

Sheet 1 (ADT List)

Q1) Using classes, Implement the Array-based ADT list, where ADT list contains the following representative functions:

1. **INSERT**(x, p, L). Insert x at position p in list L, moving elements at p and following positions to the next higher position. That is, if L is  $a_0, a_1, a_2, \dots, a_{n-1}$ , then L becomes  $a_0, a_1, a_2, \dots, a_{p-1}, x, a_p, \dots, a_{n-1}$ . If p is **END**(L), then L becomes  $a_1, a_2, \dots, a_n, x$ . If list L has no position p, the result is undefined.
2. **LOCATE**(x, L) returns the position of x on list L. If x appears more than once, then the position of the first occurrence is returned. If x does not appear at all, then **END**(L) is returned.
3. **RETRIEVE**(p, L) This function returns the element at position p on list L. The result is undefined if  $p = \text{END}(L)$  or if L has no position p. Note that the elements must be of a type that can be returned by a function if **RETRIEVE** is used. In practice, however, we can always modify **RETRIEVE** to return a pointer to an object of type `elementtype`.
4. **DELETE**(p, L). Delete the element at position p of list L. If L is  $a_0, a_1, a_2, \dots, a_{n-1}$ , then L becomes  $a_0, a_1, a_2, \dots, a_{p-1}, a_{p+1}, \dots, a_{n-1}$ . The result is undefined if L has no position p or if  $p = \text{END}(L)$ .
5. **NEXT**(p, L) and **PREVIOUS**(p, L) return the positions following and preceding position p on list L. If p is the last position on L, then **NEXT**(p, L) = **END**(L). **NEXT** is undefined if p is **END**(L). **PREVIOUS** is undefined if p is 0. Both functions are undefined if L has no position p.
6. **MAKENULL**(L). Delete the element at position p of list L. If L is  $a_0, a_1, a_2, \dots, a_n$ , then L becomes  $a_1, a_2, \dots, a_{p-1}, a_{p+1}, \dots, a_{n-1}$ . The result is undefined if L has no position p or if  $p = \text{END}(L)$ .
7. **FIRST**(L). This function returns the first position on list L. If L is empty, the position returned is **END**(L).
8. **PRINTLIST**(L). Print the elements of L in the order of occurrence.
9. **END**(L) returns the next of last position.

Finally, write a suitable main for testing these functions.

Q2) Using the ADL list in Q1,

1. Write a function **PURGE** that eliminates duplicates from list.
2. Write a function **Reverse** that reverses list.
3. Write a function **insertXafterY** that inserts a given value x after the value y in the list. If y doesn't exist in the list, add x at the end of the list.
4. Write a function **concatenate** that concatenates two lists.
5. Write a function **split** that splits the linked to two lists, one for odd numbers and one for even numbers.
6. Write a function **sum** that calculates the summation of all values in the list.

Q3) write a function **RemoveOccurrences1** that removes all occurrences of a given value x;

Q4) Modify the function **LOCATE**(x, L) to **LOCATEInRange**(x, L, Start, End), that searches for the giving value x in the list L only from the index **Start** to **End**.

Q5) Using the modified ADT list (after adding **LOCATEInRange**(x, L, Start, End) ), write a function **RemoveOccurences1** that removes all occurrences of a given value x. Compare between both two functions **RemoveOccurences1** and **RemoveOccurences2**.

Q6) Try to create a list of students (names, totalScore), then apply the same functions as in question 2-1, 2-2,2-3 (X and Y represents totalScore), 2-4, 2-5 (split it to success students and failed students). Also apply Q4.