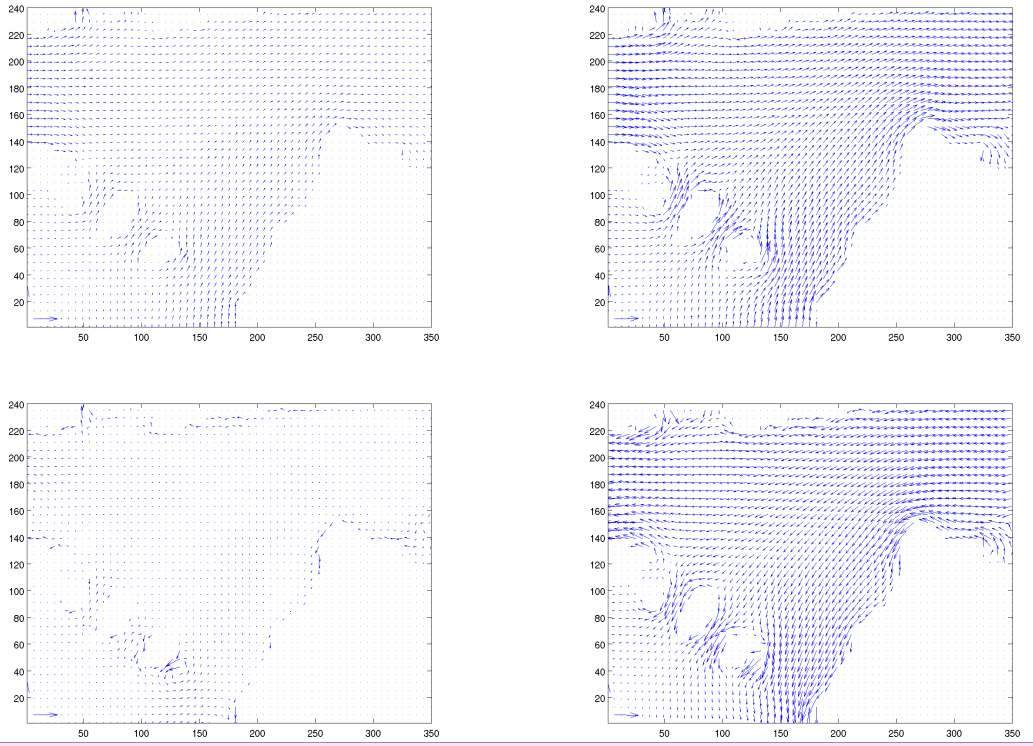
Annotated slides from 4pm class.

CS319: Scientific Computing

Introduction to CS319

Dr Niall Madden

Week 1: 10 January, 2024



	Mon	Tue	Wed	Thu	Fri
9 – 10			✓		
10 – 11					
11 – 12					
12 – 1					
1 – 2					
2 – 3					
3 – 4					
4 – 5			✓		

We need to find some other times for labs.

Please send your time-table to Niall (Niall.Madden@UniversityOfGalway.ie) ASAP, preferably today.

Scientific Computing

Dianne O'Leary describes a **computational scientist** as someone whose focus is the intelligent development and use of software to analyse mathematical models.

These models arise from problems formulated by scientists and engineering. Solutions/models can then be constructed using statistics and mathematics. Numerical methods are then employed to design algorithms for extracting useful information from the models.

start here 4 p.m.

Scientific Computing

In scientific computing, we are interested in the **correct**, **reliable** and **efficient** implementation of these algorithms. This requires knowledge of how computers work, and particularly how numbers are represented and stored.

History has shown that mistakes can be very, very costly.

Source: Wikipedia

For us, the major topics of CS319 will be

- **Computer representation of numbers**, as well as more complicated objects, such as vectors and matrices.
- Visualisation of functions and data
- **Root-finding and optimisation**
- Data fitting, and least squares;
- Efficiency and complexity of algorithms (from an experimental/applied view).
- Solving linear systems by direct and iterative methods;
- Representation and visualisation of graphs and networks

A first example

In the first few weeks of the module, we'll use C++.
But first we are going to study, without too much explanation, how to implement a simple algorithm in each of these three languages, and estimate their (in)efficiency.
The problem is to write some code that will sum all the elements in a list, and report how long it took.
In each case, we'll take the simplest possible approach, and ignore that each of these languages has (somewhat built-in) functions to do this.

Also: we want to verify we get the correct answer.

TimeAlg1.py

```
# Sum the elements of a list in Python
import time

N=10**7      # N=10^n
A = [1]*N
start = time.time()
s1=0;
for i in range(len(A)):
    s1+=A[i]
t1 = time.time() - start
print(f"N={N:6.0e}, error={s1-N}, time(s)={t1:6.2f}")
```

TimeAlg1.m

```
% Sum the elements of a link in MATLAB/Octave
N = 10^6;    % N=10^n
A = ones(1,N);
start=tic;
s1 = 0;
for i=1:length(A)
   s1=s1+A(i);
end
t1=toc(start);
fprintf('N=%8.2e, error=%d, time(s)=%8.4f\n',...
         N, s1-N, t1)
```

TimeAlg1.cpp

```
#include <iostream>
#include <time.h>
#include <math.h>
int main() {
  int N=pow(10,6); // N=10^n
  double *A = new double [N];
  for (int i=0; i<N; i++)
    A[i]=1.0;
  clock_t start=clock();
  double s1=0;
  for (int i=0; i<N; i++)
    s1+=A[i];
  double num_clocks = (double)(clock()-start);
  double t1 = num_clocks/CLOCKS_PER_SEC;
  std::cout << "N=10^" << log10(N)
            << ", error=" << s1-N
            << ", time(s)" << t1 << std::endl;
  return(0);
}
```

Like import in Python

Introduction to C++

Adapted from Wikipedia

C++ (pronounced "C plus plus") high-level, general-purpose programming language created by Bjarne Stroustrup. First released in 1985 as an extension of C programming language, but as an object-oriented language.

In the TIOBE Index for Jan 2024, C++ is ranked as the 3rd most popular language, behind (in order) Python, and C, and just ahead of Java and C#.

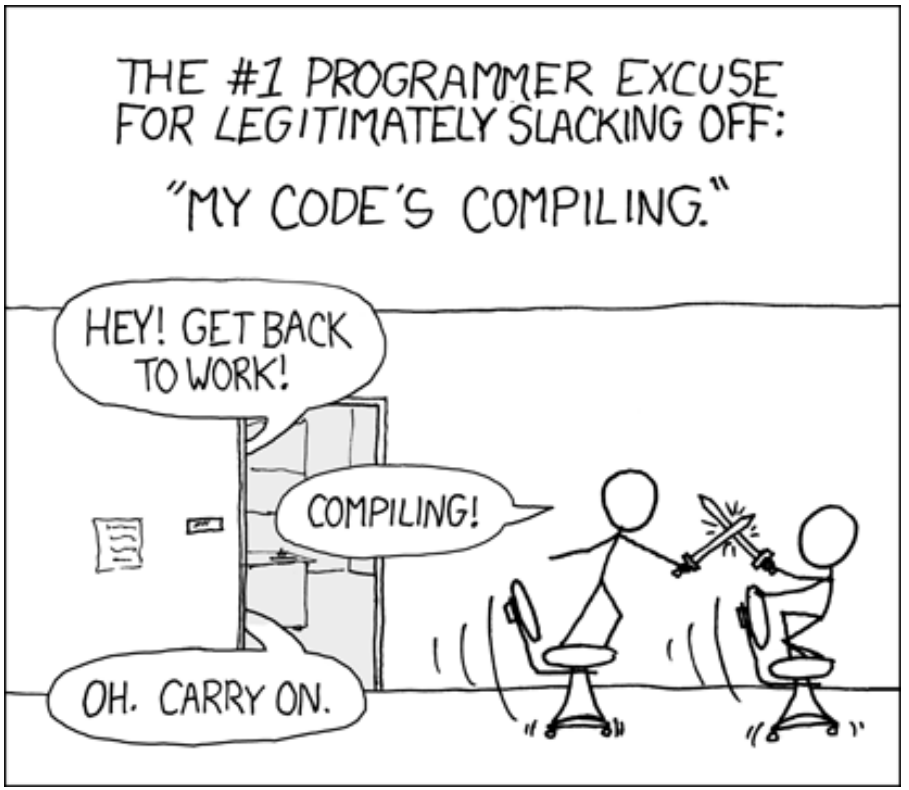
Introduction to C++

The main difference between C++ and (say) Python is the C++ is exclusively a **compiled** language (and not interpreted):

- Write the code
- Compile the code into an executable file.
- Run the executable.

C++ does not have a interactive REPL.

(Don't worry - this will be funny by March).



Source: https://xkcd.com/303

If you have don't have a C/C++ compiler installed on your computer, I suggest using one of

- `Code::blocks`
- Bloodshed's `Dev-C++`
- Xcode (for maxOS)

Both are freely available to install on your own device.
For Windows, I suggest installing `codeblocks-20.03mingw-setup.exe`, since this includes compilers as well as the IDE.
To get started, try an online compiler such as `https://www.onlinegdb.com` or `http://cpp.sh`

The C++ topics we'll cover are

1. From Python to C++: input and output, data types and variable declarations, arithmetic, loops, Flow of control (`if` statements), conditionals, and functions.
2. Arrays, pointers, strings, and dynamic memory allocation.
3. File management and data streams.

(Classes and objects will be mentioned in passing).

Finish here