Program

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

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

struct node *top = NULL;

void push(int value) {
    struct node *newnode = (struct node *)malloc(sizeof(struct node));
    newnode->data = value;
    newnode->next = top;
    top = newnode;
}

void pop() {
    struct node *temp = top;
    if (temp == NULL) {
        printf("Underflow!\n");
    } else {
        printf("Deleted Item: %d\n", temp->data);
        top = top->next;
        free(temp);
    }
}

void peek() {
    if (top == NULL) {
        printf("Stack is Empty!\n");
    } else {
        printf("Top Element: %d\n", top->data);
    }
}

void display() {
    struct node *temp = top;
    if (temp == NULL) {
        printf("Stack Empty!\n");
    } else {
        printf("Stack elements:");
        while (temp != NULL) {
            printf(" %d", temp->data);
            temp = temp->next;
        }
        printf("\n");
    }
}

int main() {
    int ch;
    int value;
    do {
        printf("\nMENU.......\n");
        printf("1. Push\n2. Pop\n3. Display\n4. Peek\n5. Exit\n");
        printf("Enter Option: ");
        scanf("%d", &ch);
        switch (ch) {
            case 1:
```

Experiment No: 14
Date:

STACK IMPLEMENTATION USING LINKED LIST

Aim:

To implement a Stack data structure using Linked List.

Algorithm:

```
1. Start
2. Define a structure Node with:
       data: An integer to store the element.
       next: A pointer to the next node.
3. Initialize top to NULL (represents the top of the stack).
4. Function push(value):
       Allocate memory for a new node.
       Store value in newnode->data.
       Set newnode->next = top.
       Update top = newnode.
5. Function pop():
       If top == NULL, print "Underflow!".
       Else:
           Store top in a temporary pointer temp.
           Print temp->data.
           Update top = top->next.
           Free the memory of temp.
6. Function peek():
       If top == NULL, print "Stack is Empty!".
       Else:
           Print "Top Element:".
           Print the value of top->data.
7. Function display():
       If top == NULL, print "Stack Empty!".
```

```
                printf("Enter the Element: ");
                scanf("%d", &value);
                push(value);
                break;
            case 2:
                pop();
                break;
            case 3:
                display();
                break;
            case 4:
                peek();
                break;
            case 5:
                break;
            default:
                printf("Invalid option! Please choose again.\n");
        }
    } while (ch != 5);
    return 0;
}
```

Output

```
MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 1
Enter the Element: 30

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 1
Enter the Element: 20

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 1
Enter the Element: 10

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 3
Stack elements: 10 20 30
```

```
    Else:
        Print "Stack elements:".
        Initialize a temporary pointer temp = top.
        While temp != NULL:
            Print temp->data.
            Set temp = temp->next.
```

8. **Main Function**:

```
    Declare variables ch (choice) and value (input value).
    Repeat Until ch = 5:
        Print the menu options:
                              1. Push
                              2. Pop
                              3. Display
                              4. Peek
                              5. Exit

        Read ch.
        Switch on ch:
            Case 1:
                Print "Enter the Element:".
                Read value.
                Call push(value).
            Case 2: Call pop().
            Case 3: Call display().
            Case 4: Call peek().
            Case 5: Exit the loop.
            Default: Print "Invalid option! Please choose again."
```

9. **Stop**

```
MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 2
Deleted Item: 10

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 4
Top Element: 20

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 2
Deleted Item: 20

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 2
Deleted Item: 30

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 2
Underflow!

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 3
Stack Empty!

MENU.......
1. Push
2. Pop
3. Display
4. Peek
5. Exit
Enter Option: 5
```

Result:

Program has been executed successfully and obtained the output.