CC-213L

Data Structures and Algorithms

Laboratory 09

Revision Lab

Version: 1.0.0

Release Date: 22-11-2024

Department of Information Technology
University of the Punjab
Lahore, Pakistan

Contents:

- Learning Objectives
- Required Resources
- General Instructions
- Activities
  - In-Lab Activity
    - Task 01: Search in rotated Sorted Array
    - Task 02: Intersection Point of two Linked Lists
    - Task 03: Min Stack Implementation
- Submissions
- References and Additional Material
- Lab Time and Activity Simulation Log

Learning Objectives:

- Revision of previous concepts

Resources Required:

- Desktop Computer or Laptop
- Microsoft ® Visual Studio 2022

General Instructions:

- In this Lab, you are **NOT** allowed to discuss your solution with your colleagues, even not allowed to ask how is s/he doing, this may result in negative marking. You can **ONLY** discuss with your Teaching Assistants (TAs) or Lab Instructor.
- Your TAs will be available in the Lab for your help. Alternatively, you can send your queries via email to one of the followings.

Teachers:		
Course / Lab Instructor	Prof. Dr. Syed Waqar ul Qounain	swjaffry@pucit.edu.pk
Teacher Assistants	Muhammad Tahir Mustafvi	bcsf20m018@pucit.edu.pk
	Maryam Rasool	bcsf21m055@pucit.edu.pk

Lab Activities:

Task 01: Search in rotated sorted array

Given an integer array nums sorted in ascending order, but it has been rotated at an unknown pivot such that the resulting array is no longer fully sorted. For example, the array [0,1,2,4,5,6,7] might be rotated to [4,5,6,7,0,1,2].

Write a function to search for a target value in this rotated sorted array. If the target exists, return its index. Otherwise, return -1.You must write an algorithm with **O(log n)** runtime complexity.

```
int search(vector<int>& nums, int target) {
    //Your implementation here
}
```

Example

Input:

```
nums = [4,5,6,7,0,1,2], target = 0
```

Output:

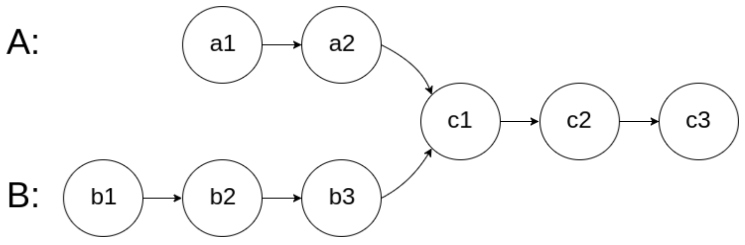
```
4
```

Task 02: Intersection of two Linked Lists

Given two singly linked lists, headA and headB. Write a function to find the node at which the two lists intersect. If the two linked lists have no intersection, return nullptr.

```
ListNode *getIntersectionNode(ListNode *headA, ListNode *headB) {
    //Your implementation here
}
```

Example



You will return node c1 in this case.

Task 03: Implement min Stack

Implement a stack that, in addition to the standard stack operations (push, pop, and top), supports retrieving the minimum element in the stack efficiently.

Your implementation must meet the following requirements:

1. push(x): Pushes element x onto the stack.
2. pop(): Removes the element on the top of the stack.
3. top(): Gets the top element.
4. getMin(): Retrieves the minimum element in the stack.

All operations must be **O(1)** in time complexity.

Submissions:

- For In-Lab Activity:
  - Save the files on your PC.
  - TA’s will evaluate the tasks offline.
- For Pre-Lab & Post-Lab Activity:
  - Submit the .cpp file on Google Classroom and name it to your roll no.

Evaluations Metric:

- All the lab tasks will be evaluated offline by TA’s
- **Division of Lab marks:** **[50 marks]**
  - Task 01: Search in rotated Sorted Array [10 marks]
  - Task 02: Intersection of two Linked Lists [20 marks]
  - Task 03: Min Stack [20 marks]

References and Additional Material:

Lab Time Activity Simulation Log:

- Slot – 01 – 02:15 – 02:30: Class Settlement
- Slot – 02 – 02:30 – 03:15: In-Lab Task 01
- Slot – 03 – 03:15 – 04:00: In-Lab Task 02
- Slot – 04 – 04:00 – 04:30: In-Lab Task 03
- Slot – 05 – 04:30 – 05:00: Discussion on Post-Lab