DATA STRUCTURE LAB 2020-2021

NAME: Shraddha Patangrao Shinde

Roll no: 19102A0030

Div: A

Sem: 3

Que 1):

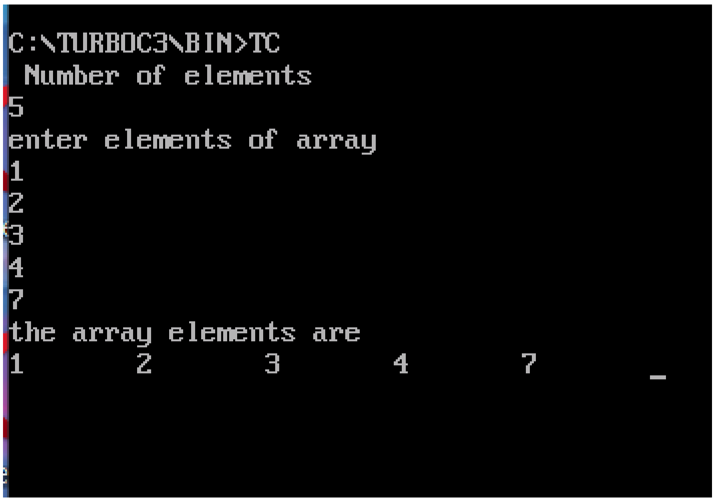
NAME: ARRAY USING POINTER

Code:

```
#include<conio.h>

#include<stdio.h>

void read(int *p,int n)

{

        int i;

        for(i=0;i<n;i++)
```

```
          {
                    scanf("%d",(p+i));
          }
}
void display(int *p,int n)
{
          int i;
          for(i=0;i<n;i++)
          {
                    printf("%d\t",*(p+i));
          }
}
int main()
{
          int a[50],n;
          printf(" Number of elements\n");
          scanf("%d",&n);
          printf("enter elements of array\n");
          read(a,n);
          printf("the array elements are\n");
          display(a,n);
          getch();
          return 0;
}
```

Output:



```
C:\TURBOC3\BIN>TC
 Number of elements
5
enter elements of array
1
2
3
4
7
the array elements are
1       2       3       4       7       _
```

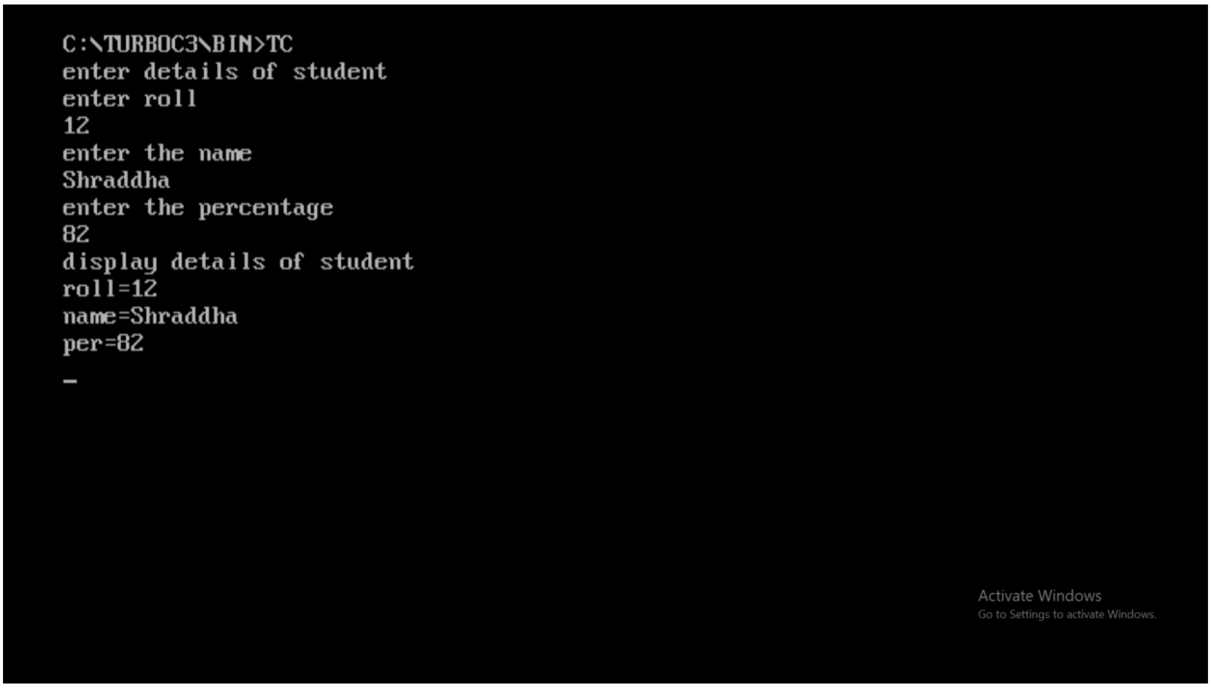
Que 2):

STRUCTURE MEMBER WITH POINTER

Code:

```
#include<stdio.h>

typedef struct

{

        int roll;

        char name[20];

        int per;

}student;

int main()

{

        student s1;

        student *p;

        p=&s1;
        printf("enter details of student\n");

        printf("enter roll\n");

        scanf("%d",&p->roll);

        fflush(stdin);

        printf("enter the name\n");

        gets(p->name);

        printf("enter the percentage\n");

        scanf("%d",&p->per);

        printf("display details of student\n");

        printf("roll=%d\n",p->roll);

        printf("name=%s\n",p->name);

        printf("per=%d\n",p->per);

        return 0;

}
```

OUTPUT:



```
C:\TURBOC3\BIN>TC
enter details of student
enter roll
12
enter the name
Shraddha
enter the percentage
82
display details of student
roll=12
name=Shraddha
per=82
_
```

Que 3):

IMPLEMENT STACK USING ARRAY

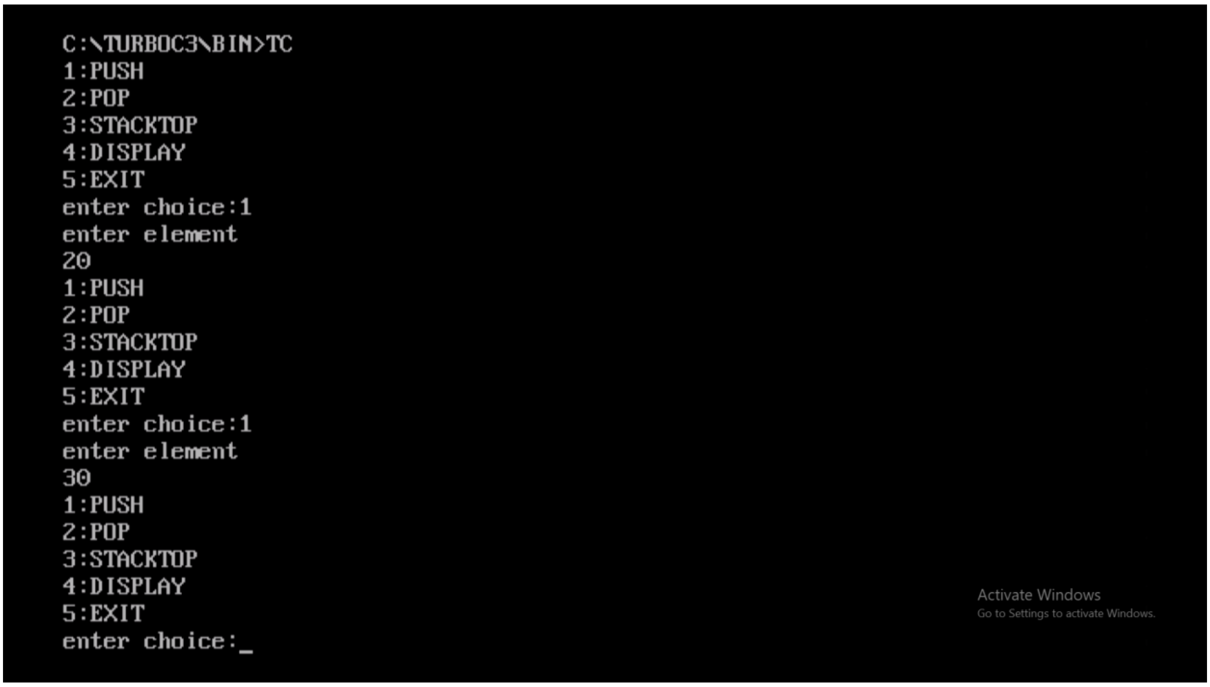
Code:

```
#include<stdio.h>

#define MAX 10

typedef struct

{

        int a[MAX];

        int top;

}stack;

void push(stack *s,int ele)

{

        if(s->top==MAX-1)

        {

                printf("stack overflow\n");

                return;

        }

                s->top++;

                s->a[s->top]=ele;

}

int isempty(stack *s)

{

        if(s->top==-1)

                return 1;
        else
           return 0;
}
int stacktop(stack *s)
{
        if(isempty(s))
                return -1;
        else
                return s->a[s->top];
}
int pop(stack *s)
{
        int x;
        if(isempty(s))
                return -1;
         x=s->a[s->top];
         s->top--;
         return x;
}
void display(stack *s)
{
        int i;
        if(isempty(s))
        {
                printf("stack underflow\n");
                return ;
```

