While folks are joining

Get you laptops ready and login to www.crio.do.
We will be coding away in the session!

DSA-1

Session 7

What’s for this session?

- Collections/Libraries
  - Arrays
- Hash and Heap
  - Introduction
  - Library methods
- Problems
  - Sorting an Array
  - Perform repeated subtractions in an array
  - Find the most repeated temperature

What are Collections?

- Collections are libraries with implementations of DS with commonly needed methods
  - Arrays, Map (Hash Map), Set (Hash Set), Priority Queue/Heap, Stack, Queue
- Import these libraries in your program and use the library methods
- Available in most languages
  - C++ STL
  - Java Collections
  - Python libraries
  - JavaScript modules

- You don’t need to implement the Data Structures from scratch while solving problems. Use available Collections.

Arrays - C++

- std::vector vs std::array

```
#include <iostream>
#include <vector>

using namespace std;

int main()
{
    vector<int> g1;

    for (int i = 1; i <= 5; i++)
        g1.push_back(i);

    cout << "Output of begin and end: ";
    for (auto i = g1.begin(); i != g1.end(); ++i)
        cout << *i << " ";

    cout << "\nOutput of cbegin and cend: ";
    for (auto i = g1.cbegin(); i != g1.cend(); ++i)
        cout << *i << " ";

    cout << "\nOutput of rbegin and rend: ";
    for (auto ir = g1.rbegin(); ir != g1.rend(); ++ir)
        cout << *ir << " ";

    cout << "\nOutput of crbegin and crend : ";
    for (auto ir = g1.crbegin(); ir != g1.crend(); ++ir)
        cout << *ir << " ";

    return 0;
}
```

- https://stackoverflow.com/questions/4424579/stdvector-versus-stdarray-in-c
- https://www.geeksforgeeks.org/vector-in-cpp-stl/

Arrays - JavaScript

```
const salad = ['🍅', '🍄', '🌳', '🥒', '🫑', '🥕', '🥑'];
const len = salad.length;
salad[len - 1]; // '🥑'
salad[len - 3]; // '🫑'
```

```
const salad = ['🍅', '🍄', '🌳', '🥒', '🫑', '🥕', '🥑'];

for(let i=0; i<salad.length; i++) {
  console.log(`Element at index ${i} is ${salad[i]}`);
}
```

- https://javascript.info/array-methods

How to Approach Problems?

For any given problem, following these milestones will help you solve the problem systematically:

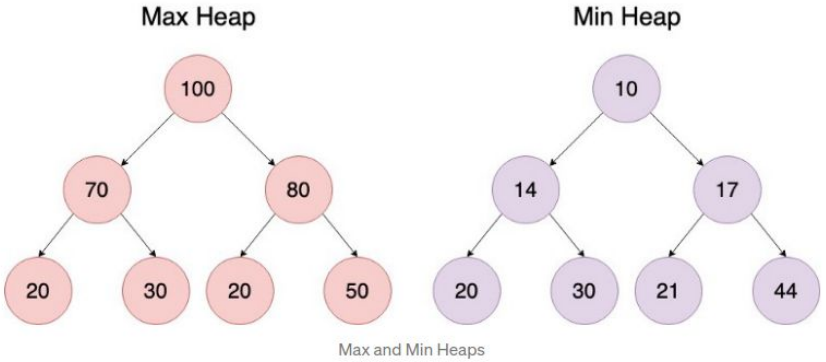
- **Milestone 1** - Understand the problem statement and confirm your understanding with some examples or test cases, including edge cases.
- **Milestone 2** - Think about approaches and select the best one you know. Explain your approach to a 10 year old. Write the pseudocode with function breakdown.
- **Milestone 3** - Expand pseudocode to code
- **Milestone 4** - Demonstrate that the solution works

Activity 1 - Sorting an Array

- How would you solve this problem, which needs sorting?

