```
/****Implementation of Deque Using Array****/

#include <stdio.h>
#include <conio.h>
#define MAXSIZE 10
int queue[MAXSIZE];
int front,rear;
void enqueue_front()
{
 int n;
 printf("Enter item to be inserted : ");
 scanf("%d",&n);
 if (((front==0) && (rear==MAXSIZE-1)) || (front==(rear+1)))
   printf("Queue is Full|n");
 else if (front==-1 && rear==-1)
 {
   front=rear=0;
   queue[front]=n;
 }
 else if (front==0)
 {
   front=MAXSIZE-1;
   queue[front]=n;
 }
 else
 {
   front=front-1;
   queue[front]=n;
 }
}

void enqueue_rear()
{
 int n;
 printf("Enter item to be inserted : ");
 scanf("%d",&n);
 if (((front==0) && (rear==MAXSIZE-1)) || (front>rear))
   printf("Queue is Full|n");
 else if (front==-1 && rear==-1)
 {
   front=rear=0;
   queue[rear]=n;
 }
 else if (rear==MAXSIZE-1)
 {
   rear=0;
   queue[rear]=n;
 }
```

```
 else
 {
   rear=rear+1;
   queue[rear]=n;
 }
}

void dequeue_front()
{
  if (front==-1)
     printf("Queue is empty\n");
  else if(rear==front)
     front=rear=-1;
  else if (front==MAXSIZE-1)
    front=0;
  else
   front=front+1;
}

void dequeue_rear()
{
  if (rear==-1)
     printf("Queue is empty\n");
  else if(rear==front)
     front=rear=-1;
  else if (rear==0)
    rear=MAXSIZE-1;
  else
    rear=rear-1;
}
void display()
{
 int i;
 for(i=front;i!=rear;i=(i+1)%MAXSIZE)
    printf("%d\t",queue[i]);
 printf("%d\n",queue[i]);
}

void main()
{
  int opt;
  clrscr();
  front=rear=-1;
  while(1)
  {
     printf("\n1.Enqueue at front\n2.Enqueue at rear\n");
     printf("3.Dequeue at front\n4.Dequeue at raar\n");
     printf("5.Display\n6.Exit\n");
```

```
    printf("Enter your option : ");
    scanf("%d",&opt);
    switch (opt)
    {
          case 1:
             enqueue_front();
             break;
          case 2:
            enqueue_rear();
            break;
          case 3:
            dequeue_front();
            break;
          case 4:
            dequeue_rear();
            break;
          case 5:
            display();
            break;
          case 6:
            exit(0);
          default:
            printf("Invalid option\n");
        }
  }
}


/***Implementation of Deque Using Doubly Linked List ***/

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
struct node
{
 int data;
 struct node *next,*prev;
};
struct node *head=NULL,*tail=NULL;
void enqueue_front()
{
 struct node *newnode;
 int n;
 newnode=(struct node*)malloc(sizeof(struct node));
 printf("Enter Data : ");
 scanf("%d",&n);
 newnode->data=n;
 if (head==NULL)
```

```
 {
    newnode->next=newnode->prev=NULL;
    head=tail=newnode;
 }
 else
 {
   newnode->next=head;
   head->prev=newnode;
   newnode->prev=NULL;
   head=newnode;
 }
}
void enqueue_rear()
{
 struct node *newnode;
 int n;
 newnode=(struct node*)malloc(sizeof(struct node));
 printf("Enter Data : ");
 scanf("%d",&n);
 newnode->data=n;
 if (tail==NULL)
 {
    newnode->next=newnode->prev=NULL;
    head=tail=newnode;
 }
 else
 {
   tail->next=newnode;
   newnode->prev=tail;
   newnode->next=NULL;
   tail=newnode;
 }
}

void dequeue_front()
{
  struct node *temp,*prev;
  int n;
  if (head==NULL)
     printf("Queue is empty\n");
  else if (head==tail)
  {
    temp=head;
    head=tail=NULL;
    free(temp);
  }
  else
  {
```

```
      temp=head;
      head=head->next;
      head->prev=NULL;
      free(temp);
   }
}
void dequeue_rear()
{
   struct node *temp,*prev;
   int n;
   if (tail==NULL)
      printf("Queue is empty\n");
   else if (head==tail)
   {
      temp=tail;
      head=tail=NULL;
      free(temp);
   }
   else
   {
      temp=tail;
      tail=tail->prev;
      tail->next=NULL;
      free(temp);
   }
}
void display()
{
  struct node *temp;
  if (head!=NULL)
  {
    temp=head;
    while (temp!=NULL)
    {
      printf("%d\n",temp->data);
      temp=temp->next;
    }
  }
  else
    printf("Queue is empty\n");
}
void main()
{
  int opt;
  clrscr();
  while (1)
  {
```

```
    printf("1.Enqueue at Front\n2.Enqueue at Rear\n3.Dequeue at Front\n4.Dequeue at
Rear\n5.Display\n6.Exit\n");
    printf("Enter ypur option : ");
    scanf("%d",&opt);
    switch(opt)
    {
    case 1:
             enqueue_front();
             break;
    case 2:
            enqueue_rear();
            break;
    case 3:
            dequeue_front();
            break;
    case 4:
            dequeue_rear();
            break;
    case 5:
            display();
            break;
    case 6:
            exit(0);
    default:
            printf("Invalid Option\n");
    }
}
}
```