Name: Arnav Thakare

Batch and Roll No.: B-29

Queue ADT

Code:

```
#include <stdio.h>
#include <stdlib.h>
struct queue {int *arr; int r; int f; int l;};
void insert(struct queue *q, int ele);
int delete(struct queue *q);
void is_empty(struct queue *q);
void is_full(struct queue *q);
int front_val(struct queue *q);
int rear_val(struct queue *q);
void display(struct queue *q);
int main()
{
   struct queue q; int c, ele, e;
   q.f=-1; q.r=-1;
   printf("Enter the size of queue: ");
   scanf("%d", &q.l);
   q.arr=(int*) malloc(q.l*sizeof(int));
while(1)
{
   printf("Enter choice (1.Insert 2.Delete 3.is_empty 4.is_full 5.front_value 6.rear_value
7.Display 8.Exit): ");
   scanf("%d", &c);

   switch(c)
   {
```

```
case 1:
printf("Enter element to insert: ");
scanf("%d", &ele);
insert(&q, ele);
break;
case 2:
e=delete(&q);
if (e==-1)
printf("Queue empty...\n");
else
{
   printf("%d element deleted from the queue.\n",e);
}
break;
case 3:
is_empty(&q);
break;
case 4:
is_full(&q);
break;
case 5:
e=front_val(&q);
if (e==-1)
printf("Queue empty...\n");
else
{
   printf("%d is the first element.\n",e);
}
break;
```

```
        case 6:
        e=rear_val(&q);
        if (e==-1)
        printf("Queue empty...\n");
        else
        {
           printf("%d is the rear element.\n",e);
        }
        break;
        case 7:
        display(&q);
        break;
        case 8:
        exit(0);
    }
}
}
void insert(struct queue *q, int ele)
{
    if(q->r==q->l-1)
    {
        printf("Queue full...\n");
    }
    else
    {
        q->r++;
        q->arr[q->r]=ele;
    }
}
```

```
int delete(struct queue *q)
{
   int e;
   if (q->f==-1&& q->r==-1)
   return(-1);
   if(q->f==q->r)
   {
      q->f=-1; q->r=-1;
   }
   else
   {
      e=q->arr[q->f+1];
      q->f++;
      return(e);
   }
}
void is_empty(struct queue *q)
{
   if ((q->f==-1)&&(q->r==-1))
   {
      printf("Queue empty...\n");
   }
   else
   {
      printf("Queue is not empty...\n");
   }
}
void is_full(struct queue *q)
{
```

```
    if(q->r==q->l-1)
    {
        printf("Queue full...\n");
    }
    else
    {
        printf("Queue not full...\n");
    }
}
int front_val(struct queue *q)
{
    return(q->arr[q->f+1]);
}
int rear_val(struct queue *q)
{
    if(q->r==-1)
    {
    return(-1);
    }
    else
    {
        return(q->arr[q->r]);
    }
}
void display(struct queue *q)
{
    int i=q->f+1;
    while(i!=(q->r+1))
    {
```

```
        printf("%d ", q->arr[i]);
        i++;
    }
    printf("\n");
}
```

Output: