06/12/25 - Intro to Algorithms

Abdul Bari - Algorithms - Video 1

Algorithm	Program
Design Time	Implementation Time
Domain Knowledge	Programmer
Any language (english/math)	Programming Language (python/js/java)
Hardware and OS	Hardware and OS
Analyzing	Testing

An algorithm is a series of steps that we use to solve a certain problem.

06/12/25 - Abdul Bari - Video 1.1 - Priori v Posteriori

Priori Analysis	Posteriori Testing
1. Algorithm	1. Program
2. Independent of Language	2. Depends on the language
3. Hardware Independent	3. Depends on the hardware
4. Time and Space Function	4. Watch time and bytes

06/12/25 - Video 1.2 - Characteristics of Algorithms

1. Input
   - Algorithms can take zero or more inputs

2. Output
   - Algorithms must generate at least one output. If not, no point in even writing an algorithm.

3. Definiteness
   - Every statement written must be clear. Meaning if we humans cannot solve, why make a computer solve it?

4. Finiteness
   - Must have a finite set of statements. Meaning that our algorithm should stop at one point.

5. Effectiveness
   - Should not write unnecessary statements. Everything we write should somehow relate to the problem we're trying to solve.

In essence, every statement in an algorithm must make sense, stop at one point, and should relate to the overall problem we want to solve.

06/12/25 - Video 1.3 - Write and Analyze Algorithms

```
Algorithm Swap(a, b)
{
  temp = a;
  a = b;
  b = temp;
}
```

This is a basic algorithm that just swaps two numbers.

this all depends on the problem we want to solve + our requirements - time and space are the most notable

How to Analyze an Algorithm

1. Time - when dealing with algorithms, we want to make sure that they're fast. We want all our algorithms to be time efficient. The less time it takes, the better. We represent time as functions like O(1) or O(n).

2. Space - this refers to much memory (space) our algorithm takes when it runs.

3. Network - how much data is transferred

4. Power - how much power our algorithm is consuming

5. CPU Registers - how many CPU registers is our algorithm taking

06/12/25 - Video 1.3 Continued

Algorithm Swap(a, b)

```
{
  temp = a;   ——— 1
  a = b;      ——— 1
  b = temp;   ——— 1
}
```

P(n) = 3

Time: every simple statement in an algorithm takes one unit of time. Simple refers to a statement, not nested, just one simple line.

Space: to find space here, we count the number of variables that we have, and that becomes our answer.

* So if we find our time or space complexity to be a constant value such as 3, 30, 300, 9000, etc, we will always officially represent it as O(1).

* When dealing with time, in reality, we denote it using big-O notation, O(1), O(n)

temp, a, and b are variables in our example, thus S(n) = 3

And now b/c both of our complexities, time and space are constant: we can represent it officially as O(1).