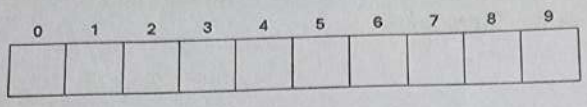
Course: Data Structures (CSE CS203A, 114-1)
Quiz II: Array, Linked List, Stack and Queue
October 21, 2025, 16:30~17:00

95

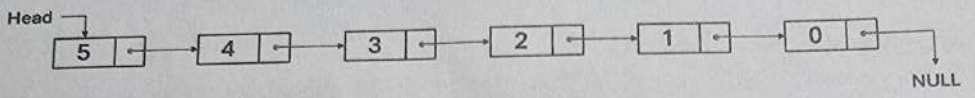
Student ID: 1131428 Student Name: 徐玥雯

Data Structures: Visualization

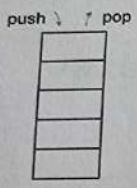
(1) Array



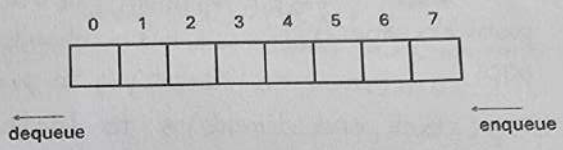
(2) Linked List



(3) Stack



(4) Queue



Q1: (30 pts; 10 pts for each) **Describe the mechanism of the function**
MoveTo(node *head, node *target, node*destination)

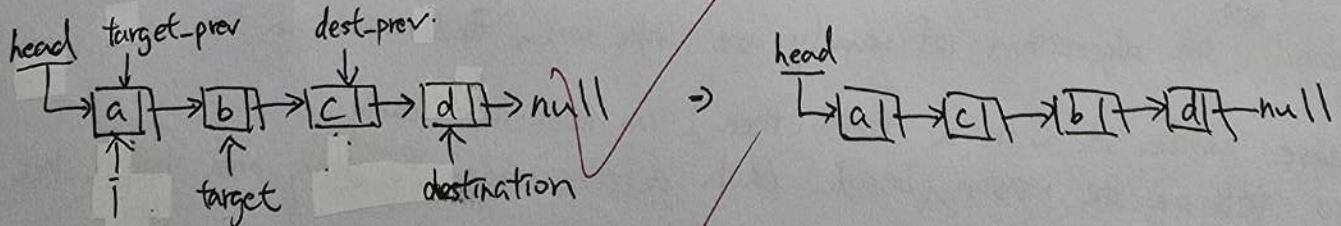
A1: Write a short paragraph explaining how the **MoveTo** function works (you may answer in English or Mandarin).

① Are there any **additional variables** required? If so, explain why they are necessary.

1. Let node target move to node destination's space, in front of the destination.

We need a pointer i to traverse whole linked list, find previous of target and destination and another target_prev, dest_prev to record it. than we can move target to destination.

② **Draw a visualization** of the singly linked list to support your explanation.



③ Is there any **variation of a linked list** (e.g., doubly linked list or circular linked list) that can simplify or improve this operation? doubly linked may do better because we can use previous link to find the node in front of target and destination.

Q2: (40 pts, 10 pts for each) **Definition of Data Structures**

Define the following data structures and list their fundamental operations.

A2:

-15

① Definition of "Stack"

A last-in-first-out structures

② Definition of "Queue"

A last-in-last-out structures

③ Preliminary operations of "Stack"

empty() ⇒ return 1 if stack is empty, size() ⇒ return the size of stack.

Access(stack, target) ⇒ to find a element in stack

push_back(stack, newElement) ⇒ to add a new element at the end.

pop(stack, end_element) ⇒ to delete the last element

④ Preliminary operations of "Queues"

Access(Queue, target) ⇒ to find a element in Queues.

push_back(Queue, newElement) ⇒ to add a new element at the end.

pop(Queue, front_Element) ⇒ to delete the first element in the front of Queue.

empty() ⇒ return 1 if Queue is empty, size() ⇒ return the size of Queue.

Q3: (30 pts) **AI Copilot Application**

Choose **up to two** data structures from the visualization list above.

Compose a **single prompt (within 300 words)** that you would use with an **AI Copilot** to explore or learn advanced concepts related to your chosen data structures.

A3:

Q: Could you give me the time complexity between array and linked list?

Please compare the action that people use usually, for example: accessing the struction, traversal.

and also, the algoreithm we usually use, like sort, find maxnum.....

Please write it with a 範例 then I can read it clearly. -10

also, tell me the reason beside each action under the array and linked list.

why it has this timecomplexity

at the last, tell me which structure has better O(), in general.