UNIVERSITY OF LIVERPOOL

Application Development with C++ (ELEC362)

Lecture 1: Introduction

mihasan@liverpool.ac.uk

This lecture

- What is covered in this lecture?

  1. Introduction to the module. 2. Overview of C++.

- Why is it covered?

  1. To explain the module's structure and assessment.

  2. To appreciate the advantages of C++ .

The learning outcomes

- This module will teach you:
  1. How to use C++ to implement console-based software.
  2. How to use C++ to implement Graphic User Interface (GUI)-based software.
  3. How to effectively use online documentations and resources for self-learning in programming context.

Note: This module is NOT a computer science degree in a single module!

Basic information

- Pre-request modules:

  ELEC129

  Or ELEC102 (XJTLU students)

  Or COMP122 (Comp Sci & EE students)

- Background knowledge : algorithms, basic programming experience, and basic maths.

- Statistics from last year: Average mark (60%) Failure rate (3%) Top mark (81).

Workload and assessment

- The module is 15 credits module (corresponding to 150 hours of study) distributed as:

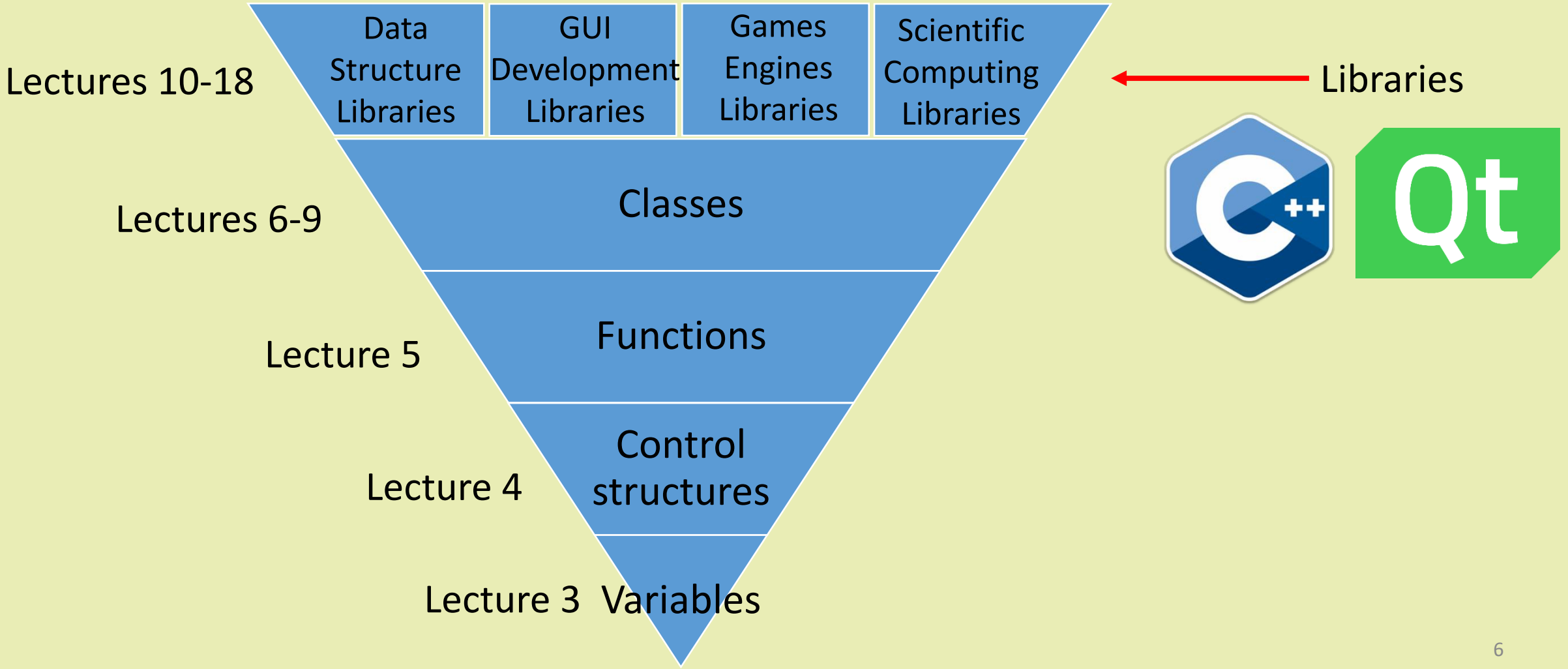
  Lectures (18 hours) Lab sessions (24 hours) Exam (2 hours)