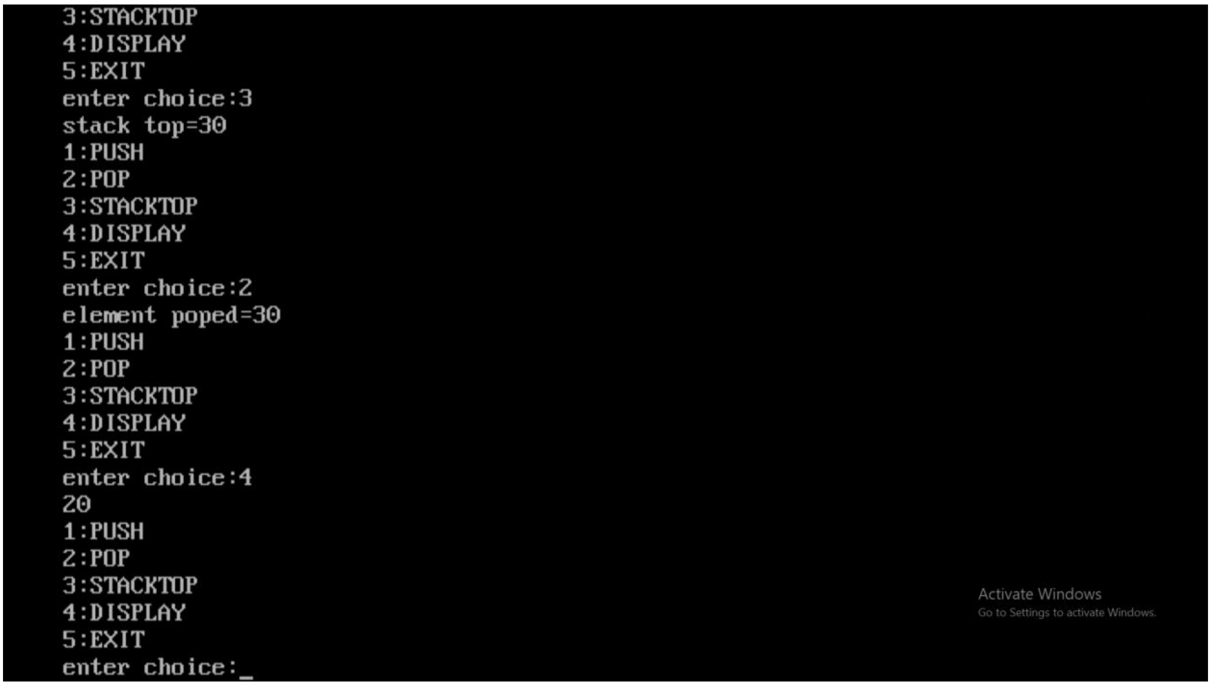
```
    }
    for(i=s->top;i>=0;i--)
        printf("%d\t",s->a[i]);
        printf("\n");
}
int main()
{
    stack s;
    int ele,ch;
    s.top=-1;
    while(1)
    {
        printf("1:PUSH\n2:POP\n3:STACKTOP\n4:DISPLAY\n5:EXIT\n");
        printf("enter choice:");
        scanf("%d",&ch);
        if(ch==5)
        break;
        switch(ch)
        {
            case 1 :printf("enter element\n");
                    scanf("%d",&ele);
                    push(&s,ele);
                    break;
            case 2 : ele=pop(&s);
                        if(ele==-1)
                        printf("stack underflow\n");
```

```
                              else

                              printf("element poped=%d\n",ele);

                              break;

                  case 3 :ele=stacktop(&s);

                              if(ele==-1)

                              printf("stack underflow\n");

                              else

                              printf("stack top=%d\n",ele);

                              break;

                  case 4 :display(&s);

                          break;

                  default :printf("invalid choice\n");

              }

          }

          return 0;

      }
```

OUTPUT:



```
C:\TURBOC3\BIN>TC
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice:1
enter element
20
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice:1
enter element
30
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice:_
```



```
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice:3
stack top=30
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice:2
element poped=30
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice:4
20
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice:_
```

QUE 4)

IMPLEMENT 2 STACK USING SINGLE ARRAY

Code:

```
#include<conio.h>
#include<stdio.h>
#define MAX 10
typedef struct
{
	int a[10];
	int top1,top2;
}stack;
void push1(stack *s,int ele)
{
	if(s->top1+1==s->top2)
	{
		printf("stack 1 overflow\n");
		return;
	}
	s->top1++;
	s->a[s->top1]=ele;
}
void push2(stack *s,int ele)
{
	if(s->top2-1==s->top1)
	{
		printf("stack 2 overflow\n");
		return;
	}
	s->top2--;
```

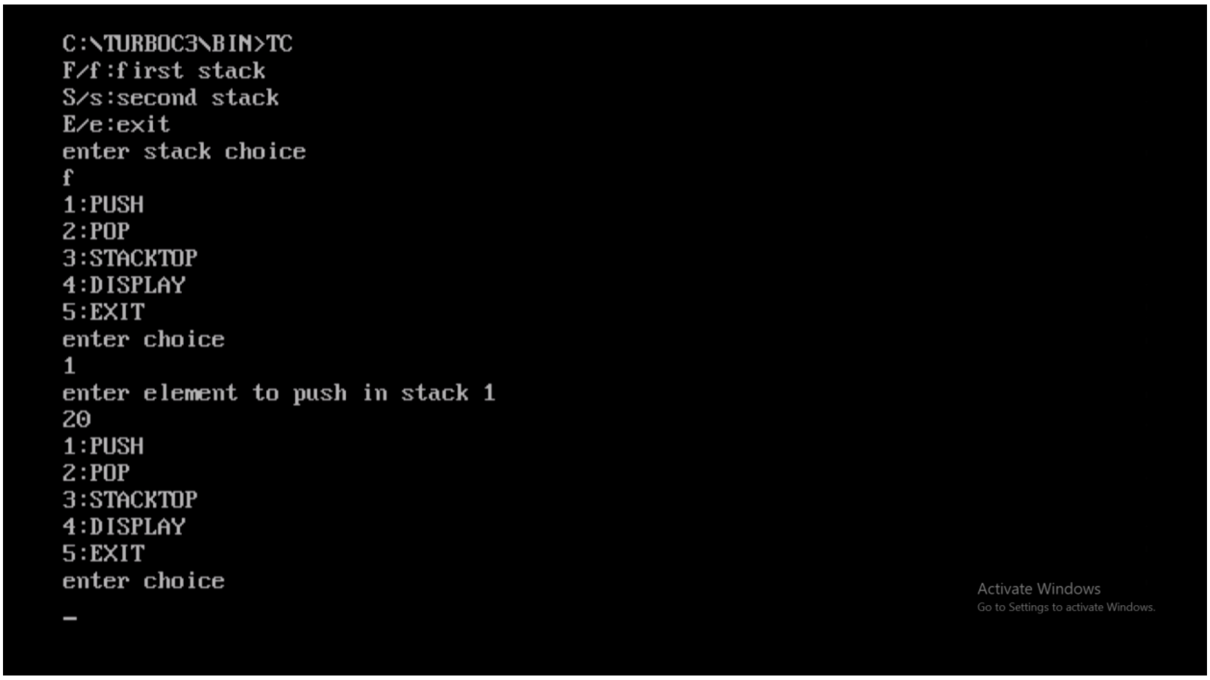
```
        s->a[s->top2]=ele;
}
int isempty1(stack *s)
{
        if(s->top1==-1)
        return 1;
        else
        return 0;
}
int isempty2(stack *s)
{
        if(s->top2==MAX)
        return 1;
        else
        return 0;
}
int stacktop1(stack *s)
{
        if(isempty1(s))
        return -1;
        return s->a[s->top1];
}
int stacktop2(stack *s)
{
        if(isempty2(s))
        return -1;
        return s->a[s->top2];
}
int pop1(stack *s)
{
        int x;
        if(isempty1(s))
        return -1;
        x=s->a[s->top1];
        s->top1--;
        return x;
}
int pop2(stack *s)
{
        int x;
        if(isempty2(s))
        return -1;
        x=s->a[s->top2];
        s->top2++;
        return x;
}
void display1(stack *s)
{
        int i;
        if(isempty1(s))
        {
                printf("stack 1 underflow\n");
                return;
        }
        else
        {
                for(i=s->top1;i>=0;i--)
                printf("%d\t",s->a[i]);
                printf("\n");
        }
}
void display2(stack *s)
{
        int i;
        if(isempty2(s))
        {
                printf("stack 2 underflox\n");
                return;
        }
        else
        {
                for(i=s->top2;i<MAX;i++)
                printf("%d\t",s->a[i]);
                printf("\n");
        }
}
int main()
{
```

```
int ch,ele;

char sch;

stack s;

s.top1=-1;

s.top2=MAX;

while(1)

{

        printf("F/f:first stack\n");

        printf("S/s:second stack\n");

        printf("E/e:exit\n");

        fflush(stdin);

        printf("enter stack choice\n");

        scanf("%c",&sch);

        if(sch=='E'|| sch=='e')

        break;

        if(sch=='F'|| sch=='f')

        {

                while(1)

                {

printf("1:PUSH\n2:POP\n3:STACKTOP\n4:DISPLAY\n5:EXIT\n");

                        printf("enter choice\n");

                        scanf("%d",&ch);

                        if(ch==5)

                        break;

                        switch(ch)
                {
                        case 1:printf("enter element to push in stack 1\n");
                                scanf("%d",&ele);
                                push1(&s,ele);
                                break;
                        case 2:ele=pop1(&s);
                                if(ele==-1)
                                        printf("stack 1 underflow\n");
                                else
                                        printf("element poped from 
stack 1=%d\n",ele);
                                        break;
                        case 3:ele=stacktop1(&s);
                                if(ele==-1)
                                        printf("stack 1 underflow\n");
                                else
                                printf("stack 1 top element=%d\n",ele);
                                        break;
                        case 4: display1(&s);
                                break;
                        default :printf("invalid choice\n");
                }
        }
    }
    else if(sch=='S'||sch=='s')
        {
```

