AMA LEARNING

Algorithms: Notes

In this lesson:

- Abstraction
- Procedural Abstraction
- Algorithms
- Algorithm Vocabulary
- Types of Algorithms
- Efficiency of Algorithms
- Measuring Efficiency
- Solving Problems with Algorithms
- Searches

Abstraction: **Represents complex idea with something simple** (this is the main idea of Computer Science Principles)

- Using a variable name to represent a concept
- Using a list to represent a collection of items as one list
- Using a function that someone else wrote but that you can use without understanding the code

Procedural Abstraction: process/procedure

- Provides a name for a process and **allows a procedure (function) to be used only knowing what it does, not how** (e.g. finding the maximum number in a list by coding)
- Allows solution to large problem to be based on solutions of smaller problems
    - Helps improve code readability
- Code can be **reused**

Algorithms:

- **Finite set of instructions to accomplish a specific task**
- Can be expressed in natural language, pseudocode, and/or programming language
- Can be **written in different ways, still accomplish some tasks**
- Algorithms that appear similar yield different side effects/results
- Different algorithms can be developed or used to solve the same problem
- Can be created from…
    - Idea
    - Modifying existing algorithms
    - Combining existing algorithms
    - Constructed using combinations of…
        - Sequencing (step by step reading of code in order)

- Selection (conditionals)
- Iteration (loops)

Algorithm Vocabulary:

- To execute (**execution**) a program = run program
- **Documentation** - add comments to program
- **Procedure** (function) interrupts sequential execution of a program
    - Function must be executed before program continues
    - Functions end when last statement/return statement is made

Types of Algorithms:

- **Sequential Algorithm:** executed from start to finish on one computer
- **Parallel Algorithm:** Executes instructions at same time of different processing devices and them combine all the individual outputs to produce a final result
    - Used when there are several lines of code
    - Less time consuming
- **Speed Up:** ratio that compares run time of sequential algorithm to parallel algorithm
    - Ex. Sequential: 30 mins; Parallel: 10 mins; Speed Up: 3:1 or 3

Efficiency of Algorithms:

- **Estimation of amount of computational resources used by the algorithm**
- Expressed as a function through size of input
- Determined through formal or mathematical reasoning
- Can be informally measured by determining the number of times a statement(s) execute
- Different correct algorithms for the same problems have different efficiencies

Measuring Efficiency:

- **Polynomial/cloer (constant, linear, square, cube) efficiencies run in a reasonable amount of time**
    - **Ex. 2n**
- Exponential/factorial efficiencies run in an unreasonable amount of time
- **Same problems can't be solved in reasonable amounts of time** because there is not efficient algorithm, so approximate solutions are found

Solving Problems with Algorithms:

- **Optimization Problem:** finding the "best"/fastest solution among many
- **Decidable Problem:** an algorithm written to find outputs for all inputs and find a solution
- **Undecidable Problem:** no algorithm will always work

- **Heuristic:** way to solve problem, but not “best” (fastest/cheapest), it is still a way to work

Searches:

- **Linear/Sequential Search:** you keep going through the list in order to find the solution (beginning to end)
- **Binary Search: starts in the middle of set of sorted data, eliminate half of data by dividing it by 2**
    - **More efficient than linear search**
- **Libraries:** contains procedures that may be used in creating new programs
    - Can come from existing code
    - Need documentation
    - This is easier because you don’t have to rewrite code