  Project and assignment (48 hours total) Self-study (58 hours)

- Mark distribution:

Assessment	Weight (%)	Duration	Assessment aspect
Assignment	10	2 weeks	Console-based software
Project	50	7 weeks	GUI-based software Independent learning
Exam	40	2 hours	C++ concepts

Topics covered



Resources

- Books: Professional C++ , Marc Gregoire (2018).

  C++ GUI Programming with Qt, J Blanchette & M Summerfield (2008).

- Online documentation: C++ : http://www.cplusplus.com/reference/

  Qt : https://doc.qt.io/

- Educational websites: https://www.geeksforgeeks.org/c-plus-plus/

  https://www.tutorialspoint.com/cplusplus/

- Online developers community: https://stackoverflow.com/

  https://forum.qt.io/

C++ Overview

- C++ is a general purpose programming language, appeared in 1983.
- Since its creation, many standard editions of C++ where released including C++ 98, C++ 11, C++ 14, and C++ 17.
- In its early times there were many other programming languages:

Language	Primary use
COBOL	Business theme
Fortran	Scientific computing
Pascal	Data structures

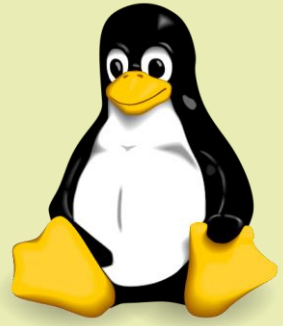
- The Object Oriented Programming (OOP) paradigm enabled C++ to be customisable for any field.

C++ Overview

- As a result, C++ became widely used on all levels of software including:

