

Aim:

Write a program to implement stack using **linked lists**.

Sample Input and Output:

```
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 1
Enter element : 33
Successfully pushed.
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 1
Enter element : 22
Successfully pushed.
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 1
Enter element : 55
Successfully pushed.
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 1
Enter element : 66
Successfully pushed.
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 3
Elements of the stack are : 66 55 22 33
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 2
Popped value = 66
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 2
Popped value = 55
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 3
Elements of the stack are : 22 33
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 5
Peek value = 22
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 4
Stack is not empty.
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit
Enter your option : 6
```

Source Code:

StackUsingLList.c

```
#include <stdio.h>
#include <stdlib.h>
struct stack
{
    int data;
```

```
    struct stack *next;
};
typedef struct stack *stk;
stk top = NULL;
stk push(int x)
{
    stk temp;
    temp = (stk)malloc(sizeof(struct stack));
    if(temp == NULL)
    {
        printf("Stack is overflow.\n");
    }
    else
    {
        temp -> data = x;
        temp -> next = top;
        top = temp;
        printf("Successfully pushed.\n");
    }
}
void display()
{
    stk temp = top;
    if(temp == NULL)
    {
        printf("Stack is empty.\n");
    }
    else
    {
        printf("Elements of the stack are : ");
        while(temp != NULL)
        {
            printf("%d ", temp -> data);
            temp = temp -> next;
        }
        printf("\n");
    }
}
stk pop()
{
    stk temp;
    if(top == NULL)
    {
        printf("Stack is underflow.\n");
    }
    else
    {
        temp = top;
        top = top -> next;
        printf("Popped value = %d\n", temp -> data);
        free(temp);
    }
}
void peek()
{
    stk temp;
```

```
if(top == NULL)
{
    printf("Stack is underflow.\n");
}
else
{
    temp = top;
    printf("Peek value = %d\n", temp -> data);
}
}
void isEmpty()
{
    if(top == NULL)
    {
        printf("Stack is empty.\n");
    }
    else
    {
        printf("Stack is not empty.\n");
    }
}
int main()
{
    int op, x;
    while(1)
    {
        printf("1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit\n");
        printf("Enter your option : ");
        scanf("%d", &op);
        switch(op)
        {
            case 1:
                printf("Enter element : ");
                scanf("%d", &x);
                push(x);
                break;
            case 2:
                pop();
                break;
            case 3:
                display();
                break;
            case 4:
                isEmpty();
                break;
            case 5:
                peek();
                break;
            case 6:
                exit(0);
        }
    }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Enter your option : 1
Enter element : 33
Successfully pushed. 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Enter your option : 1
Enter element : 22
Successfully pushed. 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Enter your option : 1
Enter element : 55
Successfully pushed. 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Enter your option : 1
Enter element : 66
Successfully pushed. 3
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 3
Enter your option : 3
Elements of the stack are : 66 55 22 33 2
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 2
Enter your option : 2
Popped value = 66 2
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 2
Enter your option : 2
Popped value = 55 3
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 3
Enter your option : 3
Elements of the stack are : 22 33 5
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 5
Enter your option : 5
Peek value = 22 4
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 4
Enter your option : 4
Stack is not empty. 6
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 6
Enter your option : 6

Test Case - 2
User Output
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 2
Enter your option : 2
Stack is underflow. 3
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 3
Enter your option : 3
Stack is empty. 5
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 5
Enter your option : 5
Stack is underflow. 4
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 4

Enter your option : 4
Stack is empty. 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Enter your option : 1
Enter element : 23
Successfully pushed. 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Enter your option : 1
Enter element : 24
Successfully pushed. 3
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 3
Enter your option : 3
Elements of the stack are : 24 23 5
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 5
Enter your option : 5
Peek value = 24 2
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 2
Enter your option : 2
Popped value = 24 2
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 2
Enter your option : 2
Popped value = 23 2
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 2
Enter your option : 2
Stack is underflow. 4
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 4
Enter your option : 4
Stack is empty. 6
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 6
Enter your option : 6