

6.1.1

```
1  #include<stdio.h>
2  #define N 10
3  int ar[N];
4  int front=-1;
5  int rear=-1;
6
7  void enqueue(int x){
8      if(rear==N-1){
9          printf("Queue is full.\n");
10     }
11     if(front==-1&&rear==-1){
12         front=0;
13         rear=0;
14     }
15     else{
16         rear++;
17     }
18     ar[rear]=x;
19     printf("Successfully inserted.\n");
20 }
21 void dequeue(){
22     int pop;
23     if((rear==-1&&front==-1)||front>rear){
24         printf("Queue is underflow.\n");
25     }
26     else{
27         pop=ar[front];
28         printf("Deleted element = %d\n",pop);
29         front++;
```

```

31     }
32     v void display(){
33         —>int i;
34     v —>if((front== -1 & rear== -1) || (front > rear)){
35         —>—>printf("Queue is empty.\n");
36         —>}
37     v —>else{
38         —>—>printf("Elements in the queue : ");
39     v —>—>for(i=front; i<=rear; i++){
40         —>—>—>printf("%d ", ar[i]);
41         —>—>}
42         —>—>printf("\n");
43         —>}
44     }
45     v void isEmpty(){
46     v —>if((front== -1 & rear== -1) || (front > rear)){
47         —>—>printf("Queue is empty.\n");
48         —>}
49     v —>else{
50         —>—>printf("Queue is not empty.\n");
51         —>}
52     }
53     v void size(){
54     v —>if(front== -1 & rear== -1){
55         —>—>printf("Queue size : 0\n");
56         —>}
57     v —>else{
58         —>—>printf("Queue size : %d\n", rear-front+1);
59         —>}
60     }

```

6.1.2

```

1  #include<stdio.h>
2  #define N 5
3  int a[N];
4  int front=-1;
5  int rear=-1;
6  int i;
7  v void enqueue(int x){
8  v   —> if(front== -1 && rear== -1){
9       —> —> front=0;
10      —> —> rear=0;
11      —> —> a[rear]=x;
12      —> —> printf("Successfully inserted.\n");
13      —> }
14  v   —> else if((rear+1)%N==front){
15      —> —> printf("Circular queue is overflow.\n");
16      —> }
17  v   —> else{
18      —> —> rear=(rear+1)%N;
19      —> —> a[rear]=x;
20      —> —> printf("Successfully inserted.\n");
21      —> }
22  }
23  v void dequeue(){
24  v   —> if(front== -1 && rear== -1){
25      —> —> printf("Circular queue is underflow.\n");
26      —> }
27  v   —> else if(front==rear){
28      —> —> printf("Deleted element = %d\n", a[front]);
29      —> —> front=-1;

```

```

30     —> —> rear=-1;
31     —> }
32     v —> else{
33         —> —> printf("Deleted element = %d\n",a[front]);
34         —> —> front=(front+1)%N;
35         —> }
36     }
37     v void display(){
38     v —> if(front== -1){
39         —> —> printf("Circular queue is empty.\n");
40         —> }
41     v —> else{
42         —> —> printf("Elements in the circular queue : ");
43         —> —> i=front;
44         —>
45     v —> do{
46         —> —> printf("%d ",a[i]);
47         —> —> i=(i+1)%N;
48         —> }while(i!=rear);
49         —> printf("%d\n",a[rear]);
50         —> }
51     }
52     v void isEmpty(){
53     v —> if(front== -1 && rear== -1){
54         —> —> printf("Circular queue is empty.\n");
55         —> }
56     v —> else{
57         —> —> printf("Circular queue is not empty.\n");
58         —> }
59     }

```

```

60     v void size(){
61     v —> if(front== -1){
62         —> —> printf("Circular queue size : 0\n");
63         —> }
64     v —> else if(front<rear){
65         —> —> printf("Circular queue size : %d\n",rear-front+1);
66         —> }
67     v —> else if(front>rear){
68         —> —> printf("Circular queue size : %d\n",rear+front);
69         —> }
70     v —> else if(front==rear){
71         —> —> printf("Circular queue size : %d\n",1);
72         —> }
73     }

```

6.1.3

```

1  #include<stdio.h>
2  #define N 10
3  int a[N];
4  int rear=-1, front=-1;
5  v void inject(int x){
6  v   —> if(rear==N-1){
7  —> —> printf("Double ended queue is overflow\n");
8  —> }
9  v   —> else{
10 —> —> rear++;
11 —> —> a[rear]=x;
12 v   —> —> if(front==-1){
13 —> —> —> front=0;
14 —> —> }
15 —> —> printf("Successfully inserted at rear side.\n");
16 —> }
17 }
18 v void eject(){
19 v   —> if(rear==-1){
20 —> —> printf("Double ended queue is underflow.\n");
21 —> }
22 v   —> else{
23 —> —> printf("Deleted element from the rear side = %d\n", a[rear]);
24 v   —> —> if(front==rear){
25 —> —> —> rear=-1;
26 —> —> —> front=-1;
27 —> —> }
28 v   —> —> else{
29 —> —> —> rear--;

