United International University
Dept. of Computer Science & Engineering
Course Title: Data Structure and Algorithm-I Laboratory
Section: C
Assignment-01
Marks: 25

---

1. Find the smallest missing element from a sorted array [07]

Given an unsorted array of non-negative distinct integers, find the smallest missing non-negative element in it.

Test Case:01
Input: nums[] = [0, 11, 15, 2, 9, 1, 6]
Output: The smallest missing element is 3
Explanation: If we sort the nums[] array, the sorted array will be like nums[] = [0, 1, 2, 6, 9, 11, 15]. So, the smallest missing number will be 3.

Test Case:02
Input: nums[] = [0, 11, 15, 2, 9, 1, 6]
Output: The smallest missing element is 3
Explanation: If we sort the nums[] array, the sorted array will be like nums[] = [0, 1, 2, 6, 9, 11, 15]. So, the smallest missing number will be 3.

N:B: You can use any sorting techniques.

2. Matching Pair [08]

Take a number N as input. Take an array of size N as input from the user. Take two integers X and Y as input. Write a function named SearchPair( ), which will take the input array and two integers as parameters. **Using any searching method** you like, this function will search for the element X and Y inside the array.

- If both elements are found, print "PAIR MATCHED".
- If only one of the elements is found, print "ONLY ME".
- If none of the elements are found, print "BETTER LUCK NEXT TIME".

Sample Input N A_1 A_2 ... A_N X Y	Sample Output
5 2 3 4 1 5 2 13	ONLY ME (2)
5 2 3 4 1 5 5 1	PAIR MATCHED
5 2 3 4 1 5 9 7	BETTER LUCK NEXT TIME

3. Singly Linked List [10]

Implement the following for a Singly Linked List that stores integers.

a. Create necessary structures and/or classes.
b. It will have both a head and a tail.
c. It will have 5 functions: **PrintList( ), ListLength( ), Insert( ), InsertSorted( ) and DeleteMin( ).**

- **PrintList( )** will print out the entire linked list.
- **ListLength( )** will return the length of the list.
- **Insert( )** will take the item to be inserted and insert it at the end of the list using tail. For example, suppose the list is **1->6**. After calling **Insert(2)**, the list will be **1->6->2**.
- **InsertSorted( )** will take the item to be inserted and insert it in a position such that a sorted list remains sorted. For example, suppose the list is 1->6. After calling **InsertSorted(7)**, the list will be **1->6->7**. Then, after calling InsertSorted(2), the list will be **1->2->6->7.**
- **DeleteMin( )** will delete the minimum element from the list. For example, suppose the list is **45->12->3->6->7**. After calling **DeleteMin( )**, the list will be **45->12->6->7.**

For operations like insert and delete, remember that you may need to update the head and/or tail pointer.

Implement a main function equivalent to the following main function to test your functions.

```
Insert(10);
Insert(17);
Insert(56);
PrintList( ); /// Singly Linked List: 10->17->56
ListLength( ); ///3
InsertSorted(23);
PrintList( ); /// Singly Linked List: 10->17->23->56
DeleteMin( );
PrintList( ); /// Singly Linked List: 17->23->56
InsertSorted(99);
PrintList( ); /// Singly Linked List: 17->23->56->99
InsertSorted(5);
PrintList( ); /// Singly Linked List: 5->17->23->56->99
LinkedList( ); ///5
```

Guideline:

Save three code files in a single directory, compress the directory into a ZIP file, and submit the ZIP file on the LMS.

Submission Deadline: Dec 30, 2024

Don't Miss The Deadline. If you miss the deadline, the following angry bird will be angry.