Program 5

Write a program to Implement Singly Linked List with following operations

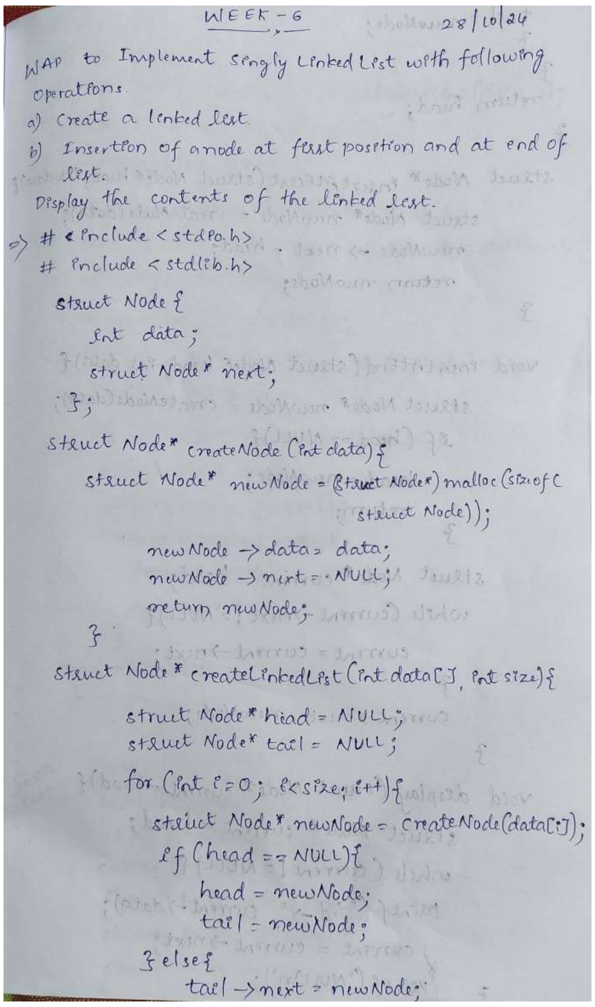
a) Create a linked list.

b) Insertion of a node at **first position** and **at end of list.**

Display the contents of the linked list.

Leetcode problem no.20 (Valid parantheses)

Observation :



WEEK - 6 28/10/24

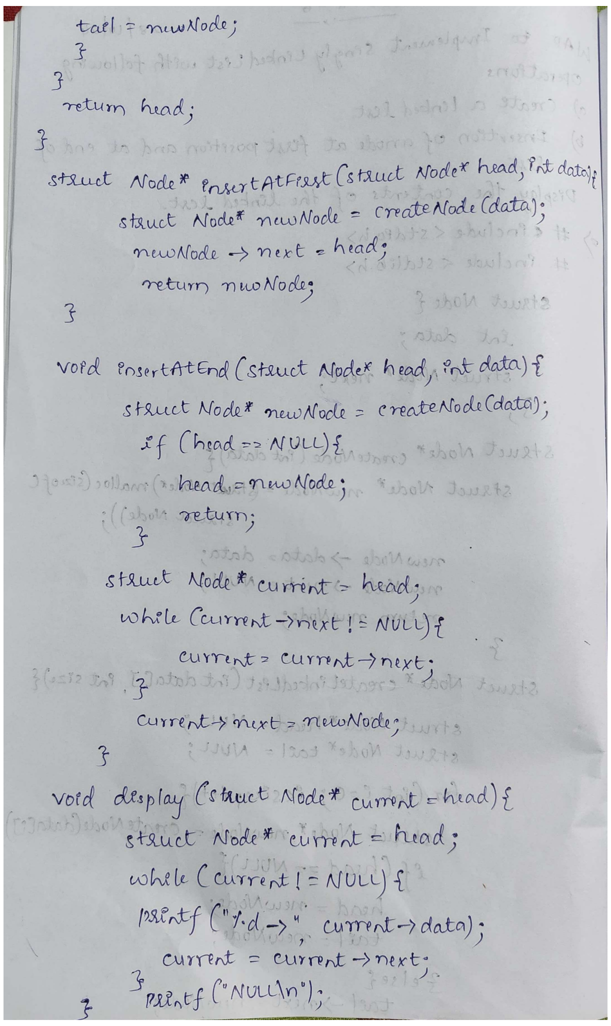
WAP to Implement singly Linked List with following operations.

