SUPER DAY 2 - Finesse CS-1134

May 7, 2022

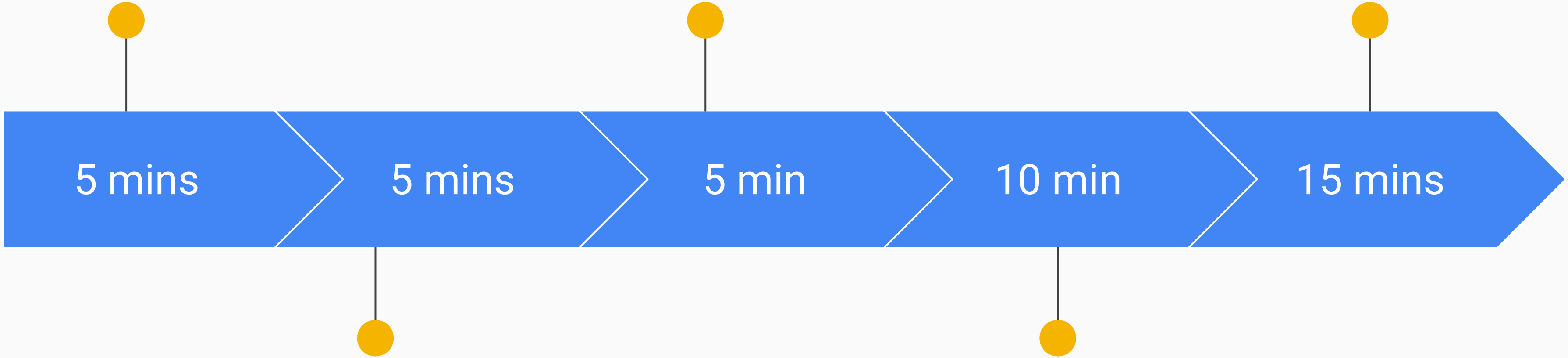
Please be sure to sign in at the front of the whiteboard.

Materials You’ll Need

- Pen
- Paper
- Computer

How This Will Work

- Presentation of topic
- Running drills
- Repeat Twice



5 mins

Explain the purposes of the data structure and its pros / cons

5 mins

How to recognize a problem

5 min

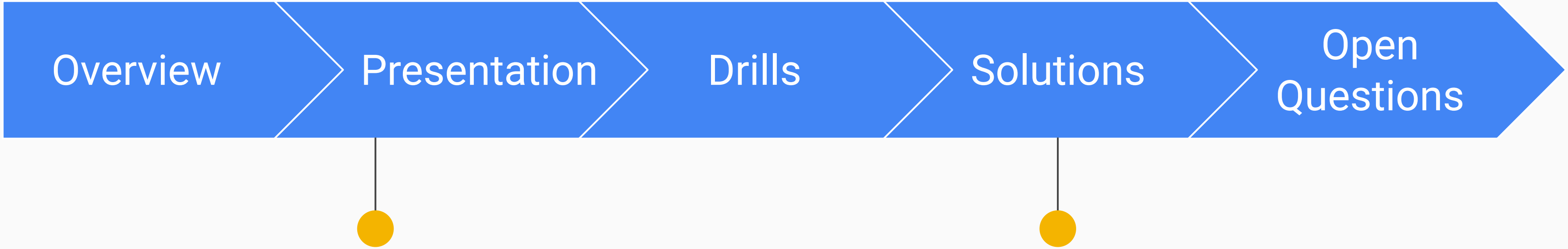
Explaining step-by-step approaches

10 min

Solving a problem live on the spot

15 mins

Space / time analysis



Overview

Presentation

Drills

Solutions

Open Questions

Repeated Twice:

1) Linked Lists
2) Hash Maps

How This Will Work

- Overview – 5min (11:00)
- Presentation 1 – 15min (11:05)
- Drills – 30min (11:20)
- Solutions – 20min (11:50)
- Break – 25min (12:15)
- Presentation 2 – 15min (12:40)
- Drills – 30min (12:55)
- Solutions – 20min (1:25)
- Open Questions – 15min (1:45)

Let’s Get To Know You

In a non-icebreaker way…

Let’s Start

Linked Lists

Purpose Recap?