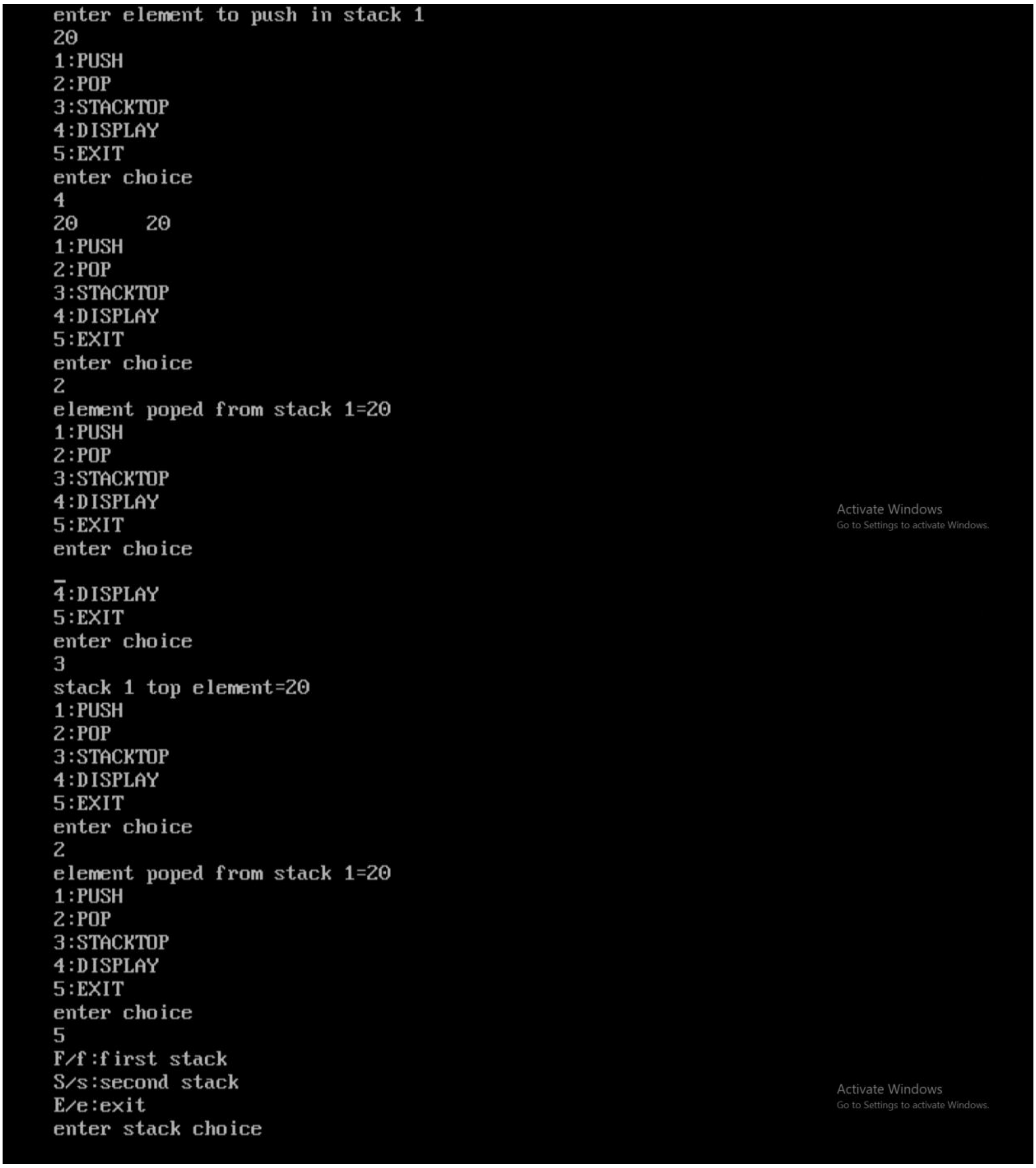
```
                    while(1)
            {

printf("1:PUSH\n2:POP\n3:STACKTOP\n4:DISPLAY\n5:EXIT\n");
                printf("enter choice\n");
                scanf("%d",&ch);
                if(ch==5)
                break;
                switch(ch)
                {
                    case 1:printf("enter element to push in stack 2\n");
                            scanf("%d",&ele);
                            push2(&s,ele);
                            break;
                    case 2:ele=pop2(&s);
                            if(ele==-1)
                                printf("stack 2 underflow\n");
                            else
                                printf("element poped from
stack 2=%d\n",ele);
                                break;
                    case 3:ele=stacktop2(&s);
                            if(ele==-1)
                                printf("stack 2 underflow\n");
                            else
                            printf("stack 2 top element=%d\n",ele);
                                break;
```

```
                                        case 4: display2(&s);
                                                    break;
                                        default :printf("invalid choice\n");
                                  }
                            }
                      }

      else
                      printf("invalid stack choice\n");
  }
getch();
return 0;
}
```

OUTPUT:



```
C:\TURBOC3\BIN>TC
F/f:first stack
S/s:second stack
E/e:exit
enter stack choice
f
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice
1
enter element to push in stack 1
20
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice
_
```



```
enter element to push in stack 1
20
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice
4
20      20
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice
2
element poped from stack 1=20
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice
_
4:DISPLAY
5:EXIT
enter choice
3
stack 1 top element=20
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice
2
element poped from stack 1=20
1:PUSH
2:POP
3:STACKTOP
4:DISPLAY
5:EXIT
enter choice
5
F/f:first stack
S/s:second stack
E/e:exit
enter stack choice
```

QUE 5)

INFIX TO POSTFIX

Code:

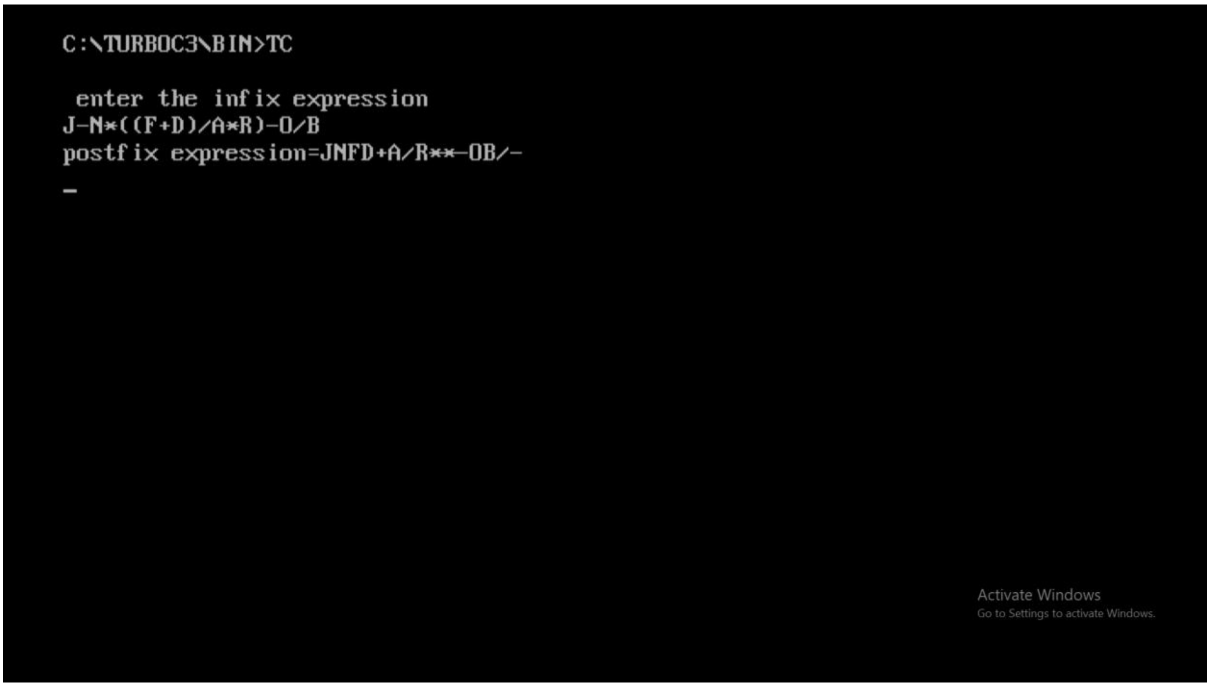
```
#include<conio.h>

#include<stdio.h>

#include<string.h>

#define MAX 50

typedef struct

{

        int a[MAX];

        int top;

}stack;

void push(stack *s,int ele)

{

        s->top++;

        s->a[s->top]=ele;

}

int pop(stack *s)

{

        int x;

        x = s->a[s->top];

        s->top--;

        return x;

}

int isempty(stack *s)

{

        if(s->top==-1)

                return 1;

        else
```

```
                return 0;
}
int stacktop(stack *s)
{
        return s->a[s->top];
}
int isoperand(char x)
{
        if((x>='A' && x<='Z')||(x>='a' && x<='z')||(x>='0' && x<='9'))
        return 1;
        else
        return 0;
}
int icp(int x)
{
        switch(x)
        {
                case'+':
                case'-':return 1;
                case'*':
                case'%':
                case'/':return 2;
        }
        return -1;
}
int isp(int x)
```

```
{
        switch(x)
        {
                case'(':return 0;
                case'+':
                case'-':return 1;
                case'*':
                case'%':
                case'/':return 2;
        }
        return -1;
}
void convert(char infix[],char post[])
{
        stack s;
        s.top=-1;
        int i,k=0;
        int x,ele;
        for(i=0;i<strlen(infix);i++)
                {
                        x=infix[i];
                        if(x=='(')
                        push(&s,x);
                        else if(x==')')
                        {
                                while(1)
                                {
                                    ele=pop(&s);
                                    if(ele=='(')
                                    break;
                                    post[k++]=ele;
                                }
                        }
                        else if(isoperand(x))
                                    post[k++]=x;
                        else if(isempty(&s))
                                    push(&s,x);
                        else if(icp(x)>isp(stacktop(&s)))
                                    push(&s,x);
                        else
                        {
                                while((isempty(&s)==0) && (icp(x)<=isp(stacktop(&s))))
                                        {
                                            ele=pop(&s);
                                            post[k++]=ele;
                                        }
                                        push(&s,x);
                        }
                }
                while(isempty(&s)==0)
                {
                        ele=pop(&s);
```

```
                        post[k++]=ele;

                }

                post[k]='\0';

}

int main()

{

        char infix[50];

        char post[50];

        printf("\n enter the infix expression\n");

        gets(infix);

        convert(infix,post);

        printf("postfix expression=%s\n",post);

    getch();

        return 0;

}
```

OUTPUT:



```
C:\TURBOC3\BIN>TC

 enter the infix expression
J-N*((F+D)/A*R)-O/B
postfix expression=JNFD+A/R**-OB/-
_
```

QUE 6)

