

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Node {
5     int data;
6     struct Node* next;
7 };
8
9 struct Stack {
10     struct Node* top;
11 };
12
13 void push(struct Stack* stack, int data) {
14     struct Node* new_node = (struct Node*) malloc(sizeof(struct Node));
15     new_node->data = data;
16     new_node->next = stack->top;
17     stack->top = new_node;
18 }
19
20 int pop(struct Stack* stack) {
21     if (stack->top == NULL) return -1;
22     struct Node* temp = stack->top;
23     int popped = temp->data;
24     stack->top = temp->next;
25     free(temp);
26     return popped;
27 }
28
29 int peekStack(struct Stack* stack) {
30     if (stack->top == NULL) return -1;
31     return stack->top->data;
32 }
33
34
35 struct Queue {
36     struct Node *front, *rear;
37 };
38
39 void enqueue(struct Queue* queue, int data) {
40     struct Node* new_node = (struct Node*) malloc(sizeof(struct Node));
41     new_node->data = data;
42     new_node->next = NULL;
43     if (queue->rear == NULL) {
44         queue->front = queue->rear = new_node;
45         return;
46     }
47     queue->rear->next = new_node;
48     queue->rear = new_node;
49 }
50
```

```
51 int dequeue(struct Queue* queue) {
52     if (queue->front == NULL) return -1;
53     struct Node* temp = queue->front;
54     int dequeued = temp->data;
55     queue->front = temp->next;
56     if (queue->front == NULL)
57         queue->rear = NULL;
58     free(temp);
59     return dequeued;
60 }
61
62 int peekQueue(struct Queue* queue) {
63     if (queue->front == NULL) return -1;
64     return queue->front->data;
65 }
66
67 int main() {
68     struct Stack stack;
69     stack.top = NULL;
70     struct Queue queue;
71     queue.front = queue.rear = NULL;
72
73     int choice, value;
74
75     while (1) {
76         printf("1: Push stack, 2: Pop stack, 3: Display stack top\n");
77         printf("4: Enqueue queue, 5: Dequeue queue, 6: Display queue front\n");
78         printf("0: Exit\nEnter choice: ");
79         scanf("%d", &choice);
80         if (choice == 0) break;
81         switch (choice) {
82             case 1:
83                 printf("Enter value to push: ");
84                 scanf("%d", &value);
85                 push(&stack, value);
86                 break;
87             case 2:
88                 value = pop(&stack);
89                 if (value == -1) printf("Stack Empty\n");
90                 else printf("Popped %d\n", value);
91                 break;
92             case 3:
93                 value = peekStack(&stack);
94                 if (value == -1) printf("Stack Empty\n");
95                 else printf("Stack top: %d\n", value);
96                 break;
97             case 4:
98                 printf("Enter value to enqueue: ");
99                 scanf("%d", &value);
100                enqueue(&queue, value);
```

```
101         break;
102     case 5:
103         value = dequeue(&queue);
104         if (value == -1) printf("Queue Empty\n");
105         else printf("Dequeued %d\n", value);
106         break;
107     case 6:
108         value = peekQueue(&queue);
109         if (value == -1) printf("Queue Empty\n");
110         else printf("Queue front: %d\n", value);
111         break;
112     default:
113         printf("Invalid choice\n");
114     }
115 }
116
117 return 0;
118 }
```