Section 14

PDS Lab **Lab - 10** **25.01.2023**

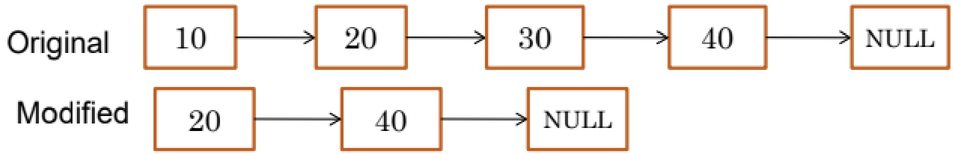
Instructions:

Give sufficient comment against each statement in your program.
You should save each program with the file name as specified against each problem.
There is a partial credit even if your program does not run successfully for all the test cases as mentioned.

1. a) Write a function **struct node *createlist(int n, int A[])**, which will create a list to store the elements of the array A[] containing n integers.

   b) Write a function **void printlist(struct node* l)** to print the elements of list l.

   c) Write a function **void even(struct node* l)** which will modify the input list l to store only the even numbered (2,4, 6 ..) nodes of the original list.



- Write a main program which will read an integer n (<10), and then n integers and store them in an array A.
- Create a linked list l to store these integers by using function **createlist(...)**.
- Modify the list l using the **even(...)** function.
- Use function **printlist(...)** to print elements of the modified list.
- Full credit if no malloc(is made within the **even** function.

[Time: 30 minutes] [5+5+5]

2. You have given a linked list, say L. You have to check if the list is sorted either in ascending or descending order.

Test case#1:

```
Input:
   L = {1, 3, 5, 7, 9}
Output:
  The input list L is in ascending order.
```

Test case#2:

```
Input:
   L = {1, 9, 7, 5, 3}
Output:
  The input list L is not in sorted order.
```

Test case#3:

```
Input:
   L = {9, 7, 5, 3, 1}
Output:
  The input list L is in descending order.
```

Hint: You can reuse the functions defined in Q.1 in this lab.

[Time: 20 minutes] [5+5+5]

3. You have given a linked list, say `L`. You have to check if the list contains any duplicate element or not. Also, find the minimum and maximum numbers in the input list.

Test case#1:

```
Input:
  L = {1, 2, 3, 9, 7, 7, 6, 5, 4, 3, 2, 1}
Outpt:
  The input list contains duplicate numbers of: 1, 2, 3, 7.
  Minimum number is: 1
  Maximum number is: 9
```

Test case#2:

```
Input:
  L = {2, 4, 6, 8, 9, 7, 5, 3, 1}
Output:
  The input list does not contain any duplicate
 number.
  Minimum number is: 1
  Maximum number is: 9
```

[Time: 20 minutes] [5+5+5]

4. Define a structure `Customer` to specify data of customer in a bank. The data to be stored are: `Account number` (integer), `Name` (character string of exact length as appropriate to store a customer's name), and `Balance` in account (integer).

   You have to read records from the keyboard and store them in a linked list and finally print the records in the list.

   - Your program should read records till user wants to continue.
   - Your program should insert a new record in the ascending order of the customers' account number.
   - Once the record entry is over, print the record in the list in descending order of the balance of the customer.

[Time: 40 minutes] [5+10+10]

5. Two lists are given, say L1 and L2. You can use the function `struct Node *createList (int n)` as defined in Q.1 to create the lists of size n_1 and n_2, respectively.

- Find a new list L such that $L = L1 \cup L2$, that is, union of the lists L1 and L2.
- Find a new list L such that $L = L1 \cap L2$, that is, intersection of the lists L1 and L2.

```
Test case#1:
Input:
   L1 = {1, 3, 5, 7, 9}
   L2 = {2, 3, 4, 5, 6, 7}
Output:
  L (Union) = {1, 2, 3, 4, 5, 6, 7, 9}
  L (Intersection) = {3, 5, 7}

Test case#2:
Input:
   L1 = {1, 3, 5, 7, 9}
   L2 = {0, 2, 4, 6, 8}
Output:
  L (Union) = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
  L (Intersection) = {}

Test case#3:
Input:
   L1 = {1, 2, 3, 4, 5}
   L2 = {5, 4}
Output:
  L (Union) = {1, 2, 3, 4, 5}
  L (Intersection) = {4, 5}
```

[Time: 50 minutes] [15+15]