

SEM 3\Exp5\Stack_LL.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  // Node structure for the linked list
5  struct Node
6  {
7      int data;
8      struct Node *next;
9  };
10
11 // Stack structure using linked lists
12 struct StackLinkedList
13 {
14     struct Node *top;
15 };
16
17 // Function to create a stack
18 struct StackLinkedList *createStack()
19 {
20     struct StackLinkedList *stack = (struct StackLinkedList *)malloc(sizeof(struct
StackLinkedList));
21     stack->top = NULL; // Initialize the top pointer
22     return stack;
23 }
24
25 // Check if the stack is empty
26 int isEmpty(struct StackLinkedList *stack)
27 {
28     return stack->top == NULL;
29 }
30
31 // Push an element onto the stack
32 void push(struct StackLinkedList *stack, int value)
33 {
34     struct Node *new_node = (struct Node *)malloc(sizeof(struct Node));
35     new_node->data = value;
36     new_node->next = stack->top;
37     stack->top = new_node;
38     printf("%d pushed onto stack\n", value);
39 }
40
41 // Pop an element from the stack
42 int pop(struct StackLinkedList *stack)
43 {
44     if (isEmpty(stack))
45     {
46         printf("Stack underflow!\n");
47         return -1;
48     }
49     struct Node *temp = stack->top;
50     int popped_value = temp->data;
51     stack->top = stack->top->next;
```

```
52     free(temp);
53     return popped_value;
54 }
55
56 // Peek at the top element of the stack
57 int peek(struct StackLinkedList *stack)
58 {
59     if (isEmpty(stack))
60     {
61         printf("Stack is empty!\n");
62         return -1;
63     }
64     return stack->top->data;
65 }
66
67 // Display the stack
68 void display(struct StackLinkedList *stack)
69 {
70     if (isEmpty(stack))
71     {
72         printf("Stack is empty!\n");
73         return;
74     }
75     struct Node *temp = stack->top;
76     printf("Stack elements: ");
77     while (temp != NULL)
78     {
79         printf("%d ", temp->data);
80         temp = temp->next;
81     }
82     printf("\n");
83 }
84
85 int main()
86 {
87     struct StackLinkedList *stack = createStack();
88     int choice, value;
89
90     printf("\nStack Operations (Linked List Implementation):\n");
91     printf("1. Push\n");
92     printf("2. Pop\n");
93     printf("3. Peek\n");
94     printf("4. Display\n");
95     printf("5. Exit\n");
96
97     while (1)
98     {
99
100         printf("Enter your choice: ");
101         scanf("%d", &choice);
102
103         switch (choice)
104         {
105             case 1:
```

```
106     printf("Enter the value to push: ");
107     scanf("%d", &value);
108     push(stack, value);
109     printf("\n");
110     break;
111 case 2:
112     value = pop(stack);
113     if (value != -1)
114         printf("Popped value: %d\n", value);
115
116     printf("\n");
117     break;
118 case 3:
119     value = peek(stack);
120     if (value != -1)
121         printf("Top value: %d\n", value);
122
123     printf("\n");
124     break;
125 case 4:
126     display(stack);
127     printf("\n");
128     break;
129 case 5:
130     // Free linked list nodes (cleanup)
131     while (!isEmpty(stack))
132     {
133         pop(stack);
134     }
135     free(stack);
136     exit(0);
137 default:
138     printf("Invalid choice!\n");
139 }
140 }
141
142 return 0;
143 }
144
```