Lecture 1: What is computation?

-Problem Solving
-Knowledge of Concepts
-Programming Skill

TOPICS

- represent knowledge with data structures
- iteration and recursion as computational metaphors
- abstraction of procedures and data types
- organize and modularize systems using object classes and methods
- different classes of algorithms, searching and sorting
- complexity of algorithms

A Computer performs calculations and remembers results fundamentally. Computers are not magical. They just do what we tell them to do. There are 2 types of knowledge which are declarative knowledge and imperative knowledge. Declarative knowledge is statements of fact. Imperative knowledge is a recipe or as we say “how-to”.

Example is written in python and IDE is anaconda.

There is a recipe for programming like sequence of simple steps followed by flow of control process that specifies when each step is executed and also the latest is a means of determining when to stop. Recipe is for an algorithm. Historically there were 2 different types of computers: fixed program computer, stored program computer. Fixed program computers only do one thing what it is tasked to do. For example, a calculator is one great example of a fixed program computer. It only does addition, subtraction, multiplication, division and it can not go on the internet or send email with it. And in order to create a machine that does another thing then you would have to create another fixed-program computer. That is when stored program computers came into play. A machine could store and execute a sequence of instructions. Which allows us to do different tasks in the same machine. That is the computer we use. Basic machine architecture has memory at the top and arithmetic logic unit below it and input and output at the bottom of it. Memory contains a bunch of data and sequence of instructions. And to interact we need a control unit. And when we load a sequence of instructions the program counter starts at the first sequence and passes it to the arithmetic logic unit and it does primitive operations and when it is done it passes data back to memory and increases program counter value by 1. After finishing the latest sequence of instructions it outputs the result. Interpreter(special program) executes each instruction in order. We can compute anything using 6 primitives and Alan Turing proved it. Also anything computable in one language is computable in any other programming language.

Expressions are a set of primitive operations. Syntactically valid, static and semantically correct. Syntactically errors are caught easily in python. Static semantics errors also can be caught in Python by adding some decisions. Python programs are sequences of definitions and commands. Its definitions evaluated and commands executed by Python interpreter.

In python, everything is an object. Programs manipulate data objects. All objects have a type that defines the kinds of things programs can do to them. There are 2 types of objects:

- scalar(can not be subdivided)
- non-scalar(have internal structure that can be accessed)

Scalar objects type:

- int
- float
- bool
- NoneType

Non scalar objects type:

- List etc

Sum, difference and product operations are simple operations because its results type will depend on operator type. However, division is different because its result will always be a float. There are also other operations like remainder and power.