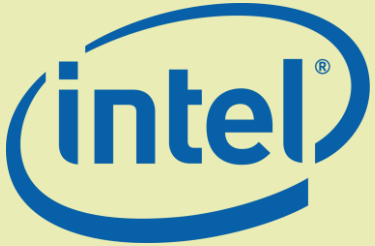
Operating systems





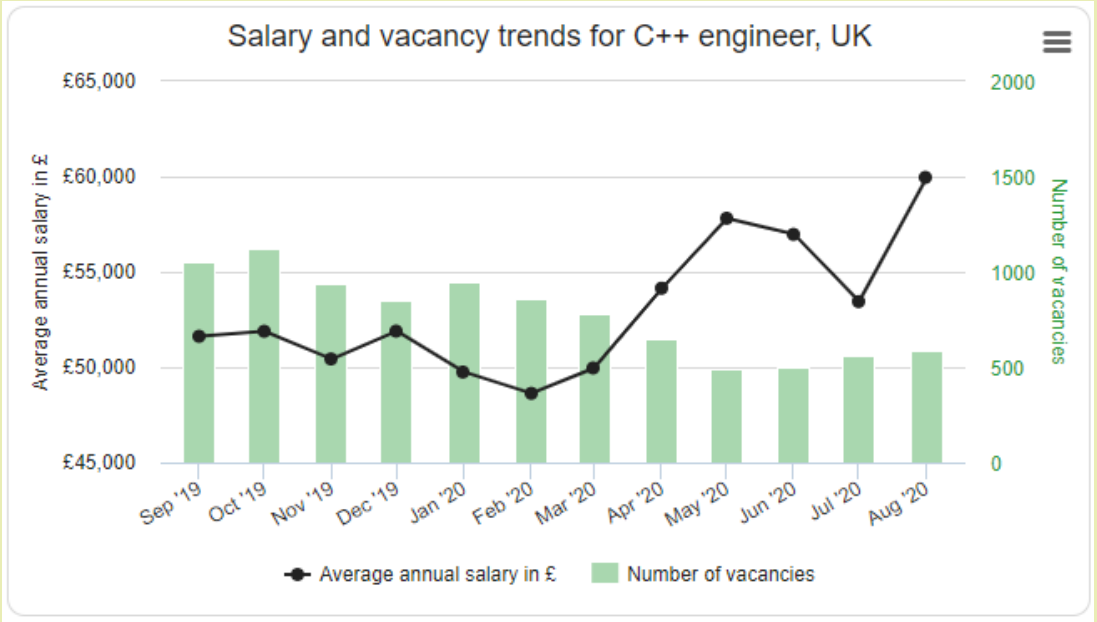
Applications

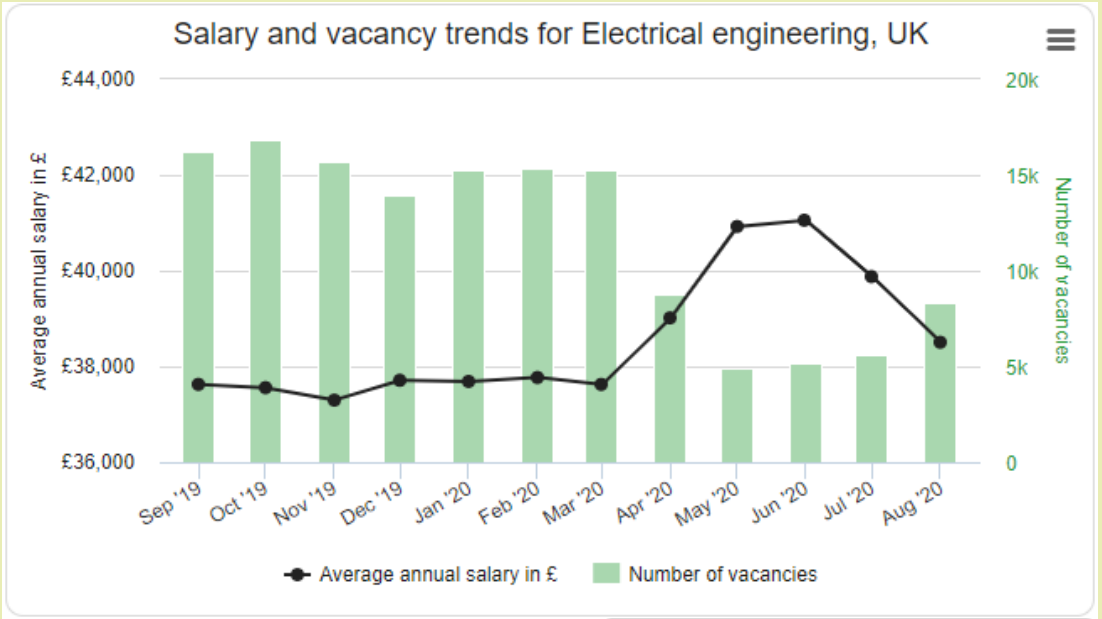
Firmware



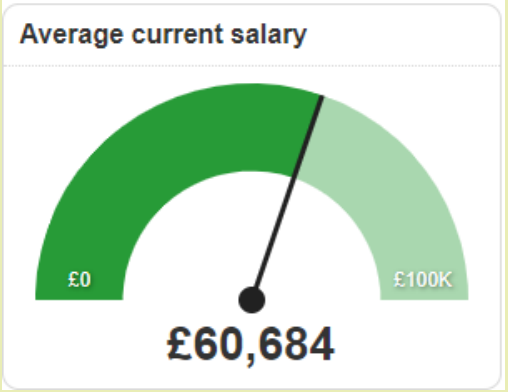
C++ Overview

- Despite it being old, C++ is still a valuable skill to have:

