Introduction

- What is a computer, and Computer Science?
- What is software?
- What is a programming language?
Computers
And Computing Science
Characteristics of a Computer

- Computers come in many shapes and sizes
  - General Computers: laptops, PC, etc.
  - Special purpose: anti-lock brakes, toasters
- Characteristics of all computers...
  - Are very fast at.. (+, -, *, /)
  - Represent data..
  - Have large main memory to store and retrieve data
  - Accept input and produce output
  - Can be.. because programs are stored in main memory (Von Neumann architecture)
How Smart Are Computers?

- Computers are very good at doing things that we find difficult to do quickly.
- But does that mean that computers are generally “smarter” than people?
Computers vs The Brain

- **Alienware PC**
  - Uses Intel Core i9
  - $\approx 500,000$ MIPS
- **Lots of memory!**
  - 64 GB of RAM
  - 4 TBs of storage

- **Human brain**
  - Processing power estimated at $100,000,000$ MIPS
  - Memory estimated at 100 TB
What is Computer Science?

- It is the

  How do you
  pick your
  next move?

How could you
describe this game
board in words and
numbers?

and
What is Computer Science?

- It is the study of **algorithms** and **data structures** including:
  - formal properties
  - hardware
  - programming languages
  - creating application
Software

Image by Pixabay on Pexels
Where Can We Find Computers?

- Computer Systems are ubiquitous
  - Telecommunications
  - Medicine
  - Information and Research
  - Entertainment
  - Finance
  - Transportation
  - ...
- Many such systems are critical

“I think there is a world market for maybe five computers”
IBM chairman, 1943
Hardware and Software

- **Hardware** refers to computer equipment
  - Central Processing Unit (CPU)
  - Secondary memory
  - Input devices
  - Output devices
- **Software** refers to the programs that..
Software

- **What is software?**
  - A set of instructions for a computer.
  - Programming:...

- **Why is programming (considered) hard?**
  - Because we want to **solve hard problems**
    - Usually things we can’t easily do by hand
  - And because **computers are fundamentally stupid**
Writing Software

- Software tells a computer how to solve a problem
  - **Human Example:** Giving friend directions on how to find you in a movie theatre?
    - What does computer need?
- But, remember, computers are *stupid*
  - They can’t deal with *ambiguity*
  - Instructions must be precisely defined in perfect grammar
Devising a Process
Imagine you are an aid worker in a small city during an earthquake.

- Most of the town is destroyed, but the open-air stadium is still standing.
- Survivors are being directed to the stadium which is big enough to hold all the survivors.

In a group of 3-4, you must devise a protocol by which survivors may be reunited with their nuclear family before they are able to move to some red-cross tents.

- Aid workers have a bull-horn to talk to many people at once.
- Also have pen/paper, and other resources. No cell phones.
- Think about handling many people efficiently.

Algorithm:

- May be in English: Write the sum of 5 plus 10
- May be in Pseudocode: print 5+10
- May be in C++: cout << 5 + 10;

Program: An implementation of.. for the computer to execute.

C++ programs are very formal
- They must be written using..
- They must be..
Euclid’s Algorithm

Input
positive integers $a$ and $b$

Output
the greatest common divisor (GCD) of $a$ and $b$

Algorithm
Repeat until $a$ and $b$ are the same value:
  if $a$ is greater than $b$:
    set $a$ to $a - b$
  else:
    set $b$ to $b - a$

Return $a$ as the answer

Try it when $a = 91$ and $b = 65$
Euclid Example

Repeat until $a$ and $b$ are the same value:
  
  if $a$ is greater than $b$:
    
    set $a$ to $a - b$
  
  else:
    set $b$ to $b - a$

Return $a$ as the answer

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>91</td>
<td>65</td>
</tr>
<tr>
<td>26</td>
<td>39</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
</tr>
</tbody>
</table>

Result
Euclid’s Algorithm in C++

```cpp
int main()
{
    cout << "Calculates the GCD of two integers\n";
    cout << "Enter the first integer: ";
    int a = 0;
    cin >> a;

    cout << "Enter the second integer: ";
    int b = 0;
    cin >> b;

    while (a != b) {
        if (a > b) {
            a = a - b;
        } else {
            b = b - a;
        }
    }

    cout << "GCD = " << a << "\n";
}
```
Properties of an Algorithm

- Every step is unambiguous
  - You must specify exactly what to do.

- Input and output are clearly defined
  - Bad: “Add up some values”
    - What type of values? How many?
    - What to do with the answer?

- Must be executable in finite amount of time
  - Must finish before the end of time.
Developing Programs

- Analysis
  - What is the problem?
- Design
  - What is the solution?
- Programming
  - Write the program
- Testing
  - Make sure the program works
Programming Goals

- Correct
- Reliable
- Well designed
- Affordable
- Maintainable
Types of Languages

- A program is written using a...

- There are different kinds of these:
  - Machine language
  - Assembly language
  - High level languages
    - C, C++, JavaScript, Python, Java, Fortran, Rust, ...
### Machine Language

- **Machine language can be processed directly by a computer.**
- **A program is a sequence of instructions.**
  - Each instruction code is...
  - Each number represented in binary.
- **Machine languages are very hard for humans to.**

Part of iTunes --> (Trust me)
Assembly Language

- Assembly languages are.
- Assembly language directly translates to machine code
  - Commands are at a..
  - Finding a '1' in some data can take many lines.
    (see example on right)

```assembly
.data
arr: .word 2, 2, 3, 4, 5, 6, 7, 8, 1, 5, 8

.text
main:
  la  $s5, arr
  addi $s1, $zero, 1
  add  $s3, $zero, $zero

loopstart:
  sll  $t0, $s3, 2
  add  $t0, $t0, $s5
  lw   $t1, 0($t0)
  beq  $t1, $s1, loopend

  addi $s3, $s3, 1
  j     loopstart

loopend:
  addi $t2, $s3, 0
```
High Level Languages

- High level languages are much easier to...

- C++ is a high level programming language
  - Compiles into machine code before executed

- Programming languages are formal and lack the richness of human languages
  - If a program is nearly, but not quite syntactically correct then it will..
  - The compiler will not “figure it out”
Brief History of C++

- C create in 1972 by Dennis Ritchie of Bell Labs
  - Use for writing and maintaining Unix (the OS).
  - Popular for low level system programs.
- C++ created in 1980's by Bjarne Stroustrup at AT&T.
  - Includes most of C as a subset of the language.
  - C++ is often “cleaner” than C (less error prone).
  - C++ supports Object Oriented Programming (CMPT 135).
  - Updated often: ...C++03, C++11, C++14, C++17, C++20...
- (There is no C+ language!)
Why C++?

- Generates efficient programs
  - Compact and run quickly (popular for games/OS/etc)
- Portable
  - Programs from one system can be run with little modifications on other systems (often...)
  - Useful for embedded systems
- Flexible
  - Allows programmers a lot of control
- What we’ll cover has some similarity to parts of C, so if you need to work in plain C it should be familiar
Computers are very fast, but not intelligent.

**Algorithm**: a set of instructions for solving a problem.

**Software**: a set of instructions for a computer.

**Programming Languages**:
- Higher level languages easier to read and write.