```

```

30 —> —> }
31 —> }
32 }
33 v void display(){
34 —> int i;
35 v   —> if(front==-1 && rear==-1){
36 —> —> printf("Double ended queue is empty.\n");
37 —> }
38 v   —> else{
39 —> —> printf("Elements in the double ended queue : ");
40 v   —> —> for(i=front; i<=rear; i++){
41 —> —> —> printf("%d ", a[i]);
42 —> —> }
43 —> —> printf("\n");
44 —> }
45 }

```

6.2.1

```
1  #include<stdio.h>
2  #define N 10
3  int a[N];
4  int front=-1;
5  int rear=-1;
6  v void enqueue(int x){
7  v   →if(rear==N-1){
8      →→printf("Queue is overflow.\n");
9      →}
10 v   →if(front==-1&&rear==-1){
11      →→front=0;
12      →→rear=0;
13      →→a[rear]=x;
14      →→printf("Successfully inserted.\n");
15      →}
16 v   →else{
17      →→rear++;
18      →→a[rear]=x;
19      →→printf("Successfully inserted.\n");
20      →}
21   }
22 v void dequeue(){
23     →int pop;
24 v   →if((front==-1&&rear==-1)||front>rear){
25     →→printf("Queue is underflow.\n");
26     →}
27 v   →else{
28     →→pop=a[front];
29     →→printf("Deleted element = %d\n",pop);
```

```

30     —>—>front++;
31     —>}
32 }
33 v void display(){
34     —>int i;
35 v —>if(front== -1 && rear== -1){
36     —>—>printf("Queue is empty.\n");
37     —>}
38 v —>else{
39     —>—>printf("Elements in the queue : ");
40 v —>—>for(i=front; i<=rear; i++){
41     —>—>—>printf("%d ", a[i]);
42     —>—>}
43     —>—>printf("\n");
44     —>}
45 }

```

6.2.2

```

1     #include<stdio.h>
2     #define N 10
3     int a[N];
4     int front=-1, rear=-1;
5 v void enqueue(int x){
6 v —>if(front== -1 && rear== -1){
7     —>—>front=0;
8     —>—>rear=0;
9     —>—>a[rear]=x;
10    —>—>printf("Successfully inserted.\n");
11    —>}
12 v —>else if((rear+1)%N==front){
13    —>—>printf("Circular queue is overflow.\n");
14    —>}
15 v —>else{
16    —>—>rear=(rear+1)%N;
17    —>—>a[rear]=x;
18    —>—>printf("Successfully inserted.\n");
19    —>}
20 }
21 v void dequeue(){
22 v —>if(front== -1 && rear== -1){
23    —>—>printf("Circular queue is underflow.\n");
24    —>}
25 v —>else if(front==rear){
26    —>—>printf("Deleted element : %d\n", a[front]);
27    —>—>front=-1;
28    —>—>rear=-1;
29    —>—>—>

```

```

30     —> }
31 v —> else{
32     —> —> printf("Deleted element = %d\n", a[front]);
33     —> —> front=(front+1)%N;
34     —> }
35 }
36 v void display(){
37     —> int i;
38 v —> if(front== -1){
39     —> —> printf("Circular queue is empty.\n");
40     —> }
41 v —> else{
42     —> —> printf("Elements in the circular queue : ");
43     —> —> i=front;
44 v —> —> while(i!=rear){
45     —> —> —> printf("%d ", a[i]);
46     —> —> —> i=(i+1)%N;
47     —> —> }
48     —> —> printf("%d\n", a[rear]);
49     —> }
50 }

```

6.2.3


```

1  #include<stdio.h>
2  int frent=10,rear=9,arr[20];
3  v void insertFront(int data){
4
5      —>frent--;
6      —>arr[frent]=data;
7      —>printf("Inserted %d at front.\n",arr[frent]);
8  }
9  v void insertRear(int data){
10     —>rear++;
11     —>arr[rear]=data;
12     —>printf("Inserted %d at rear.\n",arr[rear]);
13 }
14 v void deleteFront(){
15 v —>if(frent>rear){
16     —>—>printf("Deque is empty.\n");
17     —>}
18 v —>else{
19     —>—>printf("Deleted %d from front.\n",arr[frent]);
20     —>—>frent++;
21     —>}
22 }
23 v void deleteRear(){
24 v —>if(frent>rear){
25     —>—>printf("Deque is empty.\n");
26     —>}
27 v —>else{
28     —>—>printf("Deleted %d from rear.\n",arr[rear]);
29     —>—>rear--;

```

```

30     —>}
31 }
32 v void display(){
33 v —>if(frent>rear){
34     —>—>printf("Deque is empty.\n");
35     —>}
36 v —>else{
37     —>—>printf("Deque: ");
38 v —>—>for(int i=frent;i<=rear;i++){
39     —>—>—>printf("%d ",arr[i]);
40     —>—>}
41     —>—>printf("\n");
42     —>}
43 }
44

```