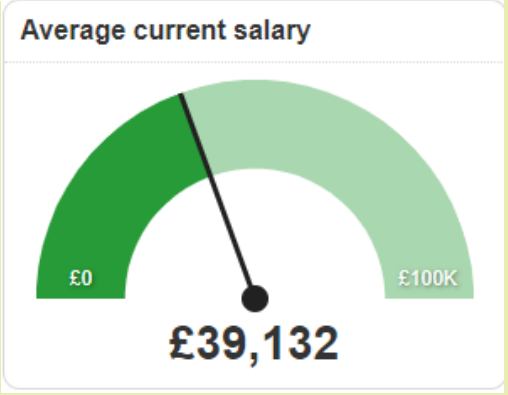




https://www.adzuna.co.uk/jobs/salaries/c++-engineer



https://www.adzuna.co.uk/jobs/salaries/electrical-engineering



High level programming languages

- Based on the level of programming languages, they are split into two categories:

1. High level languages: programming languages that use human readable syntax.

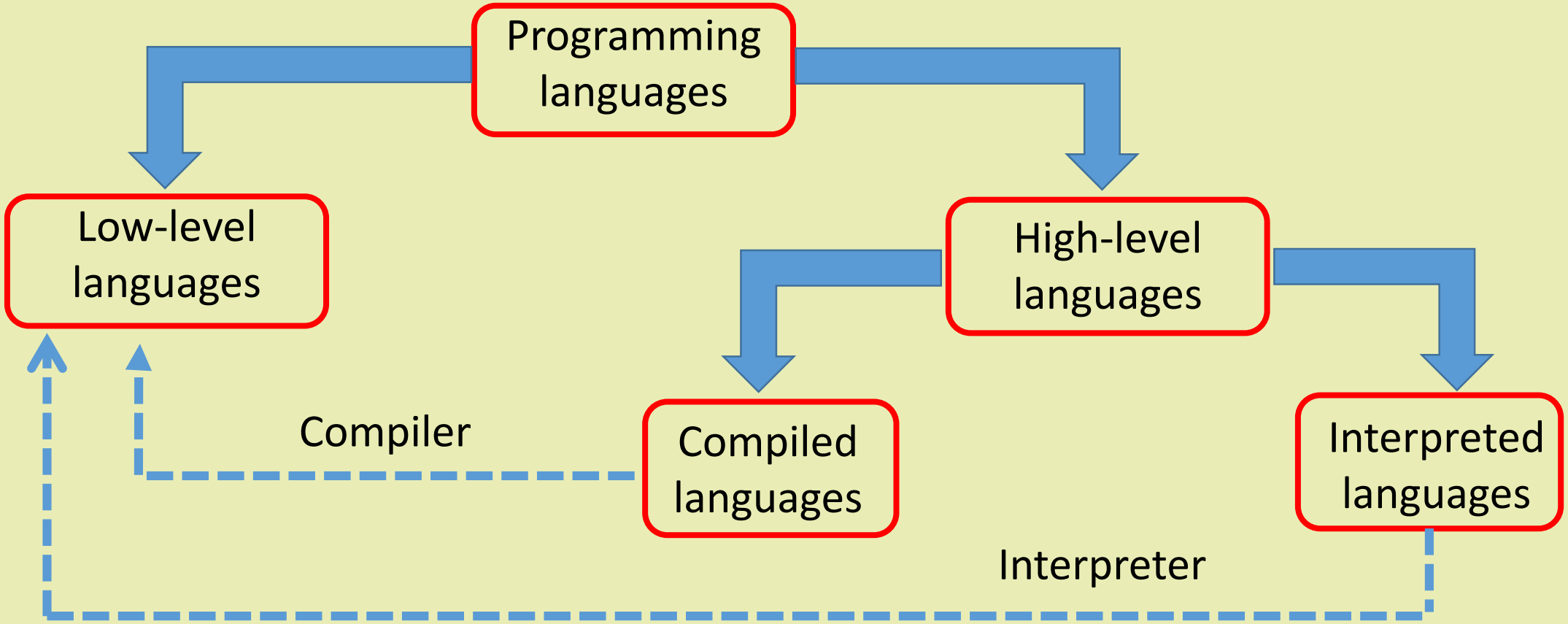
   Examples include: C++, MATLAB, Python.

2. Low level languages: programming languages that use machine readable syntax.

   Examples include: Assembly language.

- The computer only understands Low level languages.
- We need a “translator” software to translate high level to low level languages.

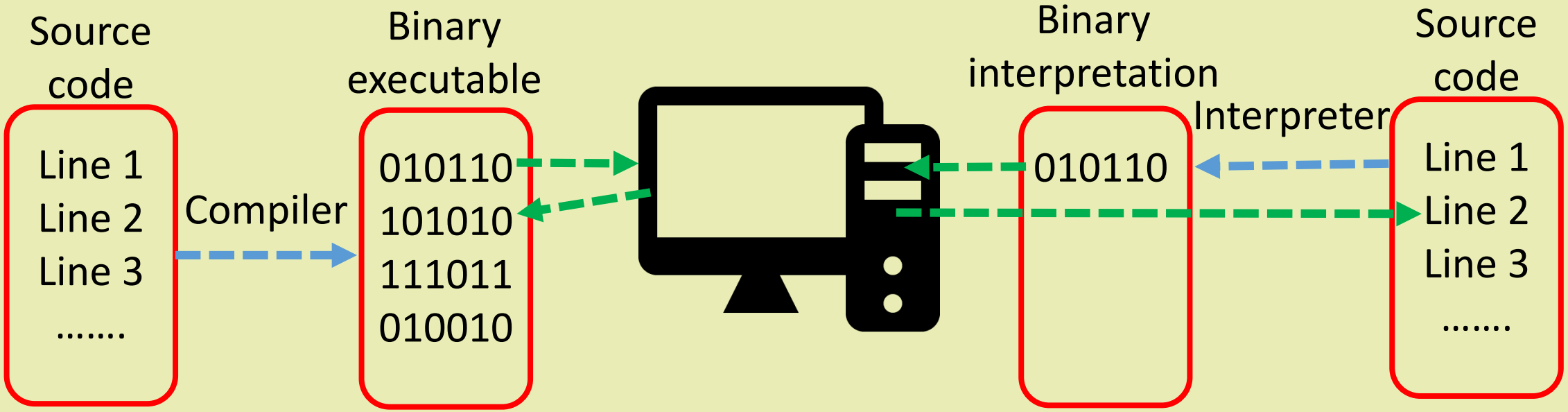
High level programming languages



- The “translator” can be either a compiler or an interpreter, thus leading to compiled languages and interpreted languages.

Compiled vs Interpreted languages

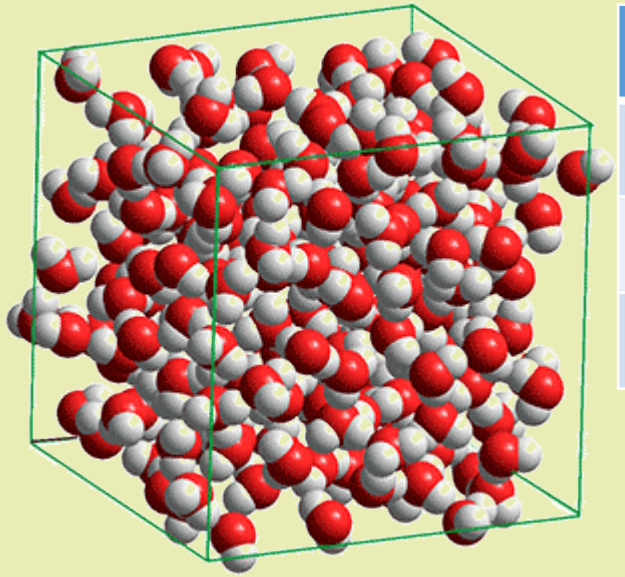
- Compiled languages:



- Interpreted languages:
- As a result of the “real-time” interpretation of interpreted languages, they are significantly slower than compiled languages.

