// Write a program to implement double ended queue (dequeue) using arrays

```
#include<stdio.h>
#include<process.h>
#include<conio.h>
#define MAX 30
typedef struct dequeue
{
int data[MAX];
int rear,front;
}dequeue;
void initialize(dequeue *p);
int empty(dequeue *p);
int full(dequeue *p);
void enqueueR(dequeue *p,int x);
void enqueueF(dequeue *p,int x);
int dequeueF(dequeue *p);
int dequeueR(dequeue *p);
void print(dequeue *p);
void main()
{
  clrscr();
int i,x,op,n;
dequeue q;
initialize(&q);
do
{
printf("\n1.Create\n2.Insert(rear\n3.Insert(front\n4.Delete(rear");
```

```
printf("\n5.Delete(front)\n6.Print\n7.Exit\nEnter your choice:");
scanf("%d",&op);

switch(op)
{
case 1: printf("\nEnter number of elements:");
scanf("%d",&n);
initialize(&q);
printf("\nEnter the data:");
for(i=0;i<n;i++)
{
scanf("%d",&x);
if(full(&q))
{
printf("\nQueue is full!!");
exit(0);
}
enqueueR(&q,x);
}
break;
case 2: printf("\nEnter element to be inserted:");
scanf("%d",&x);
if(full(&q))
{
printf("\nQueue is full!!");
exit(0);
}
enqueueR(&q,x);
break;
```

