

Double ended queue

```
#include <stdio.h>
```

```
#define qsize 5
```

```
int b=0, r=-1, ch;
```

```
int item, r[10];
```

```
int isfull()
```

```
{ return (r==rsize-1)? 1:0;
```

```
}
```

```
int isempty()
```

```
{ return (b>r)? 1:0;
```

```
}
```

```
void insert - queue()
```

```
{
```

```
if(isfull())
```

```
{
```

```
printf("queue overflow\n");
```

```
return;
```

```
}
```

```
r=r+1;
```

```
q[r]=item;
```

```
}
```

```
void delete - front()
```

```
{
```

```
if(isempty())
```

```
{
```

```
printf("queue empty\n");
```

```
return;
```

```
}
```

```
printf("Item deleted is %d\n", q[b]);
```

```
if(b>r)
```

```
{
```

```
b=0;
```

```
r=-1;
```

```
}
```

```
void insert_front()
```

```
{
```

```
if (f == 0)
```

```
{
```

```
    f = f - 1;
```

```
    q[f] = item;
```

```
    return;
```

```
}
```

```
else if ((f == 0) & (r == -1))
```

```
{
```

```
    q[++r] = item;
```

```
    return;
```

```
}
```

```
else
```

```
    printf("Insertion not possible \n");
```

```
}
```

```
void delete_rear()
```

```
{
```

```
if (isEmpty())
```

```
{
```

```
    printf("queue is empty \n");
```

```
    return;
```

```
}
```

```
printf("item deleted is %d \n", q[r--]);
```

```
if (f > r)
```

```
{
```

```
    f = 0;
```

```
    r = -1;
```

```
}
```

```
}
```

```
void display()
```

```
{
```

```
int i;
```

```
if (isEmpty())
```

```
{
```

```
    printf("queue empty \n");
```

```
    return;
```

```
}
```

classmate

```
for(i=k; i<=n; i++)  
    printf("%d\n", r[i]);
```

```
}  
void main()
```

```
{  
    for(j;j
```

```
    printf("\nxxxxxxx\n");
```

```
    printf("1. insert rear\n2. insert-front\n3. delete rear\n4. delete front\n5. display\n6. exit\n");
```

```
    printf("Enter choice: ");
```

```
    switch(ch)
```

```
{
```

```
    case 1: printf("Enter the choice item\n");
```

```
            scanf("%d", &item);
```

```
            insert_rear();
```

```
            break;
```

```
    case 2: printf("Enter the item: ");
```

```
            scanf("%d", &item);
```

```
            insert_front();
```

```
            break;
```

```
    case 3: delete_rear();
```

```
            break;
```

```
    case 4: delete_front();
```

```
            break;
```

```
    case 5: display();
```

```
            break;
```

```
    default: exit(0);
```

```
}
```

```
    printf("\nxxxxxxx\n");
```

```
}
```

```
}
```

```
*****
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice: 1
enter the item
10
```

```
*****
```

```
*****
```

```
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice: 1
enter the item
20
```

```
*****
```

```
*****
```

```
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice: 1
enter the item
30
```

```
*****
```



```
*****
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice: 1
enter the item
40
queue overflow
*****

*****

1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice: 3
item deleted is 30
*****

*****

1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice: 4
item deleted is 10
*****

*****

1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice: 5
20
*****
```

Input restricted DQ:-

```
#include <stdio.h>
#define qsize 5
int f=0, r=-1, ch;
int item, q[10];
int isfull()
{
    return (r==qsize-1)?1:0;
}

int isempty()
{
    return (f>r)?1:0;
}

int insert_rear()
{
    if (isfull())
    {
        printf("queue overflow\n");
        return;
    }
    r = r+1;
    q[r] = item;
}

void delete_front()
{
    if (isempty())
    {
        printf("queue empty\n");
        return;
    }
    printf("item deleted is %d\n", q[f++]);
    if (f>r)
    {
        f=0;
        r=-1;
    }
}
```

```
void deleteQueue()
```

```
{
```

```
if (isEmpty())
```

```
{
```

```
printf("queue is empty \n");
```

```
return;
```

```
}
```

```
printf("item deleted is %d \n", q[n-1]);
```

```
if (t > 1)
```

```
{
```

```
t = 0;
```

```
n = -1;
```

```
}
```

```
}
```

```
void display()
```

```
{
```

```
int i;
```

```
if (isEmpty())
```

```
{
```

```
printf("queue empty \n");
```

```
return;
```

```
}
```