Compiled vs Interpreted languages

- Example of compiled languages are C++ and Fortran. While examples of interpreted languages are MATLAB and Python.



Simulation steps	Time (Fortran)	Time (MATLAB)
7000	15 sec	4439 sec
25000	38 sec	13 hours
105000	138 sec	???

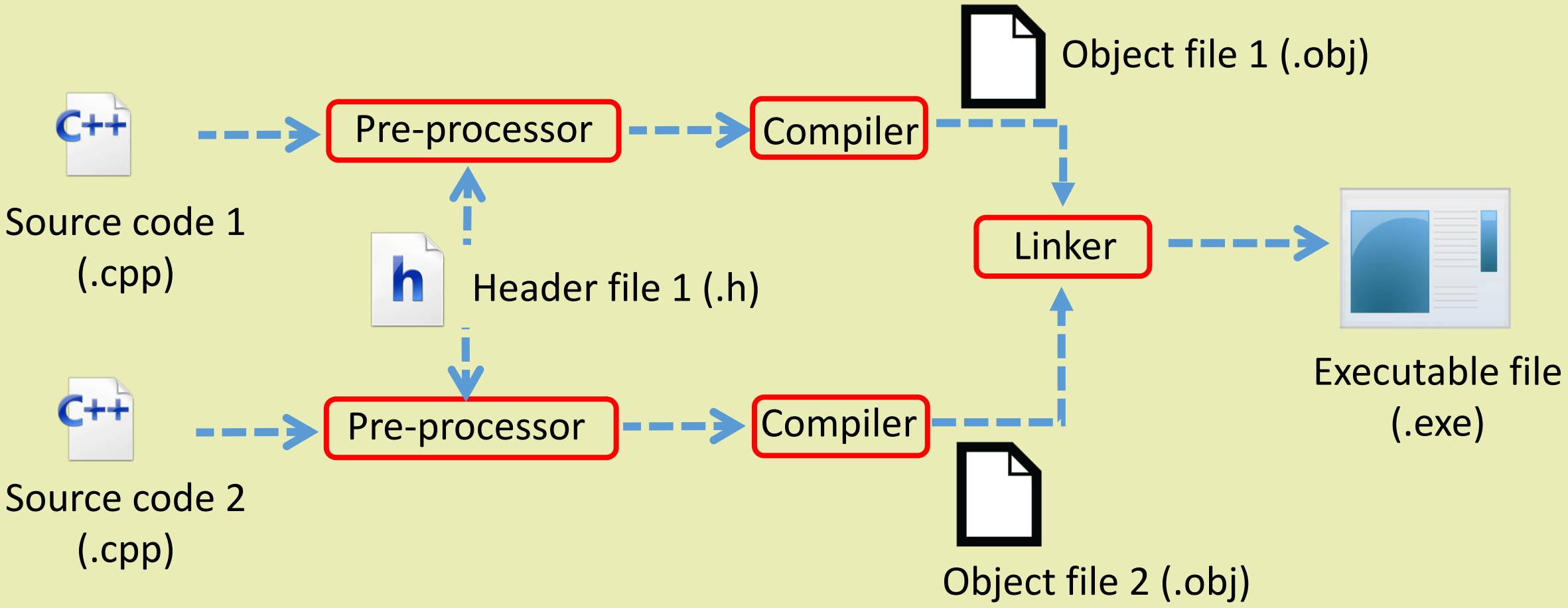
http://utkstair.org/clausius/docs/che548/pdf/md_sim.pdf

- Interpreted languages has the advantage of being much easier to code and to debug in comparison to compiled languages.