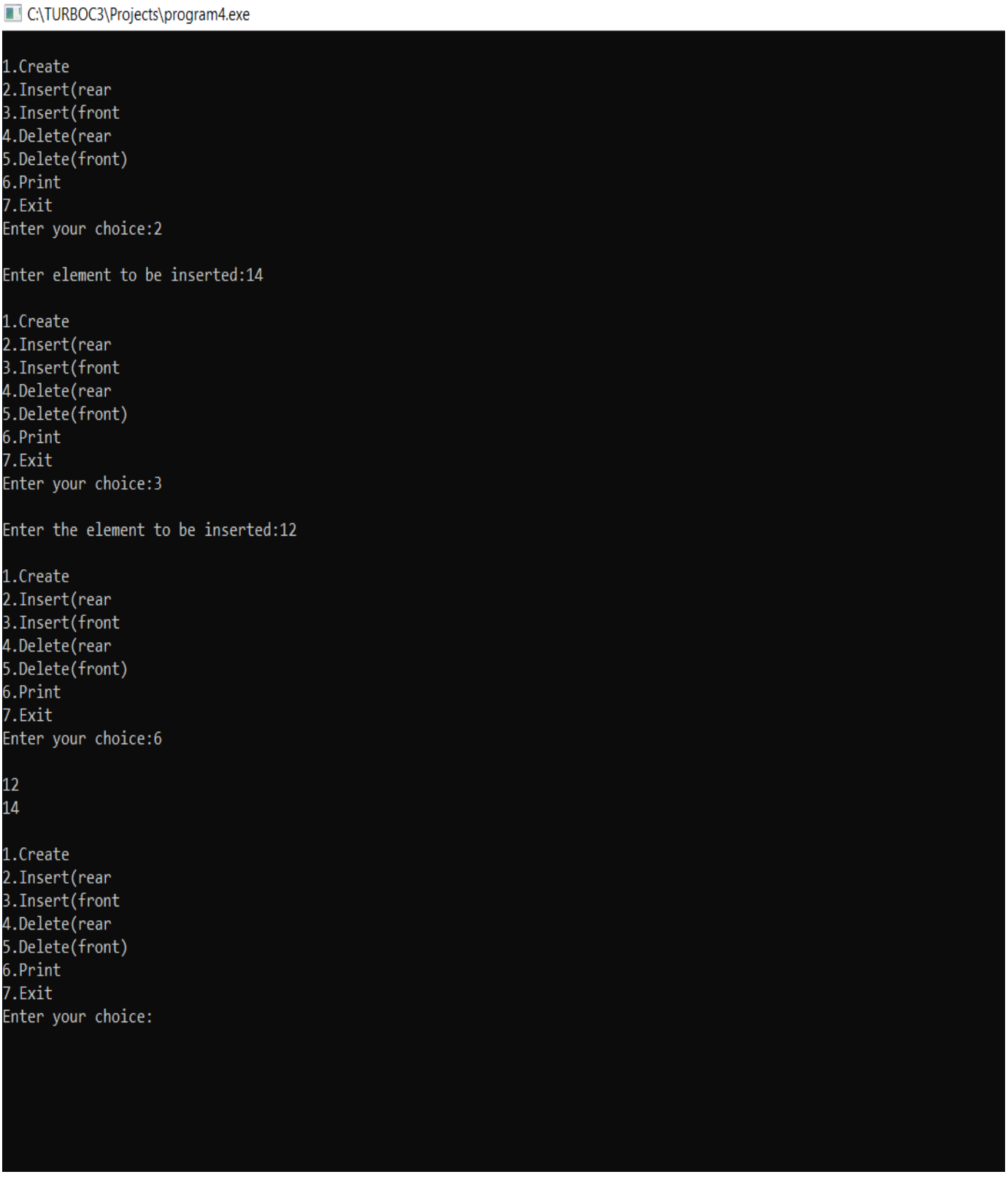
```
case 3: printf("\nEnter the element to be inserted:");
scanf("%d",&x);

if(full(&q))
{
printf("\nQueue is full!!");
exit(0);
}
enqueueF(&q,x);
break;
case 4: if(empty(&q))
{
printf("\nQueue is empty!!");
exit(0);
}
x=dequeueR(&q);
printf("\nElement deleted is %d\n",x);
break;
case 5: if(empty(&q))
{
printf("\nQueue is empty!!");
exit(0);
}
x=dequeueF(&q);
printf("\nElement deleted is %d\n",x);
break;
case 6: print(&q);
break
default: break;
```

```
}
}while(op!=7);

getch();
}
void initialize(dequeue *P)
{
P->rear=-1;
P->front=-1;
}
int empty(dequeue *P)
{
if(P->rear==-1)
return(1);
return(0);
}
int full(dequeue *P)
{
if((P->rear+1)%MAX==P->front)
return(1);
return(0);
}
void enqueueR(dequeue *P,int x)
{
if(empty(P))
{
P->rear=0;
P->front=0;
P->data[0]=x;
```

```
}
else
{
P->rear=(P->rear+1)%MAX;
P->data[P->rear]=x;
}

void enqueueF(dequeue *P,int x)
{
if(empty(P))
{
P->rear=0;
P->front=0;
P->data[0]=x;
}
else
{
P->front=(P->front-1+MAX)%MAX;
P->data[P->front]=x;
}
}
int dequeueF(dequeue *P)

int x;
x=P->data[P->front];
if(P->rear==P->front) //delete the last element
initialize(P);
else
P->front=(P->front+1)%MAX;
```

```
return(x);
}

int dequeueR(dequeue *P)
{
int x;
x=P->data[P->rear];
if(P->rear==P->front)
initialize(P);
else
P->rear=(P->rear-1+MAX)%MAX;
return(x);
}
void print(dequeue *P)
{
if(empty(P))
{
printf("\nQueue is empty!!");
exit(0);
}
int i;
i=P->front;
while(i!=P->rear)
{
printf("\n%d",P->data[i]);
i=(i+1)%MAX;
}
printf("\n%d\n",P->data[P->rear]);
}
```

Output:


C:\TURBOC3\Projects\program4.exe
1.Create
2.Insert(rear
3.Insert(front
4.Delete(rear
5.Delete(front)
6.Print
7.Exit
Enter your choice:2
Enter element to be inserted:14
1.Create
2.Insert(rear
3.Insert(front
4.Delete(rear
5.Delete(front)
6.Print
7.Exit
Enter your choice:3
Enter the element to be inserted:12
1.Create
2.Insert(rear
3.Insert(front
4.Delete(rear
5.Delete(front)
6.Print
7.Exit
Enter your choice:6
12
14
1.Create
2.Insert(rear
3.Insert(front
4.Delete(rear
5.Delete(front)
6.Print
7.Exit
Enter your choice: