United International University
DSA 1 Lab - Assignment 1

In this assignment you will learn how to work with linked lists .

1. A singly linked list uses a head pointer that points to the first node of the list.

Task 1: Add getItemAt function. Add a new function getItemAt(int pos) that returns a pointer to the item at the position pos. Assume the item at the beginning of the list corresponds to position no 1. If pos is greater than the length of the list, it will return a NULL_VALUE.

Task 2: Add destructor function. The destructor will delete all the items and thereby free all the memory allocated for the list.

2.Task 3 - 6 is to be added in the LinkedListWithTail.cpp fle. The class includes an additional pointer tail that should always point to the last node of the list. In the given implementation, tail pointer is not set properly in implemented functions: insertItem and deleteItem. Your job is to add the following features.

Task 3: Set tail pointer correctly. Add required codes in insertItem and deleteItem functions to set up the pointer correctly so that it always points to the last node of the linked list.

Task 4: Make insertLast function efficient. For this task, your job is to use this tail pointer to make the insertLast function more efficient than your first implementation in Task 1 above, which runs in O(n) time. If you have a tail pointer, then you will not require searching the whole list to find the end of the list. So, change your previous implementation accordingly so that the new version runs in O(1) constant time. Ensure that the tail pointer is correctly set after insertion.

Task 5: Add getItemAt function. Same as the one implemented for LinkedList class. It will return a pointer to the item at the position pos. Assume the item at the beginning of the list corresponds to position no 1. If pos is greater than the length of the list, it will return a NULL_VALUE.

Task 6: Add deleteLast function. This function will delete the last element of the list. You must ensure that memory of the deleted item is correctly released. In case the item is not found in the list, return a NULL_VALUE, otherwise return SUCCESS_VALUE. Ensure that the tail pointer is correctly set after deletion

3. DoublyLinkedList. Create a Doubly Linked List. Do the following.
Task 7: print list reverse . Print the list in reverse order.
Task 8: InsertItem(int elem,int num). Insert the number after the element.
Task 9: DeleteLast
Task 10: InsertFirst

Submission Guideline:

1. Create a zip folder.Name it with your student id.
2. Submit the zip file within the deadline.
3. Do not copy code from anyone or from the internet.