What is Computer Science?

Computer science is the study of automating processes, for example, calculating the square root of a number.

A computer scientist is someone who specializes in the theory of computation and the design of computational systems.

Important concepts at the heart of computer science:
- Data
- Algorithms
- Programming
- Abstraction

Data

We live in an era of data overload.

For example:
- Google, when searched for “Alan Turing”, finds approximately 750 thousand pages, ranked in order of estimated relevance and usefulness.
- Facebook has approximately 1.7 billion active users generating billions of comments and “Likes” each day.
- GenBank, a national database of DNA sequences, has over 1 billion entries.
- The Large Hadron Collider (LHC) at CERN produces collision data at the rate of 25 petabytes (1 petabyte = 10^{15} bytes) per year.

All these data would be junk without ideas and tools from computer science.
Algorithms

An algorithm is a sequence of steps for carrying out a task, such as ranking web pages in Google, or finding closely related genes in GenBank

An algorithm to estimate $\pi \approx 3.14159$

1. Draw a square that is 2 by 2 feet
2. Inscribe a circle of radius 1 foot (diameter 2 feet) inside the square — note that the ratio of the area of the circle to the area of the square is $\pi/4$
3. Grab a bucket of $n$ darts, move away from the dartboard, and put on a blindfold
4. Throw each dart randomly, but assume that your throwing skills ensure that it will land somewhere on the square dartboard
5. Record whether or not the dart landed inside the circle
6. Once you have thrown all $n$ darts, divide the number $c$ that landed inside the circle by $n$ and multiply by 4 to obtain an estimate for $\pi$ — for example, if $n = 10,000$ and $c = 7,869$, then $\pi \approx 3.1476$

Programming

A program is a list of instructions used to control the behavior of a computer

A programming language is a formal language designed to communicate those instructions to the computer

Types of programming languages

- Machine languages – interpreted directly in hardware
- Assembly languages – thin wrappers over corresponding machine languages
- High-level languages – machine independent
- System languages – designed for writing low-level tasks, like memory and process management
- Scripting languages – generally extremely high-level and powerful
- Domain-specific languages – used in highly special-purpose areas only
- Visual languages – non-text based
- Esoteric languages – mostly of academic interest

In this course, you will learn to program in Python

Abstraction

Abstraction is the idea that when designing one part of a system, we can ignore the inessential details of other parts, provided we have a high-level understanding of what they do

In the design of software systems, abstraction ensures that many people can contribute to a project without everyone needing to understand everything