Heap

- Highest priority item at the top
- **Properties**
  - Based on the Binary Tree structure
  - Max Heap - largest item at top
  - Min Heap - smallest item at top
  - Not Sorted
  - **Lookup of highest priority item is O(1)**
  - Insert/Delete - expensive, maintain priority
  - Search - is expensive
- **Priority Queue** is implemented using Heap
- **When to use**
  - Quickly find smallest/largest item



Max and Min Heaps

Source: https://medium.com/swlh/data-structures-heaps-b039868a521b

Heap

- Applications
  - To quickly find the smallest and largest element from a collection of items or array.
  - In the implementation of the Priority queue in graph algorithms like Dijkstra's algorithm (shortest path), Prim's algorithm (minimum spanning tree) and Huffman encoding (data compression).
  - In order to overcome the Worst-Case Complexity of Quick Sort algorithm from O(n^2) to O(nlog(n)) in Heap Sort.
  - For finding the order in statistics.
  - Systems concerned with security and embedded system such as Linux Kernel uses Heap Sort because of the O(nlog(n)).
- Library methods In different languages
  - Refer to resources here

Note: Heap will be covered in more detail in a pack of its own, later on

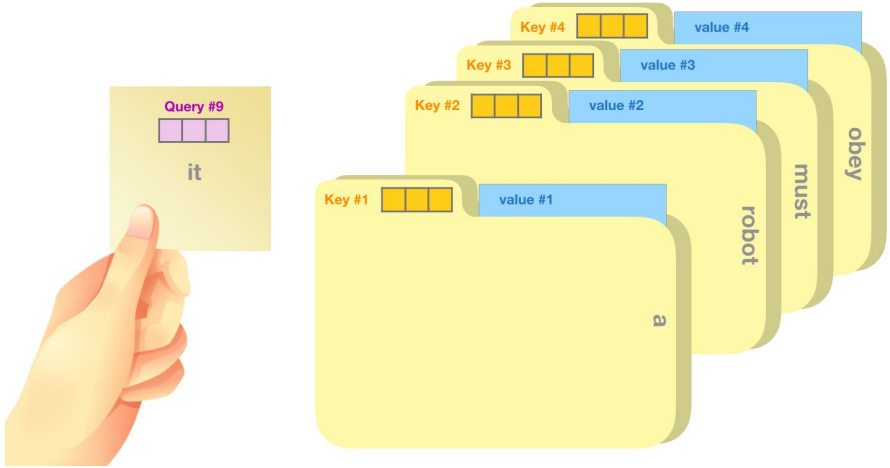
Heap - Frequently asked problems

- How to identify Heap problems?
  - If the problem is related to finding smallest or largest, you should consider Heap

- Frequently asked problems
  - Find the top K smallest/largest elements in a given array of integers
  - Find largest/smallest number
  - Find the Kth smallest/largest number