The compilation process

- The code written by the software developer is knowns as the source code.
- Source code: The version of software as it is originally written by a human in plain text.
- Header file: A file containing parts of code to be shared among multiple source files.
- Executable file: A file that runs the programme when it is opened.
- The end user only deals with the executable file.

The compilation process



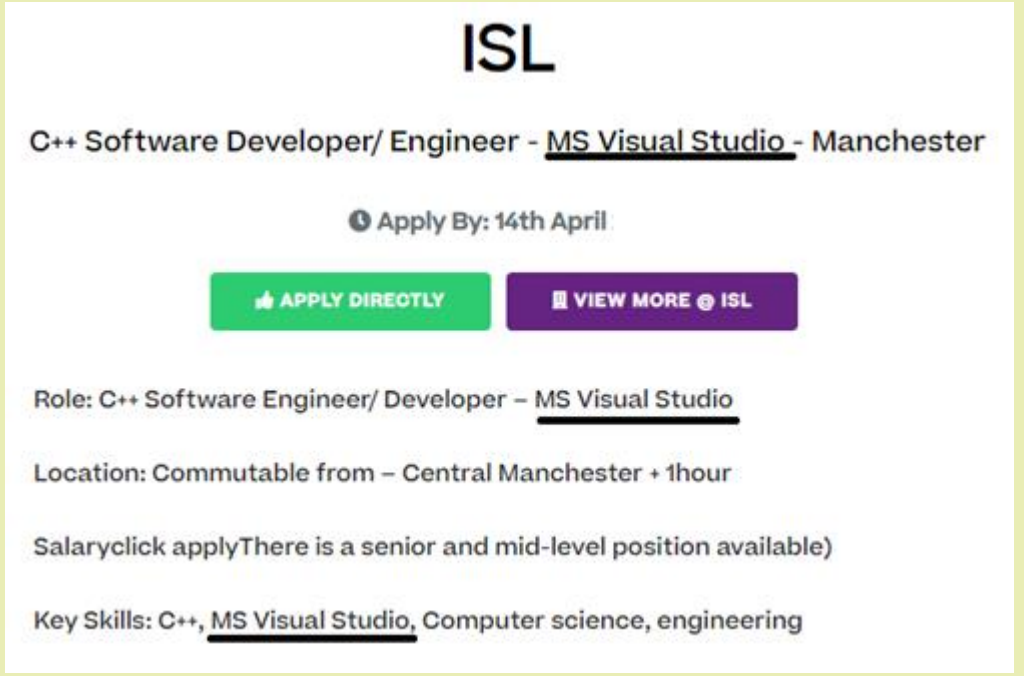
Integrated Development Environments (IDE)

- Code development is done using software known as IDE.

- For this module we will use Visual Studio (community version):

