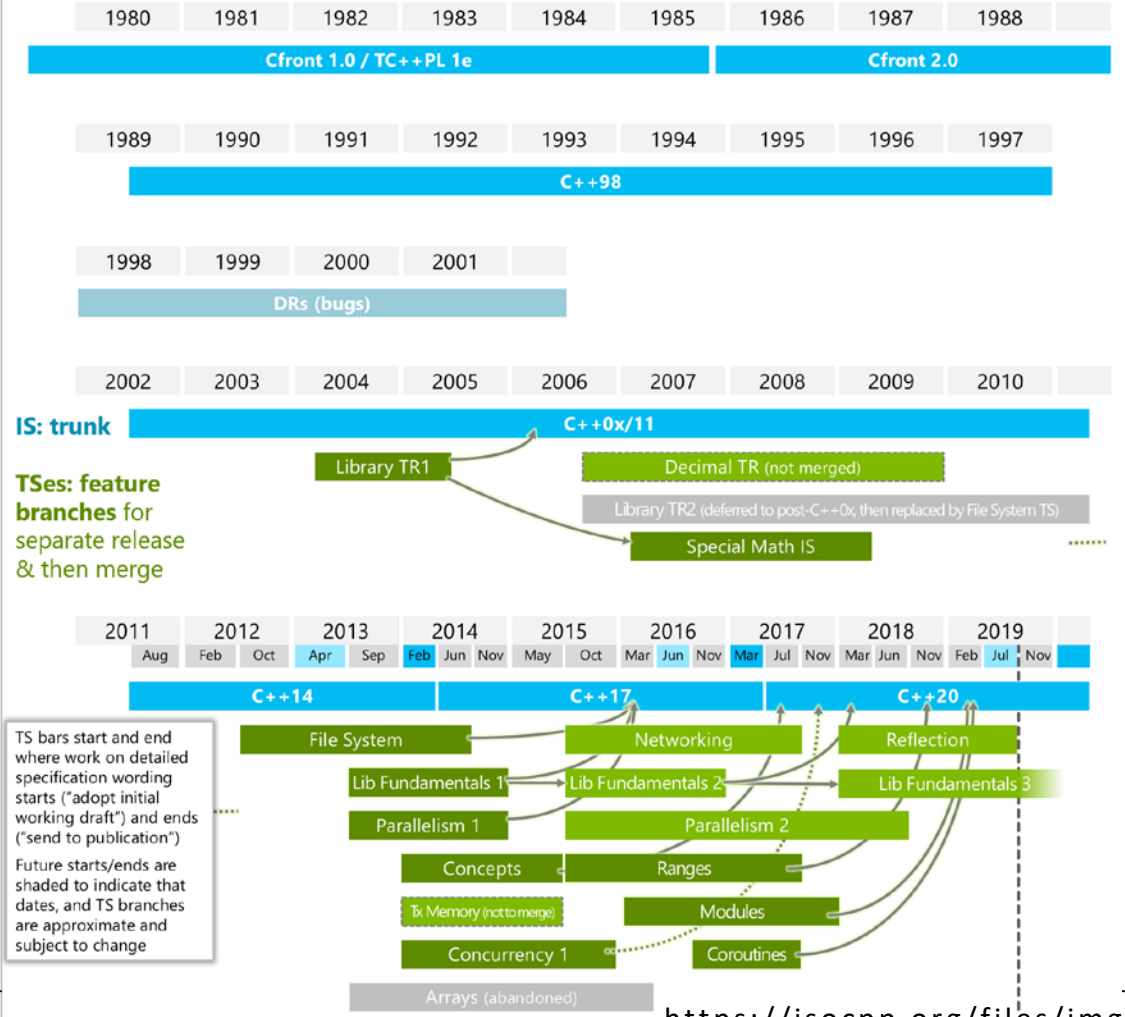
UNIVERSITY
OF WOLLONGONG
AUSTRALIA

C++ continues to evolve.

Versions and compiler support …

- **News, Status & Discussion about Standard C++ :** https://isocpp.org/
- The plan is for new C++ versions to be released every 3 years.
- If you end up using C++, you should watch language developments.

- There is a list of language features for different C++ versions, and availability under compilers at:
- https://en.cppreference.com/w/cpp/compiler_support



https://isocpp.org/files/img/wg21-timeline-2019-07.png

C++14

- generic lambda expressions
  - auto identity = [](auto x) { return x; };
  - int three = identity(3); // 3
  - std::string foo = identity("foo"); // "foo"
- relaxing constraints on constexpr functions
  - In C++11, constexpr function bodies could only contain a very limited set of syntaxes; in C++14, there is more
- variable templates
  - template<class T>
  - constexpr T pi = T(3.14159265358979323 85);

C++17

- C++17 standard
  - https://www.iso.org/standard/68564.html
- Changes between C++14 and C++17 are described at
  - https://isocpp.org/files/papers/p0636r0.html
  - https://github.com/AnthonyCalandra/modern-cpp-features
  - https://www.codingame.com/playgrounds/2205/7-features-of-c17-that-will-simplify-your-code/introduction

C++17

- Nested namespaces
  - namespace A { namespace B { namespace C { …}
- Inline variables
  - static inline int count{0}; // declare and initialize count to 0 within the class
- Special maths functions
- Filesystem library
  - library provides a standard way to manipulate files, directories, and paths in a filesystem, such as: std::filesystem::exists OR std::filesystem::create_directory

C++20

- There are already a lot of references to C++20 at : https://en.cppreference.com/w/
- Feature test macros.
- Formatting library.
- Concepts library.
- Ranges library.
- Atomic references.