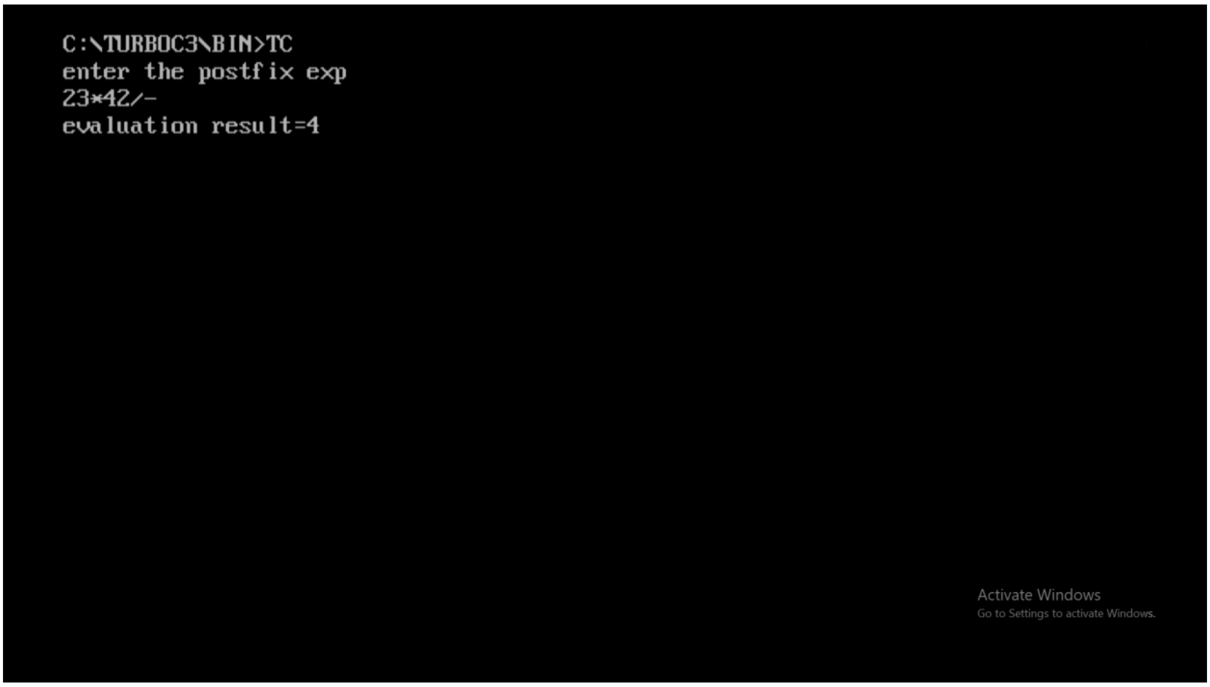
EVALUATE POSTFIX EXP:

Code:

```
#include<conio.h>

#include<stdio.h>

#include<string.h>

#define MAX 50

typedef struct

{

        int a[MAX];

        int top;

}stack;

void push(stack *s,int ele)

{

        s->top++;

        s->a[s->top]=ele;

}

int pop(stack *s)

{

        int x;

        x = s->a[s->top];

                s->top--;

                return x;

}

int isoperand(char x)
```

```
{
    if(x>='0' && x<='9')
    return 1;
    else
    return 0;
}
int EvaluatePostfix(char post[])
{
    stack s;
    s.top=-1;
    int op1,op2,v,i;
    char x;
    for(i=0;i<strlen(post);i++)
    {
        x=post[i];
        if(isoperand(x))
        {
            push(&s,int(x)-'0');
        }
        else
        {
            op2 = pop(&s);
            op1 = pop(&s);
            switch(x)
            {
                case'+':v=op1+op2;break;
                case'-':v=op1-op2;break;

                case'*':v=op1*op2;break;

                case'/':v=op1/op2;break;

                case'%':v=op1%op2;break;

            }

            push(&s,v);

        }

    }

    return pop(&s);

}

int main()

{

    char post[50];

    printf("enter the postfix exp\n");

    gets(post);

    printf("evaluation result=%d\n",EvaluatePostfix(post));

getch();

return 0;

}
```

OUTPUT:



```
C:\TURBOC3\BIN>TC
enter the postfix exp
23*42/-
evaluation result=4
```

QUE 7)

CORRECTNESS OF PARANTHESIS

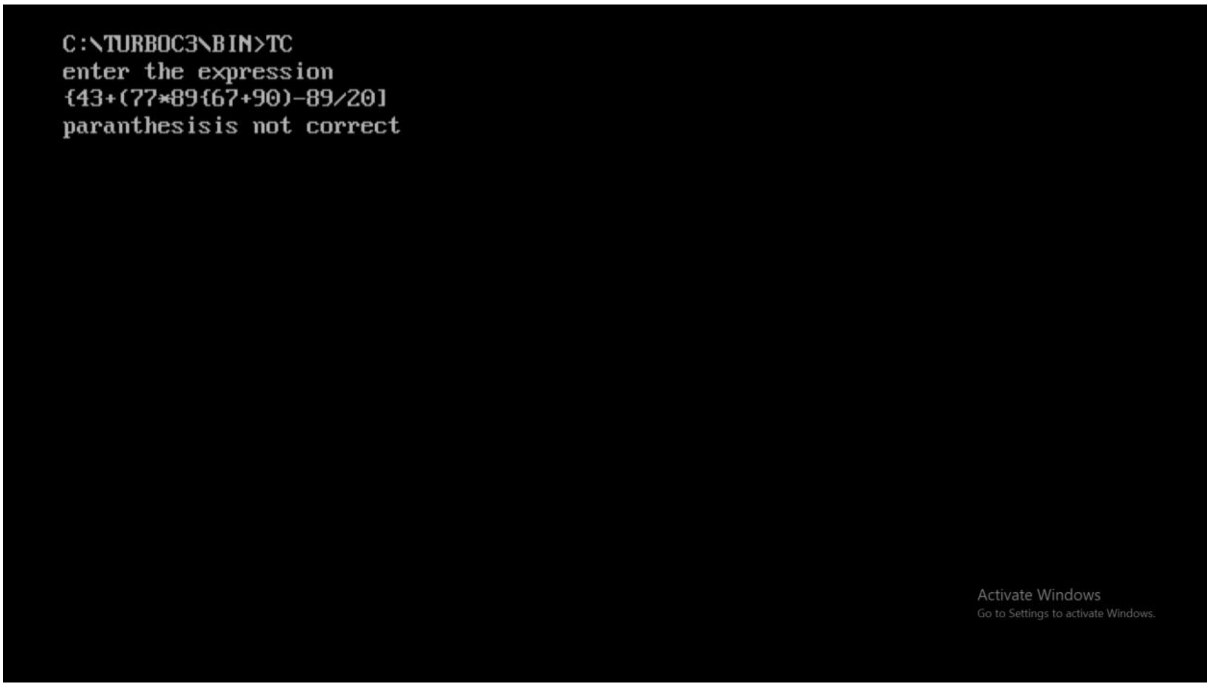
Code:

```
#include<conio.h>

#include<stdio.h>

#include<string.h>

#define MAX 50

typedef struct

{

	int a[50];

	int top;
```

```
}stack;

void push(stack *s,int ele)

{

        s->top++;

        s->a[s->top]=ele;

}

int pop(stack *s)

{

        int x;

        x=s->a[s->top];

        s->top--;

        return x;

}

int isopenbracket(char x)

{

        if(x=='(' || x=='[' ||x=='{')

        return 1;

        else

        return 0;

}

int isclosebracket(char x)

{

        if(x==')' || x==']' ||x=='}')

        return 1;

        else

        return 0;
```

```
}
int isempty(stack *s)
{
        if(s->top==-1)
        return 1;
        else
        return 0;
}
int checkkaro(char *exp)
{
        stack s;
        s.top=-1;
        int x,ele,i;
        for(i=0;i<strlen(exp);i++)
                {
                        x=exp[i];
                        if(isopenbracket(x))
                        push(&s,x);
                        else if(isclosebracket(x))
                        {
                                if(isempty(&s))
                                return 0;
                                else
                                {
                                        ele=pop(&s);
                                        if((x==')') && (ele!='('))
}

                    return 0;

                    else if((x==']') && (x!='['))

                    return 0;

                    else if((x=='}') && (ele!='{'))

                    return 0;

                }

            }

        }

        if(isempty(&s))

        return 1;

        else

        return 0;

}

int main()

{

    char exp[50];

    printf("enter the expression\n");

    gets(exp);

    if(checkkaro(exp))

    printf("paranthesis is correct\n");

    else

    printf("paranthesisis not correct\n");

  getch();

    return 0;

}
```

OUTPUT:



```
C:\TURBOC3\BIN>TC
enter the expression
{43+(77*89{67+90)-89/20]
paranthesisis not correct
```

QUE 8)