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

```
printf("%d \n", q[i]);
```

```
}
```

```
void main()
```

```
{
```

```
for (j)
```

```
{
```

```
printf("\n x x x x x x x x x x \n");
```

```
printf("1. insert - rear \n 2. delete - rear \n 3. delete - front \n 4. Display \n  
5. exit \n");
```

```
printf("Enter choice:");
```

```
scanf("%d", &ch);
```

```
switch (ch)
```

```
{
```

```
Page
```

```
Date
```


case 1: printf("Enter the item: ");

scanf("%d", &item);

insert, root();

break;

case 2: delete - ~~for~~ root();

break;

case 3: delete - print();

break;

case 4: display();

break;

default : exit(0);

}

printf("\n x x x x x x x x x x \n");

}

}

Output restricted:-


```
*****
1.insert_rear
2.delete_rear
3.delete_front
4.display
5.exit
enter choice:
1
Enter the item: 10

*****

*****
1.insert_rear
2.delete_rear
3.delete_front
4.display
5.exit
enter choice: 1
Enter the item: 20

*****

*****
1.insert_rear
2.delete_rear
3.delete_front
4.display
5.exit
enter choice: 1
Enter the item: 30

*****

*****
1.insert_rear
2.delete_rear
3.delete_front
4.display
5.exit
enter choice: 4
10
20
30
```

1.insert_rear
2.delete_rear
3.delete_front
4.display
5.exit
enter choice: 2
item deleted is 30

1.insert_rear
2.delete_rear
3.delete_front
4.display
5.exit
enter choice: 3
item deleted is 10

1.insert_rear
2.delete_rear
3.delete_front
4.display
5.exit
enter choice: 4
20

Output Restricted:-

```
#include <stdio.h>
#define qsize 5
int f=0, r=-1, ch;
int item, a[10];
int isfull()
{
    return (r==qsize-1)? 1:0;
}

int isempty()
{
    return (f>r)? 1:0;
}

void insert-rear()
{
    if (isfull())
    {
        printf("queue over flow\n");
        return;
    }
    r=r+1;
    a[r]=item;
}
```

classmate

Date _____

Page _____


```
void delete_front ()
```

```
{
```

```
if (isEmpty())
```

```
{
```

```
printf("queue empty\n");
```

```
return;
```

```
}
```

```
printf("item deleted is %d\n", q[b++]);
```

```
if (b > r)
```

```
{
```

```
b = 0;
```

```
r = -1;
```

```
}
```

```
}
```

```
void insert_front ()
```

```
{
```

```
if (b != 0)
```

```
{
```

```
b = b - 1;
```

```
q[b] = item;
```

```
return;
```

```
}
```

```
else if ((b == 0) && (r == -1))
```

```
{
```

```
q[++r] = item;
```

```
return;
```

```
}
```

```
else
```

```
printf("Insertion not possible\n");
```

```
}
```

```
void display ()
```

```
{
```

```
int i;
```

```
if (isEmpty())
```

```
{
```

```
printf("queue empty\n");
```

```
return;
```

```
}
```

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

```
printf("%d\n", q[i]);
```

```
}
```

{

1

4. display ln 5. exit ln "];

scanf ("%d", &i);

3

scanf ("%d", &item);

break;

```
scanf("%d", &iItem);
```

break:

break;

break;

3

3

13

1.insert_rear
2.insert_front
3.delete_front
4.display
5.exit
enter choice: 1
Enter the item: 10

1.insert_rear
2.insert_front
3.delete_front
4.display
5.exit
enter choice: 1
Enter the item: 20

1.insert_rear
2.insert_front
3.delete_front
4.display
5.exit
enter choice: 1
Enter the item: 30

1.insert_rear
2.insert_front
3.delete_front
4.display
5.exit
enter choice: 4
10
20
30

1.insert_rear
2.insert_front
3.delete_front
4.display
5.exit
enter choice: 3
item deleted is 10

1.insert_rear
2.insert_front
3.delete_front
4.display
5.exit
enter choice: 3
item deleted is 20

1.insert_rear
2.insert_front
3.delete_front
4.display
5.exit
enter choice: 4
30

1.insert_rear
2.insert_front
3.delete_front
4.display
5.exit
enter choice: 3
item deleted is 30
