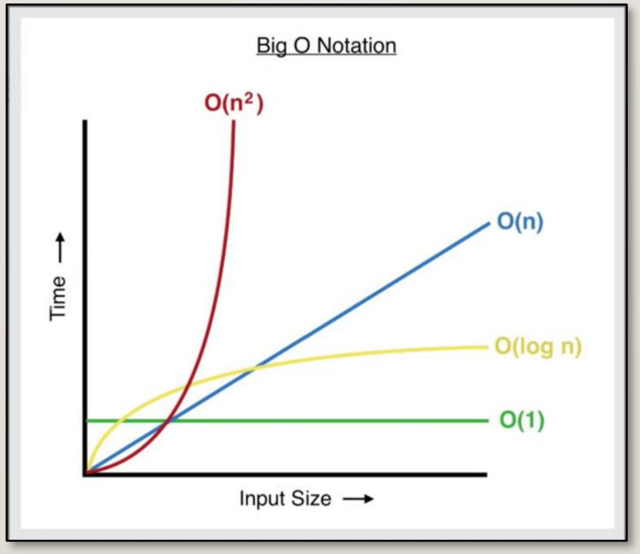
ANALYSIS OF ALGORITHMS

LECTURE 1 : BEST, WORST AND AVERAGE CASE

CASE ANALYSIS

- What are we trying to do here?
  - Want to relate the problem size to the time taken to solve the problem
- Important : We do not control n, the analysis must return a function of n



CASE ANALYSIS

- What does it mean to have multiple cases? Note that this refers to some characteristic of the input that affects the running of our algorithm.
    - B(n) - The best configuration that could have happened to us
    - W(n) - Why is this happening?!?
    - A(n) - Can we work out how our algorithm does on average?
- Example: Linear Search

CASE ANALYSIS : LINEAR SEARCH

- What is the complexity of linear search in the best case?
  - Let's think about that... If the array looks like the one below, what's the best thing to be searching for? How long will it take?

17	23	5	8	1	2

  - What about the worst thing? How long will that take?
  - Here's something a little more difficult – How long does it take on average?

CASE ANALYSIS : LINEAR SEARCH

- Average case?
  - Must first decide what we consider important. Is it the average over all cases?
  - Let's try that…
  - Consider the case where the array has exactly two elements

23	7

  - There's two things that could happen to us - the item is the first one, or it's the second one
  - In the first case, it will take 1 operation, and in the second case it will take 2 operations
  - So on average, it will take 1.5 operations $= \frac{(1+2)}{2}$

CASE ANALYSIS : LINEAR SEARCH

- Average case?
  - Consider the case where the array has exactly three elements

15	6	12

  - Now there's three things that can happen, which will take 1, 2 and 3 operations
  - So on average, it will take 2 operations $= \frac{(1+2+3)}{3}$
  - So basically, we sum up the cost of each case and divide by the number of cases

CASE ANALYSIS : LINEAR SEARCH

- Average case?
    - So for an array of size n, this will be $\frac{1+2+3+\cdots+n}{n}$
    - This is $\frac{\sum_{i=1}^{n} i}{n}$
    - Now, $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
    - So $\frac{\sum_{i=1}^{n} i}{n} = \frac{n(n+1)}{2n} = \frac{n+1}{2}$
- You can already see that the average case is a lot harder than the best or worst. And it gets even worse. In this case, we made the simplifying assumption that the item is definitely in the array, and that it is equally likely to be in each position. Without these it could be a lot harder

SOME THOUGHTS

- These cases deal with different configurations of input. Note that this is not the same as the upper bound and lower bound that we'll discuss in the next lecture