a) Create a linked list

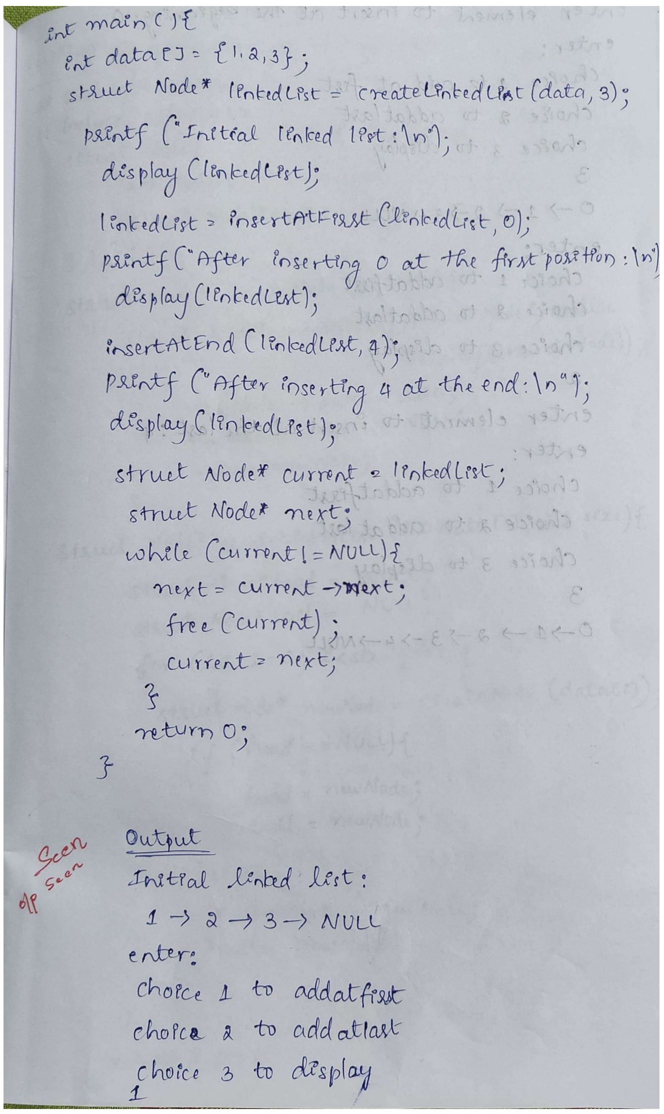
b) Insertion of anode at first position and at end of list.

Display the contents of the linked list.

```
=> # include <stdio.h>
   # include <stdlib.h>

   struct Node {
       int data;
       struct Node* next;
   };

   struct Node* createNode (int data){
       struct Node* newNode = (struct Node*) malloc (sizeof(
                                        struct Node));
           newNode -> data = data;
           newNode -> next = NULL;
           return newNode;
   }
   struct Node* createLinkedList (int data[], int size){
         struct Node* head = NULL;
         struct Node* tail = NULL;
         for (int i=0; i<size; i++){
           struct Node* newNode = createNode(data[i]);
            if (head == NULL){
               head = newNode;
               tail = newNode;
           }else{
              tail -> next = newNode;
```



```
     tail = newNode;
     }
   }
   return head;
}
struct Node* insertAtFirst (struct Node* head, int data){
        struct Node* newNode = createNode (data);
         newNode -> next = head;
         return newNode;
  }

 void insertAtEnd (struct Node* head, int data){
        struct Node* newNode = createNode(data);
         if (head == NULL){
            head = newNode;
            return;
         }
       struct Node* current = head;
        while (current->next != NULL){
              current = current -> next;
         }
         current -> next = newNode;
    }
 void display (struct Node* current = head){
       struct Node* current = head;
       while (current != NULL) {
       printf ("%d ->", current -> data);
          current = current -> next;
      }
       printf ("NULL\n");
   }
```