- To keep order
- To keep order
- To keep order
- To keep order

Time / Space Complexity

- Space complexity: O(n), n being the amount of elements
- Time complexity:
  - Adding node to the end of a list - O(1)
  - Adding node to beginning - O(1)
  - Adding node to after a PROVIDED node - O(1)
  - Finding a node - O(n)

Time: *# of elements iterated over*
Space: *how much memory is being used*

Time / Space Complexity

- Space complexity: O(n), n being the amount of elements
- Time complexity:
  - Adding node to the end of a list - O(1)
  - Adding node to beginning - O(1)
  - Adding node to after a PROVIDED node - O(1)
  - Finding a node - O(n)

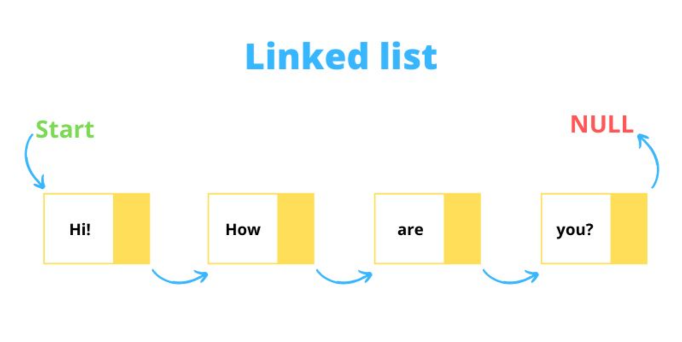
Common Combos With Linked Lists

Purpose	**Combo**
Tracking duplicates, checking/updating locations, keeping frequency counts	Linked Lists + Hash Map
Mixing in nodes with priority and non-priority	Linked Lists + Queues
Tracking specifically last items that were added	Linked Lists + Stacks

Combo 1: Linked Lists + Hash Maps

Purpose	**Combo**
Tracking duplicates, checking/updating locations, keeping frequency counts	Linked Lists + Hash Map

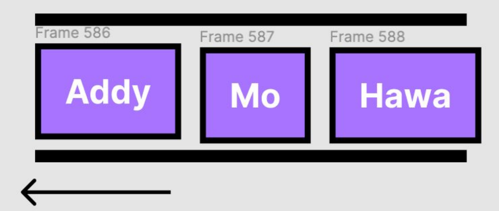
Node Data	**Node Address**
“Hi!”	*Node(“Hi!”) → 0x17345*
“How”	*Node(“How”) → 0x14598*
“are”	*Node(“are”) → 0x23467*
“you?”	*Node(“you?”) → 0x12384*



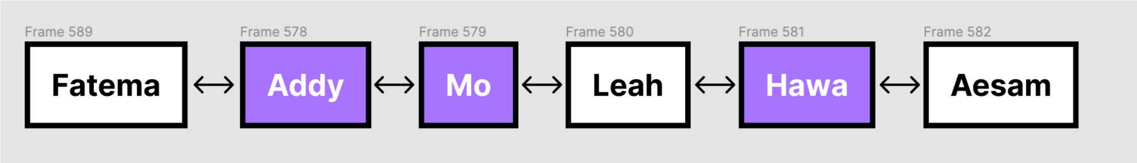
Combo 2: Linked Lists + Queues

Purpose	**Combo**
Mixing in nodes with priority and non-priority	Linked Lists + Queues

Queue of just NYU Tandon Students



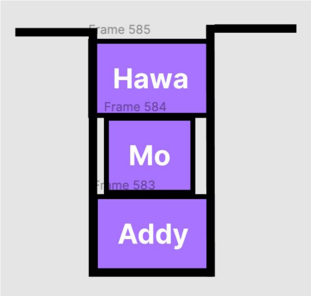
List of all nodes in order



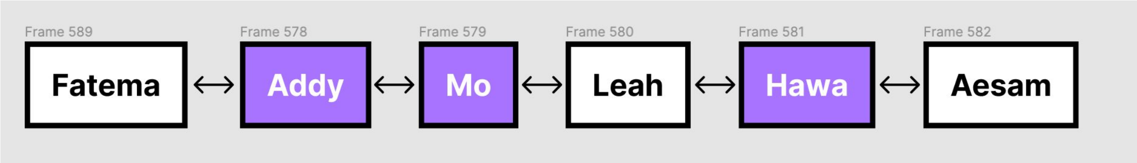
Combo 3: Linked Lists + Stack

Purpose	Combo
Tracking specifically last items that were added	Linked Lists + Stacks

Stack of last NYU tandon students to join



List of all nodes in order



Mistake #1: Trying to simulate in your head

Always provide visualizations for yourself – if one isn’t taught to you, make one for yourself

Literally before you code on your exam, **DRAW** out your solution. When your drawing works when testing it, *then* write it.

