Assignment-2

1. Create a linked list of numbers. Take input from user. Search and find the position of a particular key value. If the key-value exists, return its position, else add it to the end of the linked list.

(i) Write functions for each of these. Compare their performance in terms of number of comparisons and number of swaps.

(ii) Delete a particular element, chosen by the user, from the above linked list.

(iii)Add an element, input by the user, at a position (i) mentioned by the user, (ii) at the starting of the linked list, (ii) middle of the linked list, (iii) end of the linked list.

(iv) Update the value of a particular element, chosen by the user, to another value.

(v) If there are multiple instances of the same number, delete it from all the positions, except its (i) last position, (ii) first position, (iii) all the positions, where the element was found.

(vi)Display the contents of SLL in reverse order.

(vii) Display second largest number in SLL (Do not sort the link list).

2. Write a program to design a library management application using linked list which will maintain a record of all the books available in the library including (Book name, author, ISBN, publication year, no of copies available etc.). There is another list containing the name of the issued books and the name of student, to whom the book is issued. Provide menu which will contain options to add, delete, display and update the entry in the records.

3. Write a program to create another single link list (SLL1) having elements as 24, 6, 7, 8, 1, 2, 8, 10, 4, 27, 16, 26. Later, write a program to sort LL2 (retain all the duplicate elements). Call this list as SLL2. Finally merge SLL2 with SLL1 such that elements of merged linked list will be in sorted order (retaining all the duplicate elements).

4. Write a program to find sum of all elements of linked list created in Q1 and Q2 above separately. Write a function to perform following operations: If (sum of all elements of first LL - sum of all elements of second LL) > last element of first LL then start deleting last elements of first LL until (remaining elements of first LL - sum of all elements of second LL) ≈ 0. Otherwise delete elements of first LL from beginning until (remaining elements of first LL - sum of all elements of second LL) ≈ 0.