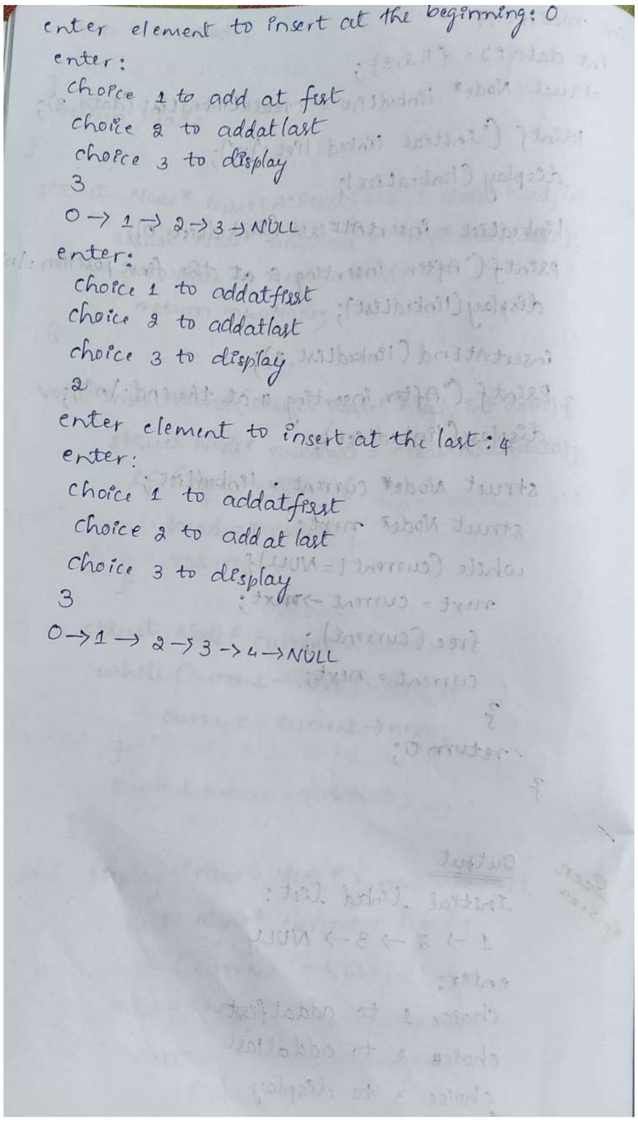


```
int main(){
 int data[] = {1,2,3};
 struct Node* linkedList = createLinkedList(data, 3);
  printf("Initial linked list:\n");
   display(linkedList);

  linkedList = insertAtFirst(linkedList, 0);
  printf("After inserting 0 at the first position:\n")
   display(linkedList);

  insertAtEnd(linkedList, 4);
  printf("After inserting 4 at the end:\n");
  display(linkedList);

   struct Node* current = linkedList;
    struct Node* next;
    while (current != NULL){
       next = current->next;
        free(current);
        current = next;
     }
    return 0;
 }
```

Seen
o/p Seen

Output

```
Initial linked list:
 1 -> 2 -> 3 -> NULL
enter:
 choice 1 to addatfirst
 choice 2 to addatlast
 choice 3 to display
1
```