IMPLEMENT QUEUE USING ARRAY

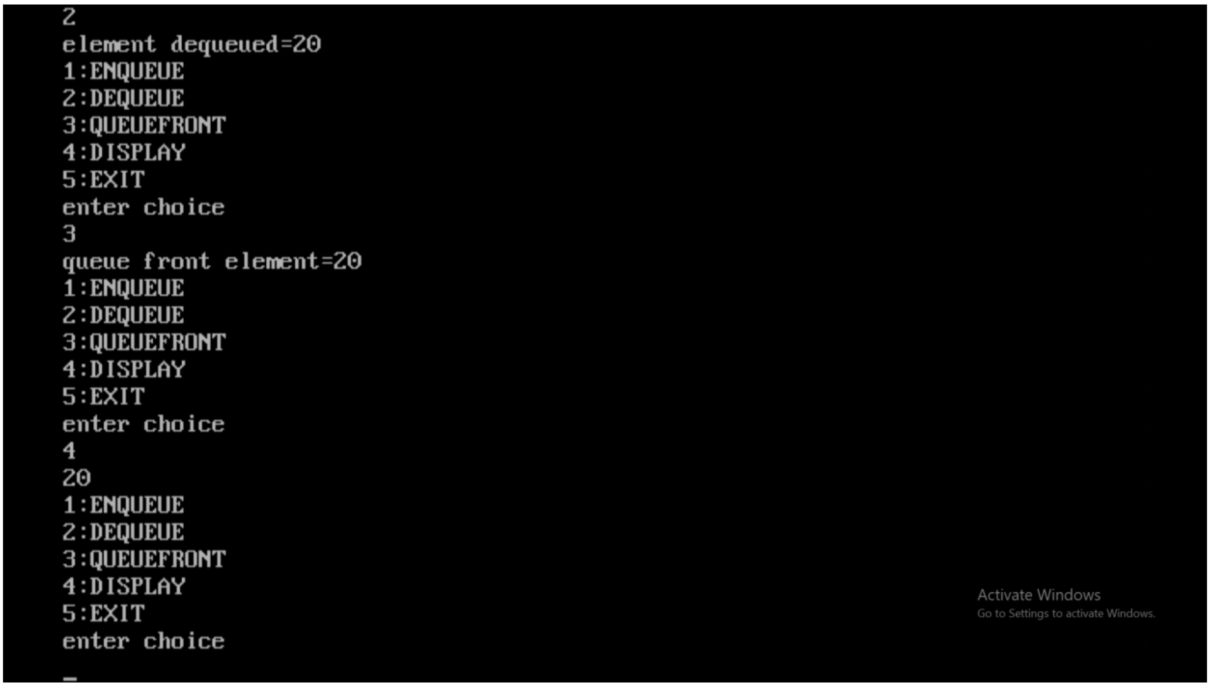
Code:

```
#include<conio.h>

#include<stdio.h>

#define MAX 5

typedef struct

{

        int a[MAX];

        int front,rear;

}queue;

void enqueue(queue *q,int ele)

{

        if(q->rear==MAX-1)
        {
                printf("queue overflow\n");
                return;
        }
        q->rear++;
        q->a[q->rear]=ele;
}
int isempty(queue *q)
{
        if(q->rear+1==q->front)
        return 1;
        else
        return 0;
}
int dequeue(queue *q)
{
        int x;
        if(isempty(q))
        return -1;
        x=q->a[q->front];
        q->front++;
        return x;
}
int queuefront(queue *q)
{
        if(isempty(q))
        return -1;
        else
        return q->a[q->front];
}
void display(queue *q)
{
        int i;
        if(isempty(q))
        {
                printf("queue underflow\n");
                return;
        }
        else
        {
                for(i=q->front;i<=q->rear;i++)
                printf("%d\t",q->a[i]);
                printf("\n");
        }
}
int main()
{
        int ch,ele;
        queue q;
        q.front=0;
        q.rear=-1;
        while(1)
```

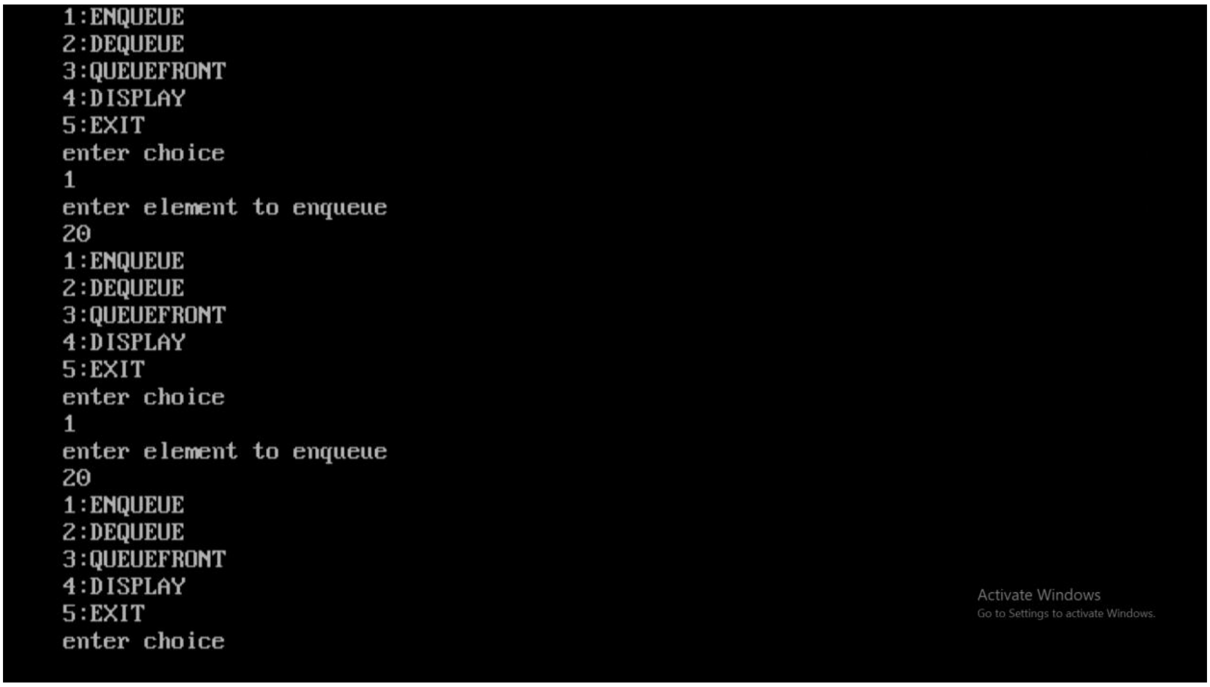
```
            {

        printf("1:ENQUEUE\n2:DEQUEUE\n3:QUEUEFRONT\n4:DISPLAY\n5:EXIT\n");
                    printf("enter choice\n");
                    scanf("%d",&ch);
                    if(ch==5)
                    break;
                    switch(ch)
                    {
                        case 1:printf("enter element to enqueue\n");
                                scanf("%d",&ele);
                                enqueue(&q,ele);
                                break;
                        case 2:ele=dequeue(&q);
                                if(ele==-1)
                              printf("queue underflow\n");
                                else
                                printf("element dequeued=%d\n",ele);
                                break;
                        case 3:ele=queuefront(&q);
                                    if(ele==-1)
                                printf("queue underflow\n");
                                    else
                                    printf("queue front
    element=%d\n",ele);
                                    break;
                        case 4:display(&q);
```

```
                                        break;
                            default :printf("invalid choice\n");
                    }
            }
      getch();
            return 0;
}
```

OUTPUT:



```
2
element dequeued=20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
3
queue front element=20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
4
20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
_
```



```
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
1
enter element to enqueue
20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
1
enter element to enqueue
20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
```

QUE 9)

IMPLEMENT CIRCULAR QUEUE USING ARRAY

Code:

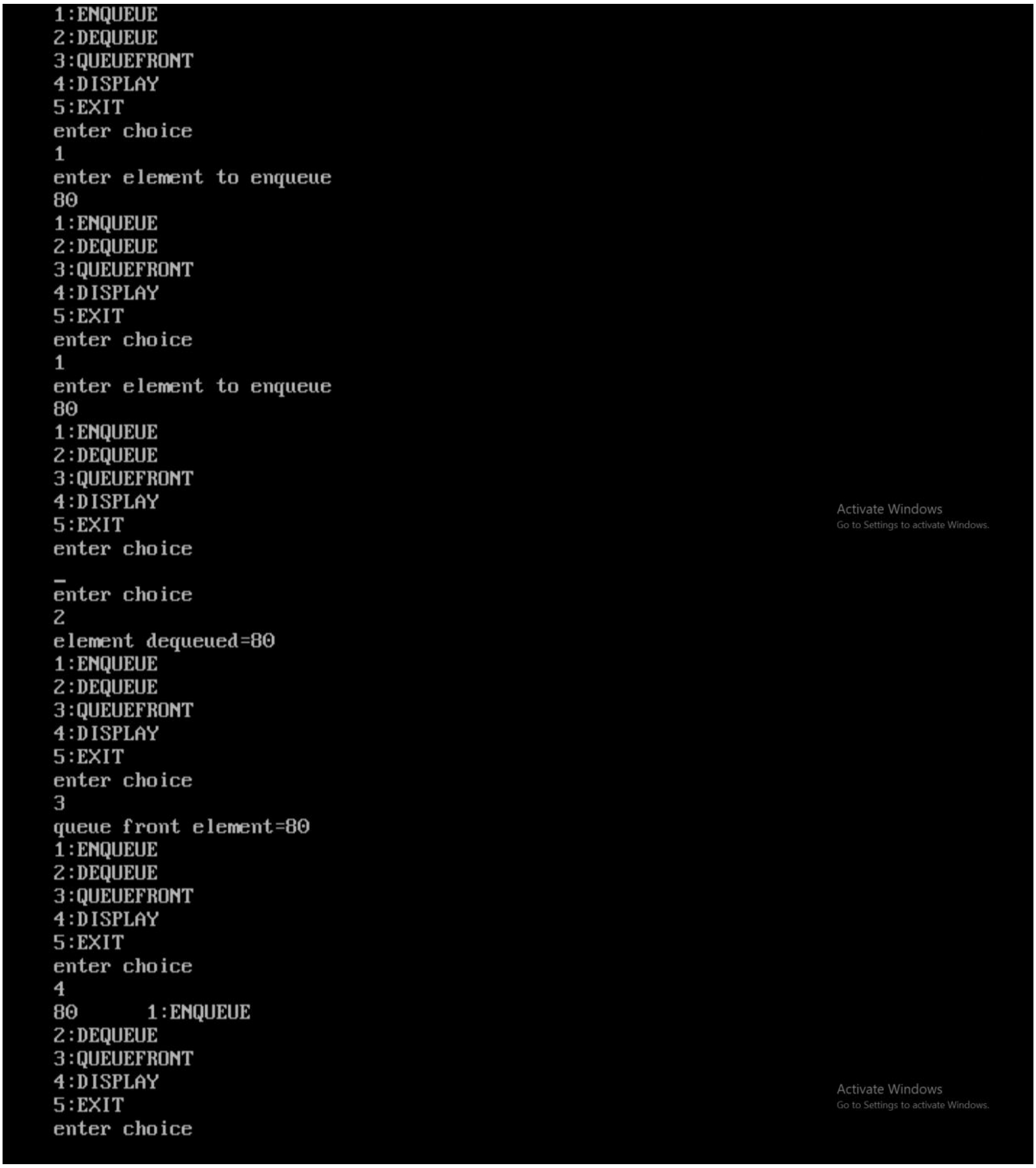
```
#include<conio.h>

#include<stdio.h>

#define MAX 10

typedef struct

{

        int a[MAX];

        int front ,rear;

        int count;

}cqueue;

void enqueue(cqueue *q,int ele)

{
        if(q->count==MAX)
        {
                printf("circular queue overflow\n");
                return;
        }
        q->count++;
        q->rear=(q->rear+1)%MAX;
        q->a[q->rear]=ele;
}
int isempty(cqueue *q)
{
        if(q->count==0)
        return 1;
        else
        return 0;
}
int dequeue(cqueue *q)
{
        int x;
        if(isempty(q))
        return -1;
        q->count--;
        x=q->a[q->front];
        q->front=(q->front+1)%MAX;
        return x;
}
```

```
int queuefront(cqueue *q)

{

        if(isempty(q))

        return -1;

        else

        return q->a[q->front];


}

void display(cqueue *q)

{

        int i;

        if(isempty(q))

        {

                printf("queue underflow\n");

                return;

        }

        i=q->front;

        while(1)

        {

                printf("%d\t",q->a[i]);

                if(i==q->rear)

                break;

                i=(i+1)%MAX;

        }

}

int main()
```

```
{
        int ch,ele;
        cqueue q;
        q.front=0;
        q.rear=MAX-1;
        q.count=0;
        while(1)
        {
printf("1:ENQUEUE\n2:DEQUEUE\n3:QUEUEFRONT\n4:DISPLAY\n5:EXIT\n");
                printf("enter choice\n");
                scanf("%d",&ch);
                if(ch==5)
                break;
                switch(ch)
                {
                        case 1:printf("enter element to enqueue\n");
                                        scanf("%d",&ele);
                                        enqueue(&q,ele);
                                        break;
                        case 2:ele=dequeue(&q);
                                        if(ele==-1)
                                        printf("queue underflow\n");
                                        else
                                        printf("element dequeued=%d\n",ele);
                                        break;
```

```
            case 3:ele=queuefront(&q);

                    if(ele==-1)

                    printf("queue underflow\n");

                    else

                    printf("queue front element=%d\n",ele);

                    break;

            case 4:display(&q);

                    break;

            default:printf("invalid choice\n");

        }

    }

  getch();

    return 0;


}
```

OUTPUT:



```
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
1
enter element to enqueue
80
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
1
enter element to enqueue
80
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
_
enter choice
2
element dequeued=80
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
3
queue front element=80
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
4
80      1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
```

QUE 10)

DESCENDING PRIORITY QUEUE WITH NORMAL DELETE

Code:

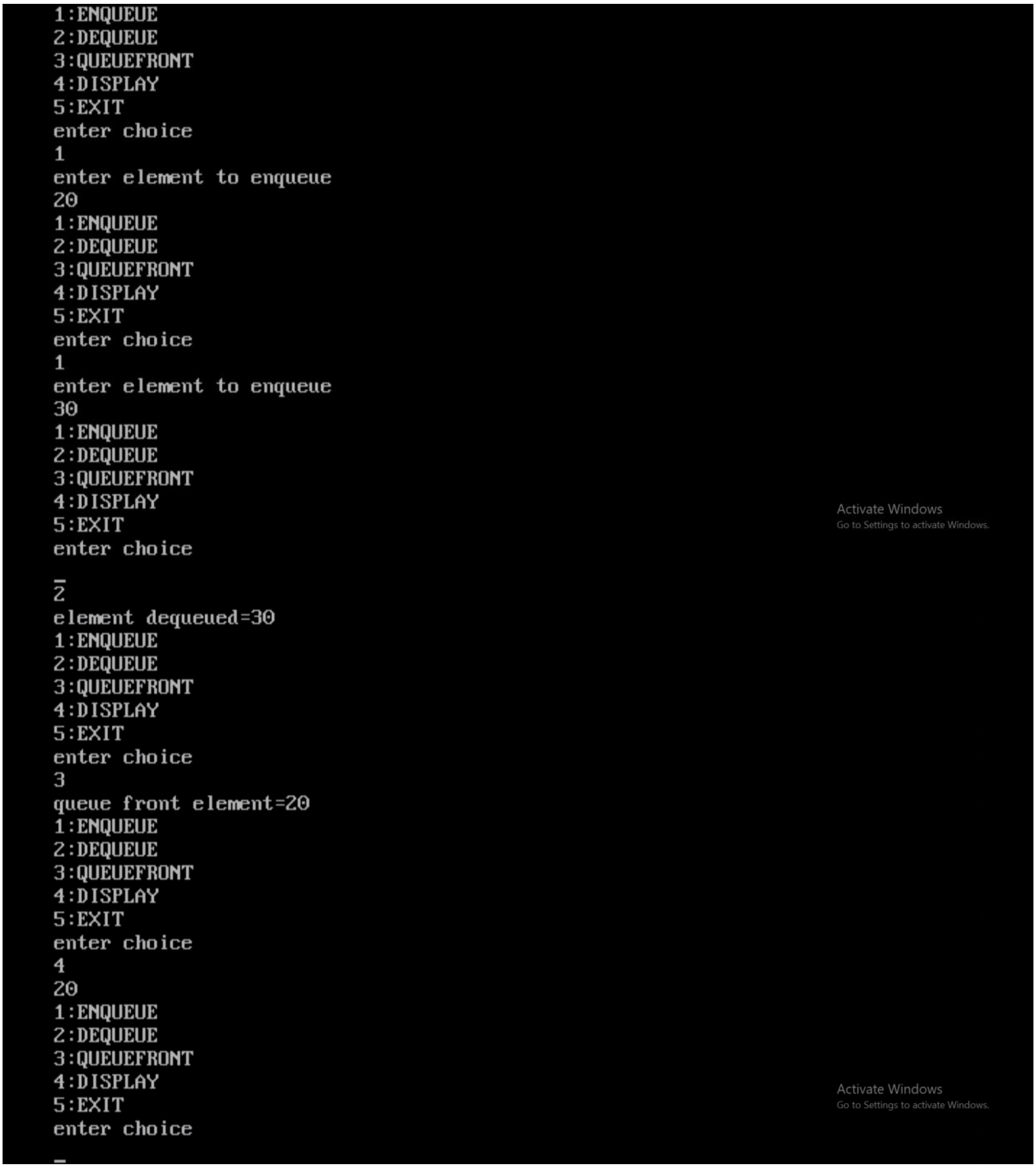
```
#include<conio.h>

#include<stdio.h>
```

```
#define MAX 5
typedef struct
{
	int a[MAX];
	int front,rear;
}queue;
void enqueue(queue *q,int ele)
{
	int i,j;
	if(q->rear==MAX-1)
	{
		printf("queue overflow\n");
		return;
	}
	for(i=q->front;i<=q->rear;i++)
		if(ele>q->a[i])
		break;
	for(j=q->rear;j>=i;j--)
		q->a[i+1]=q->a[j];
		q->a[i]=ele;
	q->rear++;
}
int isempty(queue *q)
{
	if(q->rear+1==q->front)
	return 1;
```

```
    else
    return 0;
}
int dequeue(queue *q)
{
    int x;
    if(isempty(q))
    return -1;
    x=q->a[q->front];
    q->front++;
    return x;
}
int queuefront(queue *q)
{
    if(isempty(q))
    return -1;
    else
    return q->a[q->front];
}
void display(queue *q)
{
    int i;
    if(isempty(q))
    {
        printf("queue underflow\n");
        return;
    }
    else
    {
        for(i=q->front;i<=q->rear;i++)
        printf("%d\t",q->a[i]);
        printf("\n");
    }
}
int main()
{
    int ch,ele;
    queue q;
    q.front=0;
    q.rear=-1;
    while(1)
    {

printf("1:ENQUEUE\n2:DEQUEUE\n3:QUEUEFRONT\n4:DISPLAY\n5:EXIT\n");
        printf("enter choice\n");
        scanf("%d",&ch);
        if(ch==5)
        break;
        switch(ch)
        {
            case 1:printf("enter element to enqueue\n");
                    scanf("%d",&ele);
```

```
                                                            enqueue(&q,ele);

                                                            break;

                                        case 2:ele=dequeue(&q);

                                                            if(ele==-1)

                                                      printf("queue underflow\n");

                                                            else

                                                            printf("element dequeued=%d\n",ele);

                                                            break;

                                        case 3:ele=queuefront(&q);

                                                                      if(ele==-1)

                                                            printf("queue underflow\n");

                                                                      else

                                                                      printf("queue front
element=%d\n",ele);

                                                                      break;

                                        case 4:display(&q);

                                                            break;

                                        default :printf("invalid choice\n");

                              }

                    }

                getch();

                    return 0;

          }
```

OUTPUT:



```
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
1
enter element to enqueue
20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
1
enter element to enqueue
30
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice

2
element dequeued=30
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
3
queue front element=20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
4
20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
_
```

QUE 11)

DESCENDING PRIORITY QUEUE WITH NORMAL INSERT

Code:

#include<conio.h>

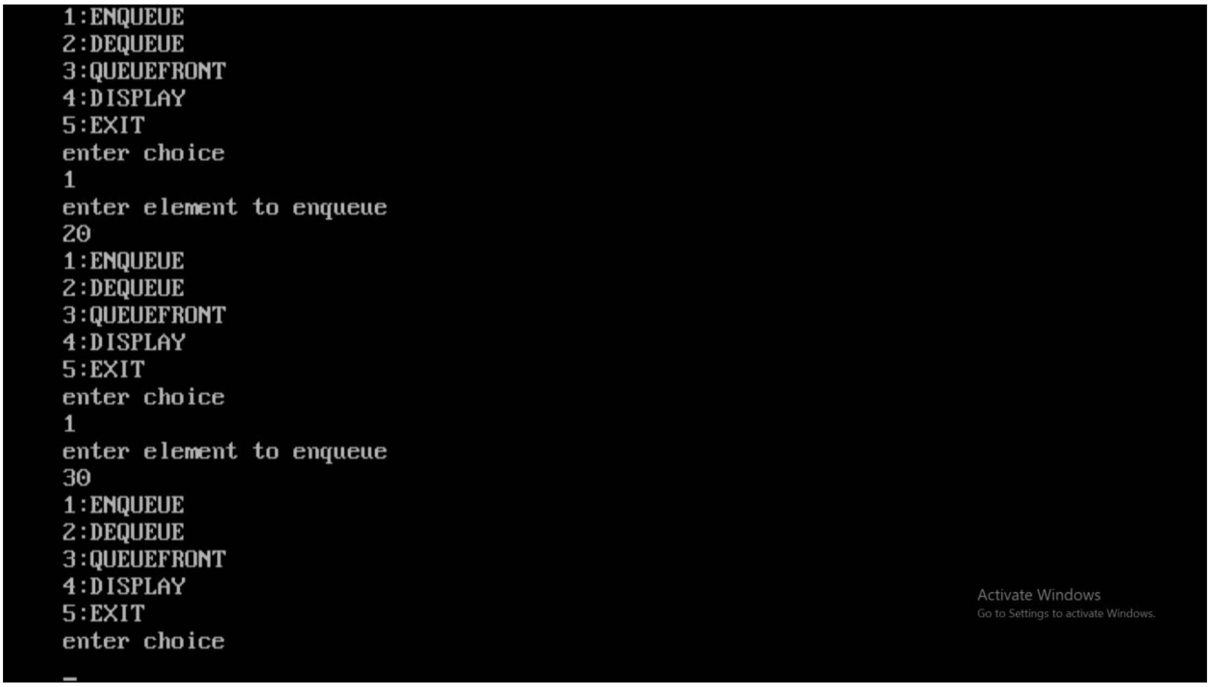
#include<stdio.h>

```
#define MAX 5

typedef struct

{

        int a[MAX];

        int front,rear;

}queue;

void enqueue(queue *q,int ele)

{

        if(q->rear==MAX-1)

        {

                printf("queue overflow\n");

                return;

        }

        q->rear++;

        q->a[q->rear]=ele;

}

int isempty(queue *q)

{

        if(q->rear+1==q->front)

        return 1;

        else

        return 0;

}

int dequeue(queue *q)

{

        int max, pos, i;
        if(isempty(q))

        return -1;

        max=q->a[q->front];

        pos=q->front;

        for(i=q->front+1;i<=q->rear;i++)

                {

                        if(q->a[i]>max)

                        {

                                max=q->a[i];

                                pos=i;

                        }

                }

                for(i=pos;i<q->rear;i++)

                        q->a[i]=q->a[i+1];

                        q->rear--;

        return max;

}

int queuefront(queue *q)

{

        if(isempty(q))

        return -1;

        else

        return q->a[q->front];

}

void display(queue *q)

{
	int i;
	if(isempty(q))
	{
		printf("queue underflow\n");
		return;
	}
	else
	{
		for(i=q->front;i<=q->rear;i++)
		printf("%d\t",q->a[i]);
		printf("\n");
	}
}
int main()
{
	int ch,ele;
	queue q;
	q.front=0;
	q.rear=-1;
	while(1)
	{
printf("1:ENQUEUE\n2:DEQUEUE\n3:QUEUEFRONT\n4:DISPLAY\n5:EXIT\n");
		printf("enter choice\n");
		scanf("%d",&ch);
		if(ch==5)
```

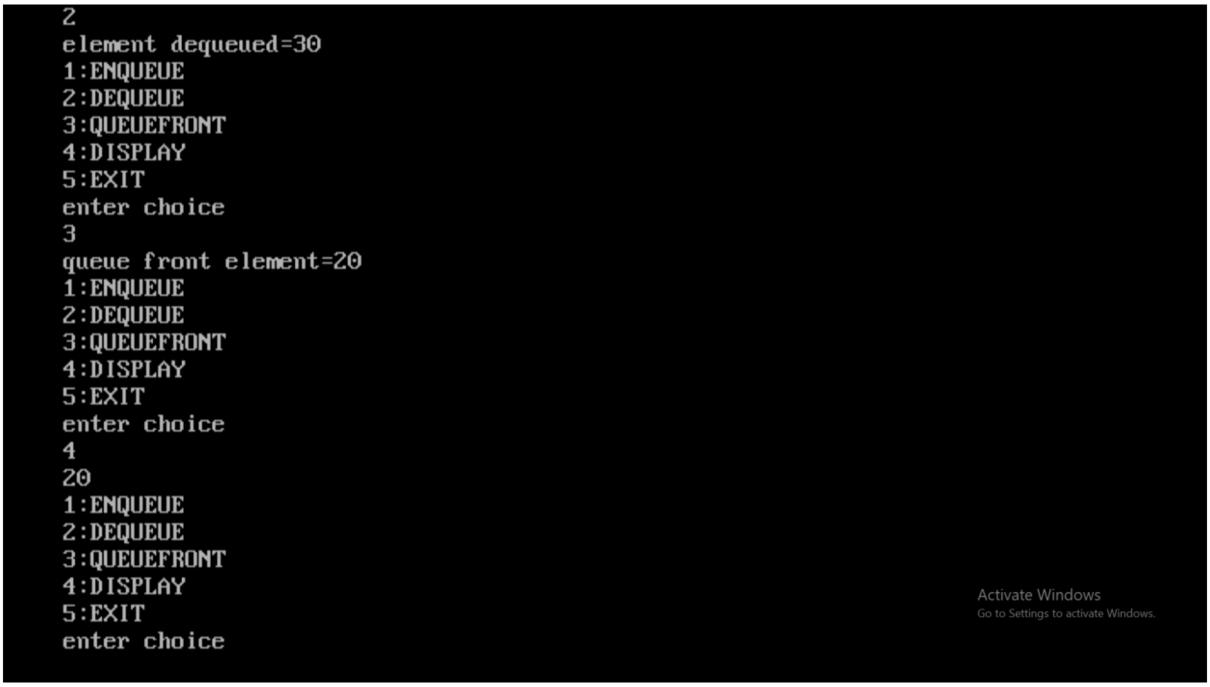
```
            break;
            switch(ch)
            {
                case 1:printf("enter element to enqueue\n");
                        scanf("%d",&ele);
                        enqueue(&q,ele);
                        break;
                case 2:ele=dequeue(&q);
                        if(ele==-1)
                      printf("queue underflow\n");
                        else
                        printf("element dequeued=%d\n",ele);
                        break;
                case 3:ele=queuefront(&q);
                            if(ele==-1)
                        printf("queue underflow\n");
                            else
                            printf("queue front
element=%d\n",ele);
                            break;
                case 4:display(&q);
                        break;
                default :printf("invalid choice\n");
            }
        }
        return 0;
```

}

OUTPUT:



```
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
1
enter element to enqueue
20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
1
enter element to enqueue
30
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
_
```



```
2
element dequeued=30
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
3
queue front element=20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
4
20
1:ENQUEUE
2:DEQUEUE
3:QUEUEFRONT
4:DISPLAY
5:EXIT
enter choice
```

QUE 12)

IMPLEMENT DOUBLE ENDED QUEUE

Code:

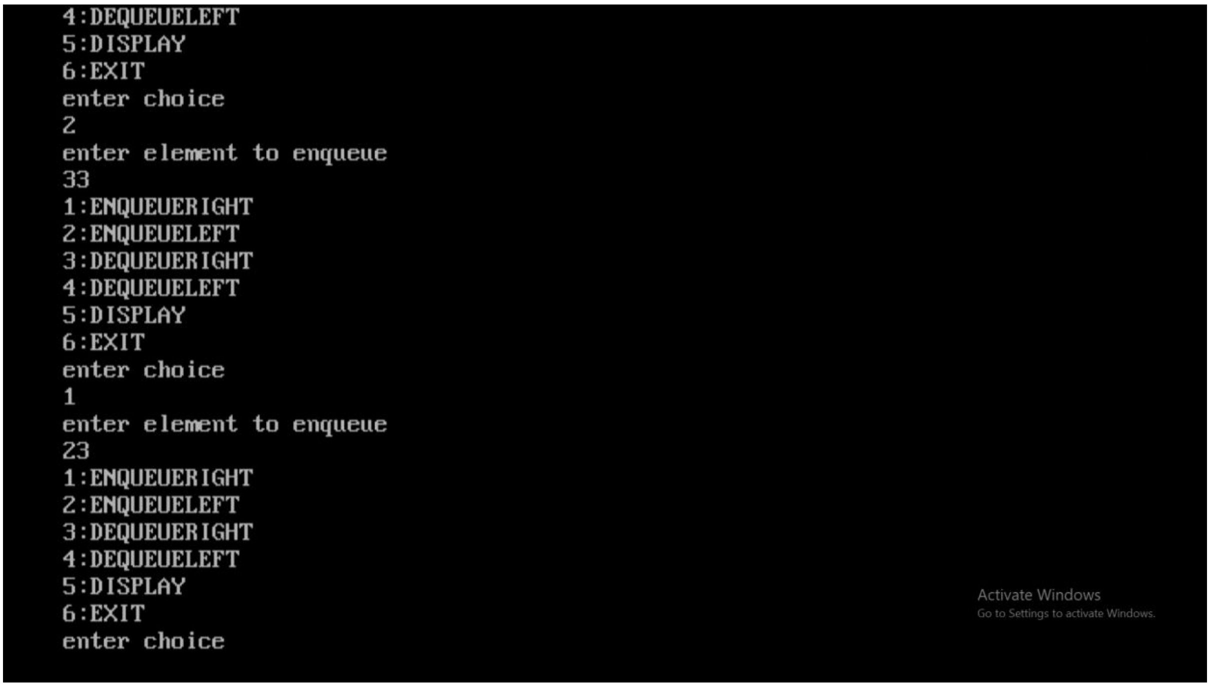
```
#include<conio.h>

#include<stdio.h>

#define MAX 10

typedef struct

{

        int a[MAX];

        int front ,rear;

}queue;

void enqueueright(queue *q,int ele)

{

        if(q->rear==MAX-1)

        {

                printf("queue overflow\n");

                return;

        }

        q->rear++;

        q->a[q->rear]=ele;

}

void enqueueleft(queue *q,int ele)

{

        if(q->front==-1)

        {

                printf("queue overflow\n");

                return;

        }

        q->a[q->front]=ele;
        q->front--;
}
int isempty(queue *q)
{
        if(q->rear==q->front)
        return 1;
        else
        return 0;
}
int dequeueright(queue *q)
{
        int x;
        if(isempty(q))
        return -1;
        x=q->a[q->rear];
        q->rear--;
        return x;
}
int dequeueleft(queue *q)
{
        int x;
        if(isempty(q))
        return -1;
        q->front++;
        x=q->a[q->front];
        return x;
```

```
}
void display(queue *q)
{
        int i;
        if(isempty(q))
        {
                printf("queue underflow\n");
                return;
        }
        else
        {
                for(i=q->front+1;i<=q->rear;i++)
                printf("%d\t",q->a[i]);
                printf("\n");
        }
}
int main()
{
        int ch,ele;
        queue q;
        q.front=MAX/2;
        q.rear=MAX/2;
        while(1)
        {

printf("1:ENQUEUERIGHT\n2:ENQUEUELEFT\n3:DEQUEUERIGHT\n4:DEQUEUELEFT
\n5:DISPLAY\n6:EXIT\n");
```