Speeds up time

Live Run

- Write a class that favors giving Broadway tickets to NYU Students
    - lottery = TicketWaitlist();
    - len(lottery) – *returns length of how many students in line*
    - lottery.addStudent(student_name, school_name) – *adds student to list of waitlist students*
    - lottery.handOutTicket() – *removes the first in line of a waitlist*
    - lottery.NYUHandOutTicket() – *removes the first NYU student in line from the waitlist*
    - lottery.displayWaitlist – *display all the students in waitlist in order*

Drill: Company Layoff

- Write a class of a company that lays off employees, prioritizing the most recently joined employees
  - company = Company(priority_dept) – *initializes company and the priority_debt to lay off*
  - len(company) – *returns number of employees currently working*
  - company.add_employee(employee_name, dept_name) – *hires new employee with their assigned department*
  - company.fire() – *fires the last employee added, return name of the fired employee*
  - company.fireFromPriorityDept() – *fires the last employee added that works in a priority department, return name of the fired employee*
  - company.displayEmployees() – *displays the list of all company employees*

We will be back
at 1:38

Update from Prof. Tal

You can only access the data members for the linked list data structures

Example:

you may use node.data, node.next, node.prev
Tree nodes are good too :)

Hash Maps

Purpose Recap?

- Associating some data with a *specific and unique key*
    - Means that there is each element has its own unique values

Time / Space Complexity

- Time:
    - Access – O(1)
- Space: O(k), k being how many keys you'd want to enter

Time: *# of elements iterated over*
Space: *how much memory is being used*

Time / Space Complexity

- Time:
  - Access – O(1)
- Space: O(k), k being how many keys you’d want to enter

Examples of Hash Map Usages

- **Keeping count** of students from different universities
  - How to do this: iterate over a given list of students, get what their school is, and either add it to the map, or access it and increment

University Name	**Count of Students**
Drexel University	6000
New York Univerisity	5000
University of Maryland	8000
Princeton University	2500

Examples of Hash Map Usages

- **Keeping locations addresses** of nodes
  - How to do this: when iterating over any collection, just add their key (most likely their data if unique) and add the node as the value

Key	**Node**
“Thriller”	Node(“Thriller”) → *0x17345*
“Bad”	Node(“Bad”) → *0x56486*
“Dangerous”	Node(“Dangerous”) → *0x17398*
“HIStory”	Node(“HIStory”) → *0x86940*

Examples of Hash Map Usages

- **Keeping miscellaneous information**
  - Below is list of albums

Key	**Value**
“Michael Jackson”	[“Thriller”, “Bad”, “Dangerous”]
“The Weeknd”	[“After Hours”, “Dawn FM”, “Trilogy”]
“Kanye West”	[“Donda”, “College Dropout”, “Yeezus”]
“Chance the Rapper”	[“10 Day”, “Acid Rap”, “Coloring Book”]

Mistake #1: Trying to simulate in your head

Always provide visualizations for yourself – if one isn't taught to you, make one for yourself

Literally before you code on your exam, **DRAW** out your solution. When your drawing works when testing it, *then* write it.

How would you solve the problem in real life with just your mind – no code?

Live Run

- Write a class of one’s album inventory
  - catalog = AlbumCatalog();
  - len(catalog) – *returns length of how many albums you have*
  - catalog.add_album(artist_name, album_title, genre) – *adds an artists album*
  - catalog.displayArtistAlbums(artist_name) – *shows all artist’s albums*
  - catalog.displayGenreFirst(genre)– *display all albums with a genre first before displaying the rest*
  - catalog.displayAlbums() – *display every album*

Drill: Instagram Feed

- Write a class of the Instagram Feed that displays the most recent posts and enables you to like and comment
    - feed = InstagramFeed();
    - len(feed) – *returns length of how many posts are in your feed*
    - feed.add(creator_name) – *adds a post to your feed*
    - feed.showPost() – *show only the most recent post*
    - feed.comment(creator_name, comment)– *adds a comment to that creator's post*
    - feed.releaseFeed() – *release and display the entire feed*

Recap

1. Linked List Purpose:
   a. To keep order and sequence
   b. To keep order and sequence
2. Hash map purpose:
   a. To give specific information associated to each and every unique key
3. Learn your combos:
   a. Linked List + Stack
   b. Linked List + Queue
   c. Linked List + Hash Maps
   d. Hash Maps + Stack
   e. Hash Maps + Queue

Exit Poll

Honestly, let me know how helpful this was to you!