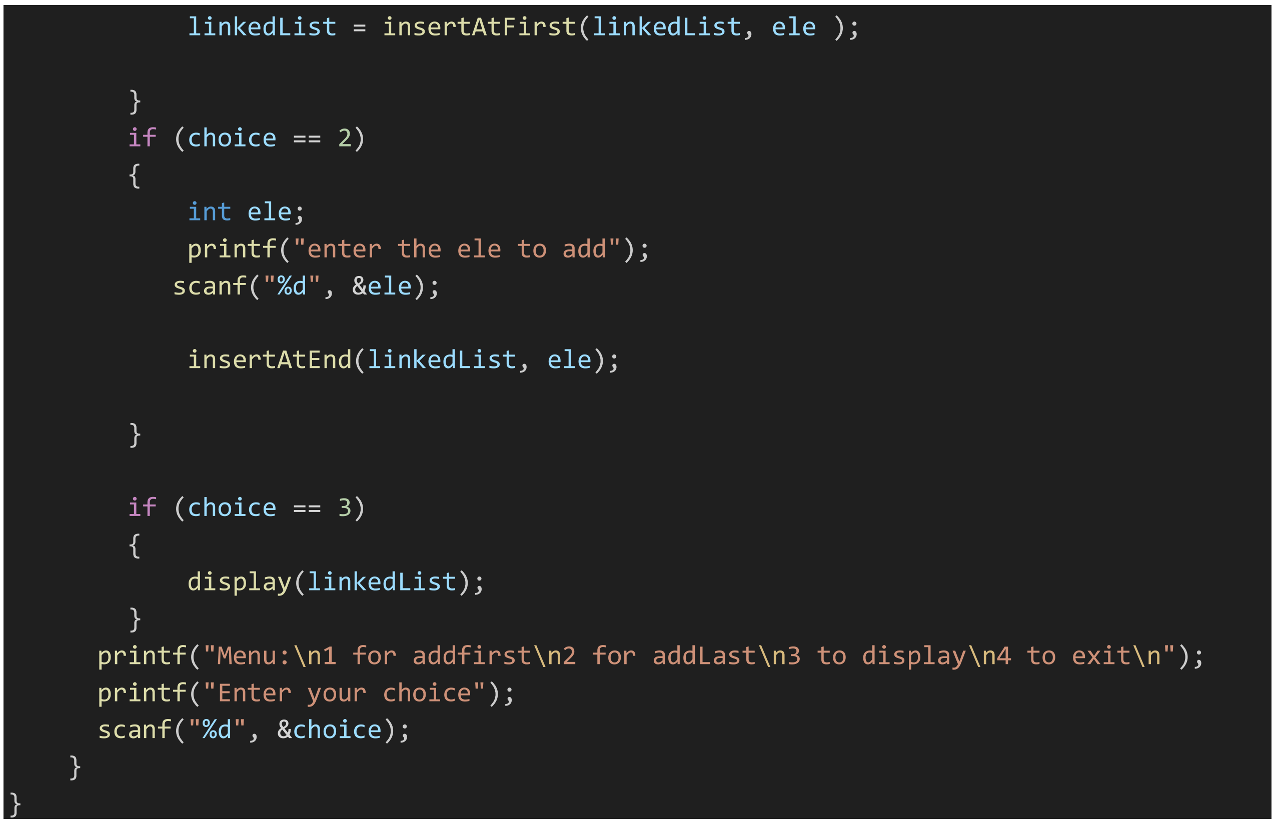


```
enter element to insert at the beginning: 0
 enter:
  choice 1 to add at first
  choice 2 to addatlast
  choice 3 to display
  3
  0 -> 1 -> 2 -> 3 -> NULL
 enter:
  choice 1 to addatfirst
  choice 2 to addatlast
  choice 3 to display
  2
 enter element to insert at the last: 4
 enter:
  choice 1 to addatfirst
  choice 2 to add at last
  choice 3 to display
 3
0 -> 1 -> 2 -> 3 -> 4 -> NULL
```

Code:

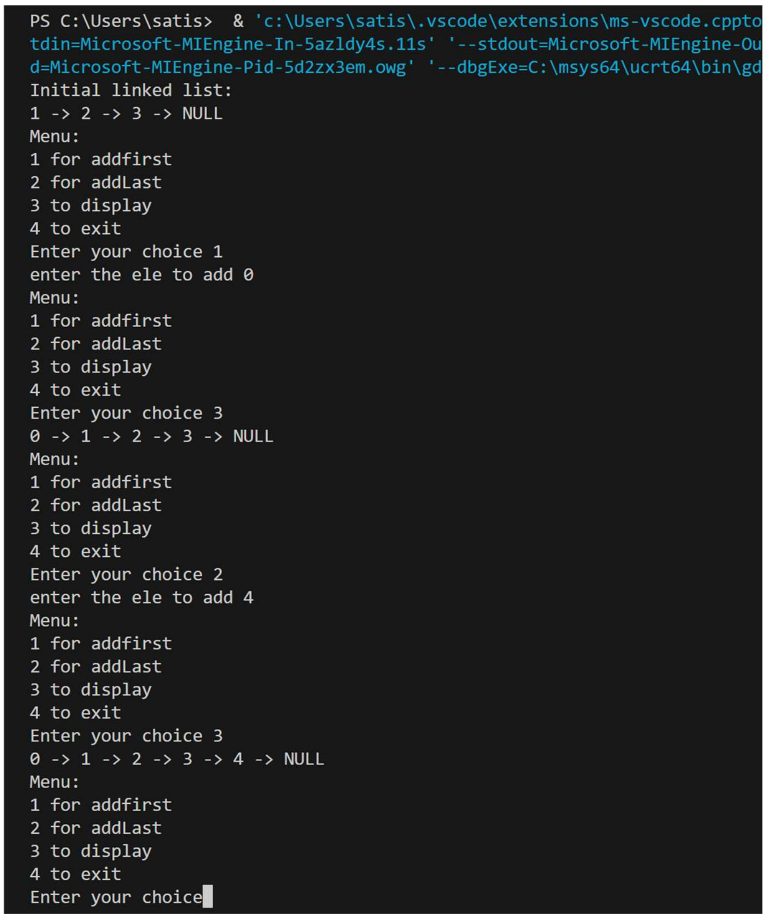
```c
 #include <stdio.h>
#include <stdlib.h>

struct Node
{
    int data;
    struct Node *next;
};

struct Node *createNode(int data)
{
    struct Node *newNode = (struct Node *)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->next = NULL;
    return newNode;
}

struct Node *createLinkedList(int data[], int size)
{
    struct Node *head = NULL;
    struct Node *tail = NULL;

    for (int i = 0; i < size; i++)
    {
        struct Node *newNode = createNode(data[i]);
        if (head == NULL)
        {
            head = newNode;
            tail = newNode;
        }
        else
        {
            tail->next = newNode;
            tail = newNode;
        }
    }

    return head;
}

struct Node *insertAtFirst(struct Node *head, int data)
{
    struct Node *newNode = createNode(data);
    newNode->next = head;
    return newNode;
}
```

```c
void insertAtEnd(struct Node *head, int data)
{
    struct Node *newNode = createNode(data);
    if (head == NULL)
    {
        head = newNode;
        return;
    }

    struct Node *current = head;
    while (current->next != NULL)
    {
        current = current->next;
    }
    current->next = newNode;
}

void display(struct Node *head)
{
    struct Node *current = head;
    while (current != NULL)
    {
        printf("%d -> ", current->data);
        current = current->next;
    }
    printf("NULL\n");
}

int main()
{

    int data[] = {1, 2, 3};
    struct Node *linkedList = createLinkedList(data, 3);

    printf("Initial linked list:\n");
    display(linkedList);

    int choice;
    printf("Menu:\n1 for addfirst\n2 for addLast\n3 to display\n4 to exit\n");
    printf("Enter your choice");
    scanf("%d", &choice);

    while( choice != 4)
    {

        if (choice == 1)
        {
             int ele;
            printf("enter the ele to add");
           scanf("%d", &ele);
```

