Arrays and Linked List

Week 2

Required Activities

- Check Announcements regularly (every 2-3 days)
- Read DSA book: Chapter 2, 3, and 7
- Review supplemental materials (shows how to implement arrays and linked list)
- Complete and submit test video Assignment 1

Array

 data structure that can store a collection of items (elements) of the same data type. Each item is identified by index or key and depending on the programming language starts with 0 or 1. Elements are stored in conjunctive locations

0	1	2	3	4	5	6	7	8	9

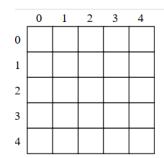
- In Python, Java, C++, and C# index starts at 0
- To access or store arrayName[index]
 - ex: data[4] = 5;
- Depending on prog language, may need to declare array and keep variable to track size of the array (e.g. C++)

Types of arrays

One-dimensional – linear array (list) with single index: name[index]

0	1	2	3	4	5	6	7	8	9

Two-dimensional – matrix or table where you need two indices: name[i1][i2]
Use nested loops to traverse.



• **Multi-dimensional** – need two or more indices. E.g. cube[x][y][z]

Array operations

- Access by index O(1) e.g. numbers[6]
- Traverse using loop such as for-loop O(n)
- Searching linear O(n) and binary O(log₂n)
- Insertion need to move elements O(n)
- Deletion need to move elements O(n)
- Merging for sorted elements O(n+m)
- Sorting depends on the sort algorithm

Array Advantage & Disadvantage

Advantage

- Easy to access and find elements at random using index
- Good cache locality because stored in block
- Each element is independent from others

Disadvantage

- Size fixed so need to know size in advance (memory allocated at compile)
- Expensive to insert and delete because have to move elements
- May be costly to increase size

Linked List

Collection of nodes where nodes contains data and field with address of next node. Nodes are of the same data type



- Types of linked lists:
 - Singly linked list can only traverse one way Doubly linked list can traverse either way

 - Circular linked list front and end connected
- Typical operations:
 - Initialize
 - Check if empty
 - Find size/length
 - Retrieve element
 - Insert new element
 - Remove element

Linked List Operations Cost

- Access traverse until find node O(n)
- Insert/Delete at beginning or end when known O(1)
- Insert/Delete at end when need to traverse O(n)
- Insert/Delete in middle search time + O(1)
- Sequential search O(n)

Linked List Advantage & Disadvantage

Advantage

- Insertion and deletion are fast
- Memory is allocated at runtime
- Stored anywhere in the memory (does not need to be block)
- Size easily increased

Disadvantage

- Has to be accessed sequentially (need to traverse list)
- Stored anywhere in the memory (no good caching)
- Extra space and dependency to connect elements (e.g. pointers)

Questions?

- Post in the discussions
- Send email to <u>RMcFadden@HarrisburgU.edu</u>
- Respond usually within 48hours