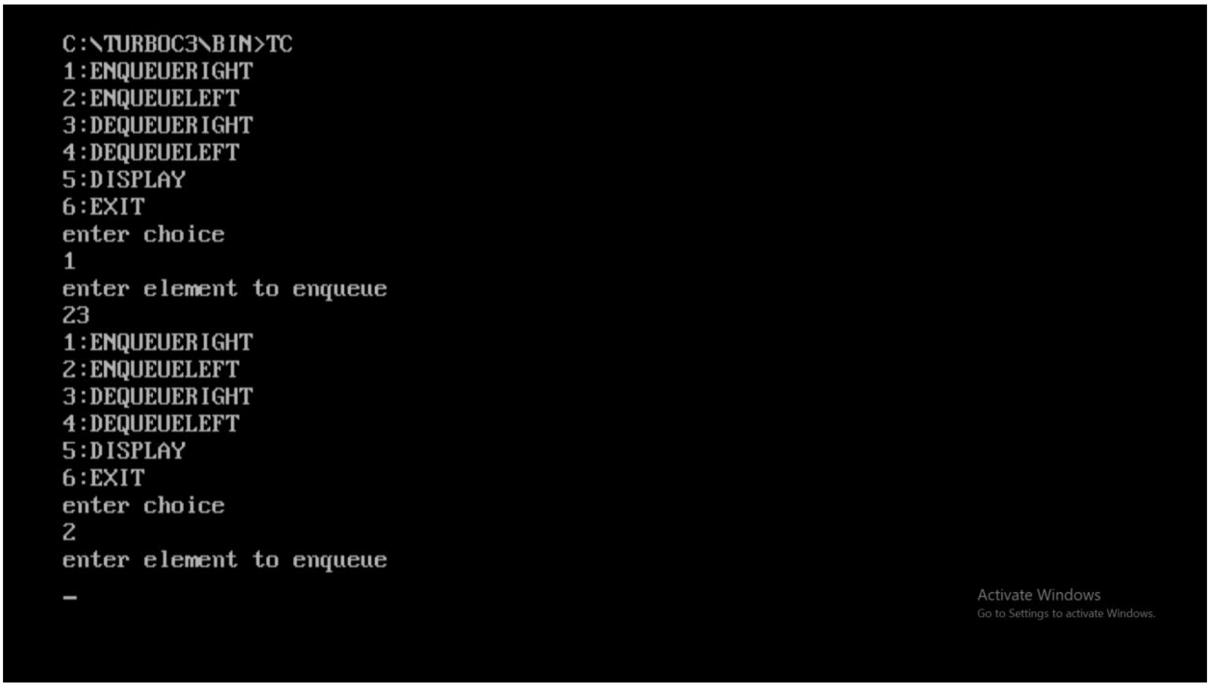
```
printf("enter choice\n");
scanf("%d",&ch);
if(ch==6)
break;
switch(ch)
{
        case 1:printf("enter element to enqueue\n");
                        scanf("%d",&ele);
                        enqueueright(&q,ele);
                        break;
        case 2:printf("enter element to enqueue\n");
                        scanf("%d",&ele);
                        enqueueleft(&q,ele);
                        break;
  case 3:ele=dequeueright(&q);
                if(ele==-1)
                printf("queue underflow\n");
                else
                printf("element dequeued=%d\n",ele);
                break;
  case 4:ele=dequeueleft(&q);
                if(ele==-1)
                printf("queue underflow\n");
                else
                printf("element dequeued=%d\n",ele);
                break;
```

```
                    case 5:display(&q);
                                break;
                    default :printf("invalid choice\n");
                  }
           }
     getch();
        return 0;
}
```

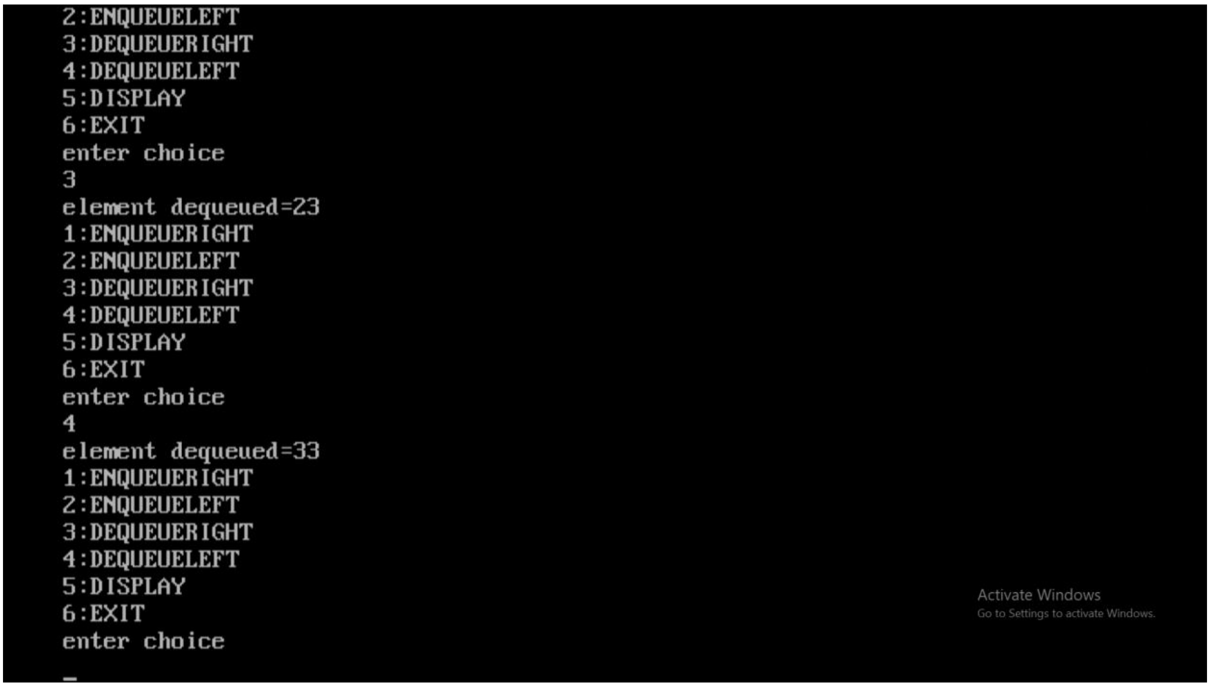
OUTPUT:



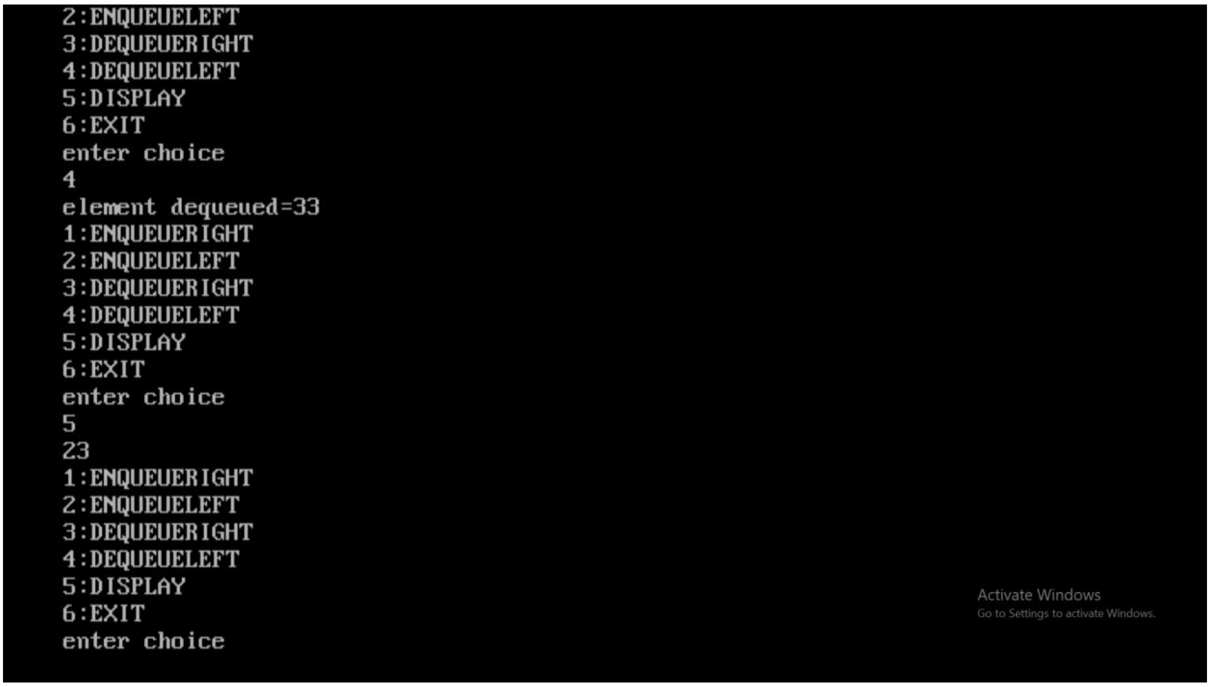
```
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
2
enter element to enqueue
33
1:ENQUEUERIGHT
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
1
enter element to enqueue
23
1:ENQUEUERIGHT
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
```



```
C:\TURBOC3\BIN>TC
1:ENQUEUERIGHT
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
1
enter element to enqueue
23
1:ENQUEUERIGHT
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
2
enter element to enqueue
_
```



```
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
3
element dequeued=23
1:ENQUEUERIGHT
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
4
element dequeued=33
1:ENQUEUERIGHT
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
_
```



```
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
4
element dequeued=33
1:ENQUEUERIGHT
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
5
23
1:ENQUEUERIGHT
2:ENQUEUELEFT
3:DEQUEUERIGHT
4:DEQUEUELEFT
5:DISPLAY
6:EXIT
enter choice
```