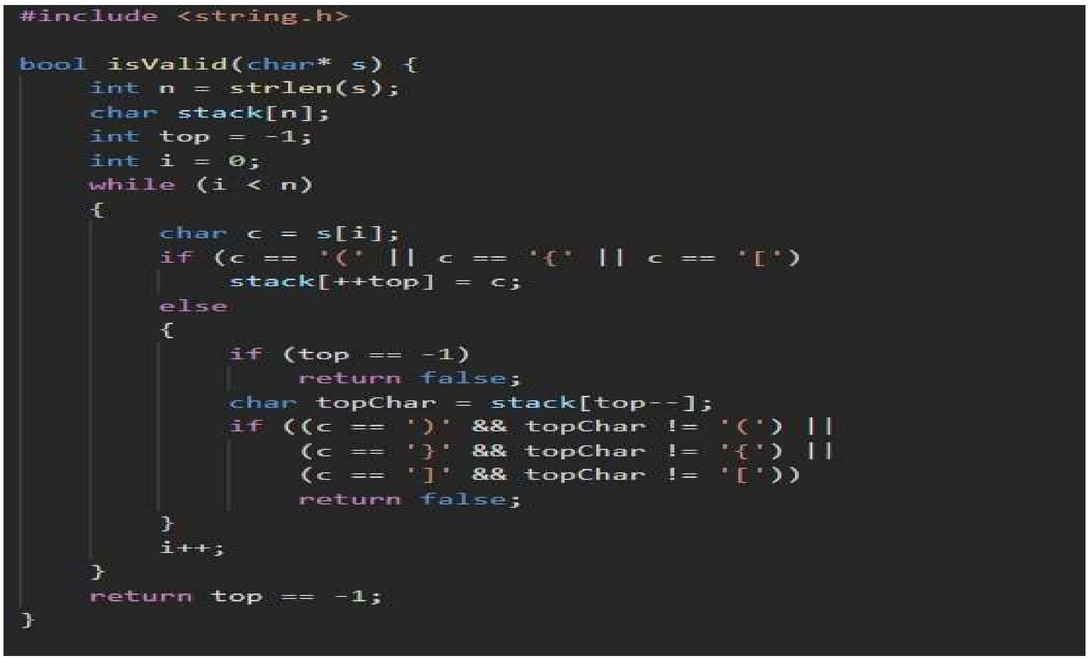


```c
            linkedList = insertAtFirst(linkedList, ele );

        }
        if (choice == 2)
        {
            int ele;
            printf("enter the ele to add");
           scanf("%d", &ele);

            insertAtEnd(linkedList, ele);

        }

        if (choice == 3)
        {
            display(linkedList);
        }
      printf("Menu:\n1 for addfirst\n2 for addLast\n3 to display\n4 to exit\n");
      printf("Enter your choice");
      scanf("%d", &choice);
    }
}
```

Output:



```
PS C:\Users\satis>  & 'c:\Users\satis\.vscode\extensions\ms-vscode.cppto
tdin=Microsoft-MIEngine-In-5azldy4s.11s' '--stdout=Microsoft-MIEngine-Ou
d=Microsoft-MIEngine-Pid-5d2zx3em.owg' '--dbgExe=C:\msys64\ucrt64\bin\gd
Initial linked list:
1 -> 2 -> 3 -> NULL
Menu:
1 for addfirst
2 for addLast
3 to display
4 to exit
Enter your choice 1
enter the ele to add 0
Menu:
1 for addfirst
2 for addLast
3 to display
4 to exit
Enter your choice 3
0 -> 1 -> 2 -> 3 -> NULL
Menu:
1 for addfirst
2 for addLast
3 to display
4 to exit
Enter your choice 2
enter the ele to add 4
Menu:
1 for addfirst
2 for addLast
3 to display
4 to exit
Enter your choice 3
0 -> 1 -> 2 -> 3 -> 4 -> NULL
Menu:
1 for addfirst
2 for addLast
3 to display
4 to exit
Enter your choice
```



```c
#include <string.h>

bool isValid(char* s) {
    int n = strlen(s);
    char stack[n];
    int top = -1;
    int i = 0;
    while (i < n)
    {
        char c = s[i];
        if (c == '(' || c == '{' || c == '[')
            stack[++top] = c;
        else
        {
            if (top == -1)
                return false;
            char topChar = stack[top--];
            if ((c == ')' && topChar != '(') ||
                (c == '}' && topChar != '{') ||
                (c == ']' && topChar != '['))
                return false;
        }
        i++;
    }
    return top == -1;
}
```

Leet Code Valid Parenthesis