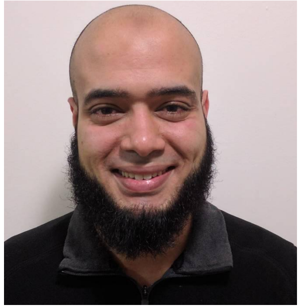
Data Structures

Why Data Structures

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



2 Common Questions

- **Why do we <u>need</u> data structures?**

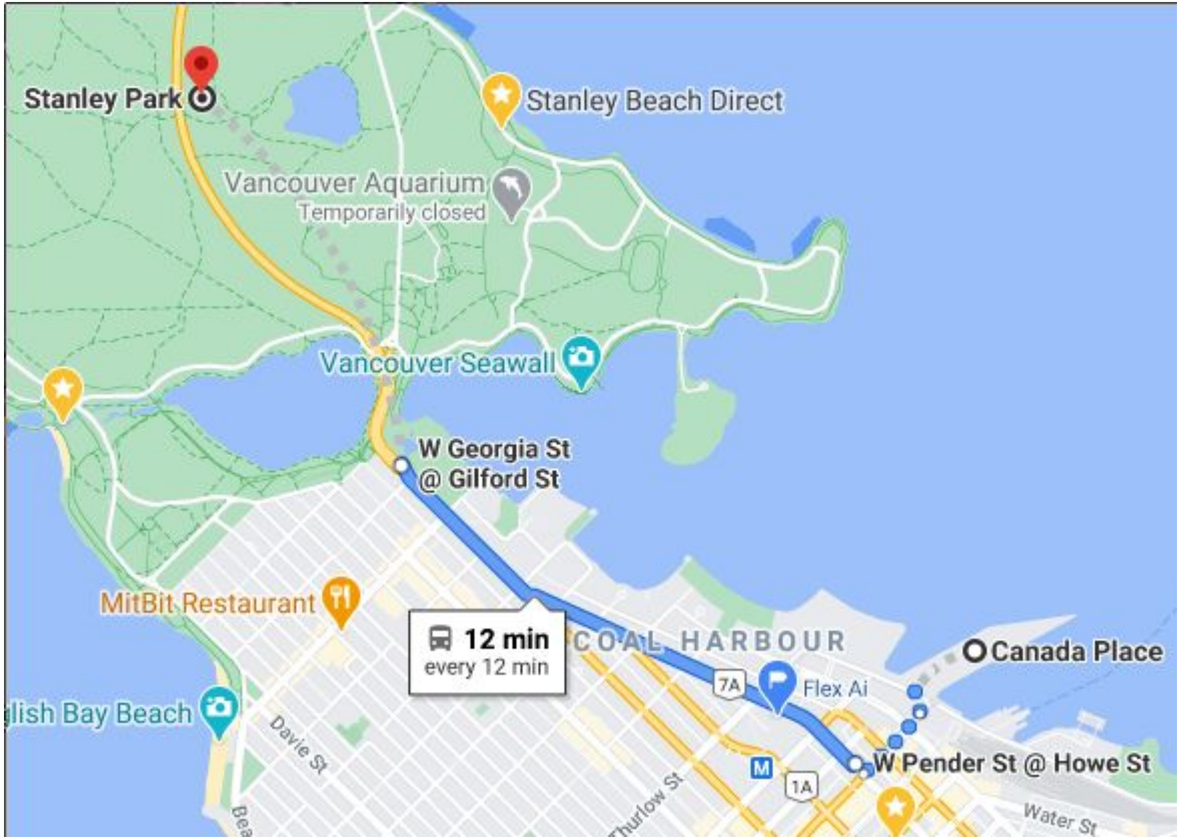
- **Why do we <u>study</u> data structures if we have built-in ones?**

The need

- Last lecture should implicitly be an answer that
- Primitive arrays are not enough
- We face many cases where we need our **own** user-defined data structures
    - What if want to make **correspondence** between a string and some object
        - E.g. data[“mostafa”] = someobject
        - But array is only based on integers? We need something that has greater flexibility!
            - This is where **map or hash-table** can help us!
        - E.g. we have 1 Billion word text from articles and want to search for a specific word
            - Array of words is very slow.
            - But Trie or Suffix Tree data structures are much faster!
- Let’s see an industrial case: Google maps

Google Maps

- With maps, we can go from a place to another
- We need **data structure** to represent **points and streets**!
  - We have a lot of data!
  - Constraints: time/date/car
- We need **efficient** functions to find **optimal paths**



Data Organization Perspective

- From a purpose to another, we may arrange the same **data** in different ways
  - There are many ways to implement a data structure
- When we have much data (Millions of users on facebook), things become much more complex and critical!
  - E.g. Search the engines for scientific or social purposes
  - E.g. If a storm hits specific list of locations, how many homes will face a power outage?
  - E.g. We are in a war, and want to destroy the **minimum number of bridges** in a city to disconnect 2 points of our enemy? Rockets are expensive!

Data Structure Efficiency

- Assume we have N (10000) employees
- Is a loop over these N employees as fast as 3 nested loops?
    - No, it seems like 10000 operations vs 10^12 operations!
    - Seems the first is efficient, but the 2nd is not!
- So how to measure the efficiency of a function?
    - The **complexity** (asymptotic) **analysis** in the algorithms field answers that.
    - The efficiency can be for **time and memory (space)**
    - For the same problem, we may arrange data in a way that is so fast in computations, but another is much memory efficient
        - You maybe lucky and your DS is both time and memory efficient!
    - On a mobile: you may target a memory efficient approach
    - On real-time service: you may target a time efficient approach

Why not just use built-in DS?

- Using built-in data structures as a black box without understanding their details is big risk in real projects
  - Typically you won’t realize the time/memory order
  - Typically you will use them **improperly** as you don’t recognize their differences
- More critical
  - In practice, you may need to design your own data structures, but you lack the skill
  - E.g. what are the different data organization **perspectives** that we may use?
    - What are the pros/cons of them?
    - The time/memory differences?
    - The tradeoffs?
- Side effect
  - Your thinking skills are improved. This is good for problem-solving & algorithms course

Use DS first & learn later approach

- You may learn **using built-in** data structures first, before learning them
  - What not How
- Many built-in data structures exist
- It is good during a programming course to use them!
  - See my **Mastering 4 Critical Skills in C++17 course**
  - You can even solve many problem-solving problems on them (See the course)
    - Mainly: Black-box solving
- When it comes to learning later, things are much easier to guess
  - Even you can implement some basic data structures by yourself!
- Trap: Many students after learn the usage, the decide to skip studying!
  - Don’t be that guy

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."