Activity 2 - Perform repeated subtractions in an Array

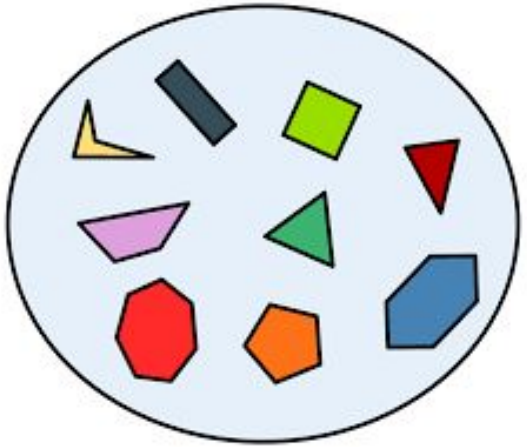
Map

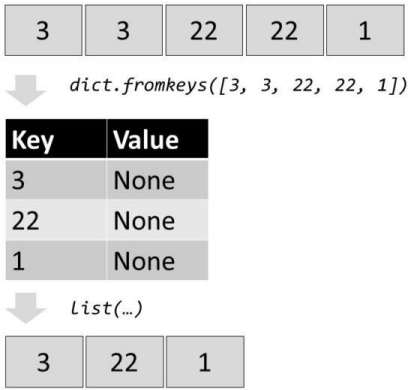


- **Key** = Unique Id. **Value** = Details for that key
  - Key and Value can be of different data types
- **Properties**
  - Keys CANNOT repeat
  - INSERT not possible if Key already exists
  - UPDATE Value for a Key
  - DELETE based on Key
  - ORDER of storage not important, POSITION doesn't matter
  - SEARCH based on Keys (generally, this is more optimal) or Values
  - SORT - may be sorted based on Key
- **When to use**
  - When we have a unique id based on which we want to pull up details
  - Keeping track of number of occurrences

Key Data Structures - Set

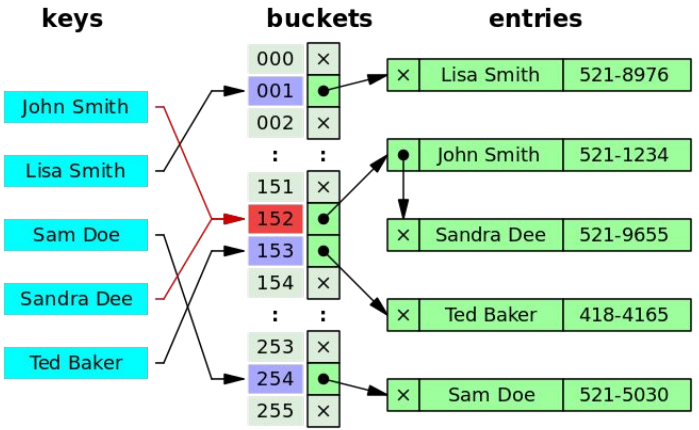
- **Unique Collection of elements**
  - Similar to Map, but **No Values, only Keys**
- **Properties**
  - Values cannot repeat
  - Size is important = Number of unique values
  - Insert not possible if it already exists
  - Insert and Delete - order of storage not important, position doesn't matter
  - Search - worst case will be as long as size
  - Sort - not supported
- **When to use**
  - When we need to keep track of presence/absence of unique elements





Hash Map and Hash Set

- Hash based implementation of Map is called **Hash Map**
- Defines how to come up with a **bucket** for the **given Key**
- **Properties**
  - Lookup/Insert/Delete by Key are **Constant time** operations
  - **Hashing algorithm** has to be optimal to avoid **collisions**. Worst case will behave like a list
  - ORDER is not guaranteed
- **Hash Set** - Hash based implementation of a Set



- Simple Hash function

```
int generateHash(int x) {
    return x % 16; // 16 buckets
}
```

Hash

- Applications
  - DB Table name and its fields
  - Library names and their methods (think import statements in code)
  - Caching
- Library methods In different languages
  - Refer to resources here

Note: Hash will be covered in more detail in a pack of its own, later on

Hash - Frequently asked problems

- How to identify Hash problems?
  - If you need to keep track of count of a particular element (Hash Map) or track occurrence of elements (Hash Set), you should consider Hash

- Frequently asked problems
  - Count the frequency of occurrence of letters or numbers
  - Find most/least frequently occurring word/letter/number/temperature etc.
  - Find a pair of numbers that add up to zero or a particular sum

Activity 3 - Most repeated temperature

Takeaways

- Familiarize yourself with Collections/Libraries for Data Structures
- Familiarize yourself with the methods in these Collections
- Leverage these to solve DSA problems

Questions?

Take home exercises

- Find the max element
- Most frequent character in a string

To be solved before the next session on Thursday, 7:30 PM

Feedback

Thank you for joining in today. We'd love to hear your thoughts and feedback.

https://bit.ly/dsa-nps

Thank you