

LAB ASSIGNMENT 6

Stacks and Queues

1. Write a menu driven program with 4 options (Insert, Delete, Display, and Exit) to demonstrate the working of Queues using arrays.

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 #define MAX 10
4 int queue_arr[MAX];
5 int rear=-1;
6 int front=-1;
7 void insert(int item);
8 int del();
9 int peek();
10 void display();
11 int isFull();
12 int isEmpty();
13 int main()
14 {
15     int choice,item;
16     while(1)
17     {
18         printf("\n1.Insert\n");
19         printf("\n2.Delete\n");
20         printf("\n3.Display element at the front\n");
21         printf("\n4.Display all elements of the queue\n");
22         printf("\n5.Quit\n");
23         printf("\nEnter your choice : ");
24         scanf("%d",&choice);
25         switch(choice)
26         {
27             case 1:
28                 printf("\nInput the element for adding in queue : ");
29                 scanf("%d",&item);
30                 insert(item);
31                 break;
32             case 2:
33                 item=del();
34                 printf("\nDeleted element is %d\n",item);
35                 break;
36             case 3:
37                 printf("\nElement at the front is %d\n",peek());
38                 break;
39             case 4:
40                 display();
41                 break;
42             case 5:
43                 exit(1);
44         }
45     }
46     default:
47         printf("\nWrong choice\n");
48     }
49     return 0;
50 }
```

```
43         exit(1);
44     default:
45         printf("\nWrong choice\n");
46     }
47 }
48 return 0;
49 }
50
51 int peek()
52 {
53     if( isEmpty() )
54     {
55         printf("\nQueue Underflow\n");
56         exit(1);
57     }
58     return queue_arr[front];
59 }
60 int isEmpty()
61 {
62     if( front== -1 || front==rear+1 )
63         return 1;
64     else
65         return 0;
66 }
67 int isFull()
68 {
69     if( rear==MAX-1 )
70         return 1;
71     else
72         return 0;
73 }
74 void display()
75 {
76     int i;
77     if ( isEmpty() )
78     {
79         printf("\nQueue is empty\n");
80         return;
81     }
82     printf("\nQueue is :\n\n");
83     for(i=front;i<=rear;i++)
```

OUTPUT:

```
1.Insert
2.Delete
3.Display element at the front
4.Display all elements of the queue
5.Quit

Enter your choice : 1

Input the element for adding in queue : 34

1.Insert
2.Delete
3.Display element at the front
4.Display all elements of the queue
5.Quit

Enter your choice : 1

Input the element for adding in queue : 56

1.Insert
2.Delete
3.Display element at the front
4.Display all elements of the queue
5.Quit

Enter your choice : 3

Element at the front is 34
```

2. Write a menu driven program with 4 options (Insert, Delete, Display, and Exit) to demonstrate the working of Queues using linked-list.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 struct Node {
4     int data;
5     struct Node* next;
6 };
7 struct Node* front = NULL;
8 struct Node* rear = NULL;
9 void enqueue(int data);
10 int dequeue();
11 void display();
12 int isEmpty();
13 int main() {
14     int choice, data;
15     while (1) {
16         printf("\nMenu:\n");
17         printf("1. Insert\n");
18         printf("2. Delete\n");
19         printf("3. Display\n");
20         printf("4. Exit\n");
21         printf("Enter your choice: ");
22         scanf("%d", &choice);
23         switch (choice) {
24             case 1:
25                 printf("Enter data to insert: ");
26                 scanf("%d", &data);
27                 enqueue(data);
28                 break;
29             case 2:
30                 data = dequeue();
31                 if (data != -1) {
32                     printf("Deleted element: %d\n", data);
33                 }
34                 break;
35             case 3:
36                 display();
37                 break;
38             case 4:
39                 printf("Exiting...\n");
40                 exit(0);
41             default:
42                 printf("Invalid choice!\n");
43         }
44     }
45 }
```

```
43     }
44 }
45
46 return 0;
47 }
48 void enqueue(int data) {
49     struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
50     newNode->data = data;
51     newNode->next = NULL;
52
53     if (isEmpty()) {
54         front = rear = newNode;
55     } else {
56         rear->next = newNode;
57         rear = newNode;
58     }
59 }
60 void display() {
61     if (isEmpty()) {
62         printf("Queue is empty!\n");
63         return;
64     }
65
66     struct Node* temp = front;
67     printf("Queue elements: ");
68     while (temp != NULL) {
69         printf("%d ", temp->data);
70         temp = temp->next;
71     }
72     printf("\n");
73 }
74 int isEmpty() {
75     return front == NULL;
76 }
```

BHAVLEEN KAUR
102155010
3EIC-2

OUTPUT:

```
Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter data to insert: 56

Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 78
Invalid choice!

Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Queue elements: 56
```

3. Write a menu driven program with 4 options (Insert, Delete, Display, and Exit) to demonstrate the working of Circular Queues (arrays)

```
1 #include <stdio.h>
2 #define MAX_SIZE 100
3 int queue[MAX_SIZE];
4 int front = -1, rear = -1;
5 void enqueue(int data);
6 int dequeue();
7 void display();
8 int isFull();
9 int isEmpty();
10 int main() {
11     int choice, data;
12     while (1) {
13         printf("\nMenu:\n");
14         printf("1. Insert\n");
15         printf("2. Delete\n");
16         printf("3. Display\n");
17         printf("4. Exit\n");
18         printf("Enter your choice: ");
19         scanf("%d", &choice);
20         switch (choice) {
21             case 1:
22                 printf("Enter data to insert: ");
23                 scanf("%d", &data);
24                 if (isFull()) {
25                     printf("Queue Overflow\n");
26                 } else {
27                     enqueue(data);
28                 }
29                 break;
30             case 2:
31                 data = dequeue();
32                 if (data != -1) {
33                     printf("Deleted element: %d\n", data);
34                 }
35                 break;
36             case 3:
37                 display();
38                 break;
39             case 4:
40                 printf("Exiting...\n");
41                 exit(0);
42             default:
```

```
72             default:
43                 printf("Invalid choice!\n");
44             }
45         }
46         return 0;
47     }
48     void enqueue(int data) {
49         if (isEmpty()) {
50             front = rear = 0;
51         } else {
52             rear = (rear + 1) % MAX_SIZE;
53         }
54
55         if (isFull()) {
56             printf("Queue Overflow\n");
57             return;
58         }
59
60         queue[rear] = data;
61     }
62     int dequeue() {
63         if (isEmpty()) {
64             printf("Queue Underflow\n");
65             return -1;
66         }
67         int data = queue[front];
68         if (front == rear) {
69             front = rear = -1;
70         } else {
71             front = (front + 1) % MAX_SIZE;
72         }
73         return data;
74     }
75     // Function to display the queue elements
76     void display() {
77         if (isEmpty()) {
78             printf("Queue is empty!\n");
79             return;
80         }
```