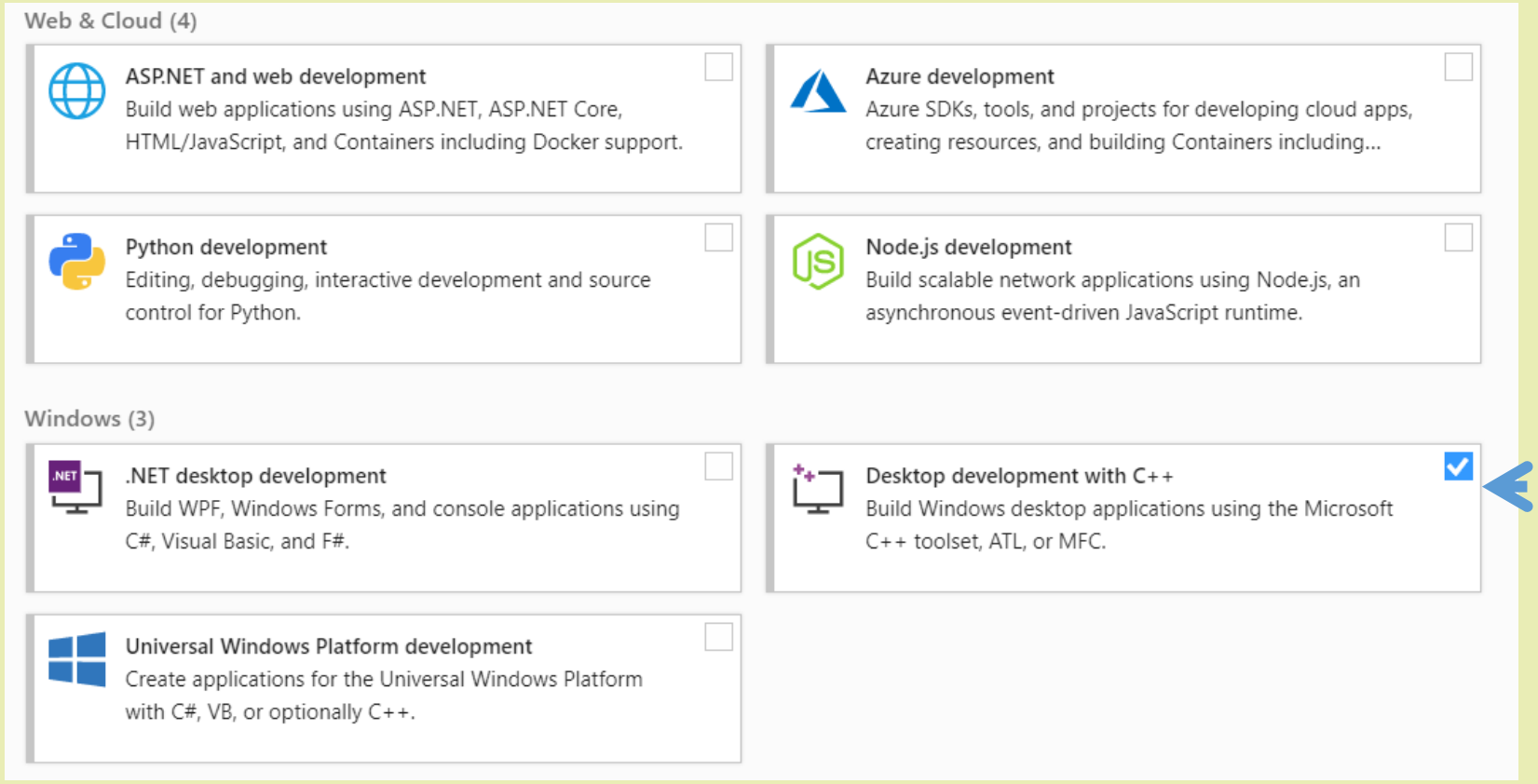
https://visualstudio.microsoft.com/downloads/

- Visual studio is standardly used in the job market.
- Knowing it will boost your CV!!



Installing Visual Studio

- Download Visual Studio from https://visualstudio.microsoft.com/downloads/
- Download the community version, choose the follow option:



- Login within 30 days (only on your PC or laptop, not on Uni's computers).

This option

Basic rules of a C++ code

- Go to L1D1.cpp
- Every programme consists of a set of statements.
- Every line has to end with a semicolon;
- The code consists of one or more blocks (a block starts with { and ends with }).
- Every programme must have one and only one `main` block/function.
- For comments, // is used to make a line of comment, while /* and */ are used to mark the beginning and the end of a block of comments.

Commenting codes

- Comments are explanatory statements written by the developer in the code to make it more readable by other developers.
- Comments are very important particularly in projects with multiple developers.

Practical notes:

- Follow the comment convention when working on a project with multiple developers.
- Convention for this module: state the functionality of every code block in a comment, in addition to commenting any statement with a new function.

Summary

- An overview of C++
- Difference between complied and interpreted languages.
- The role of compliers and how executable files are created.
- Introduction to Visual Studio.
- Basic rules of C++ coding.