dequeue.c | mulpri.c | despri2.c

dequeue.c > main()

```c
#include<stdio.h>
#include<stdlib.h>
#include<conio.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;
  }
void 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;
```

dequeue.c | mulpri.c | despri2.c

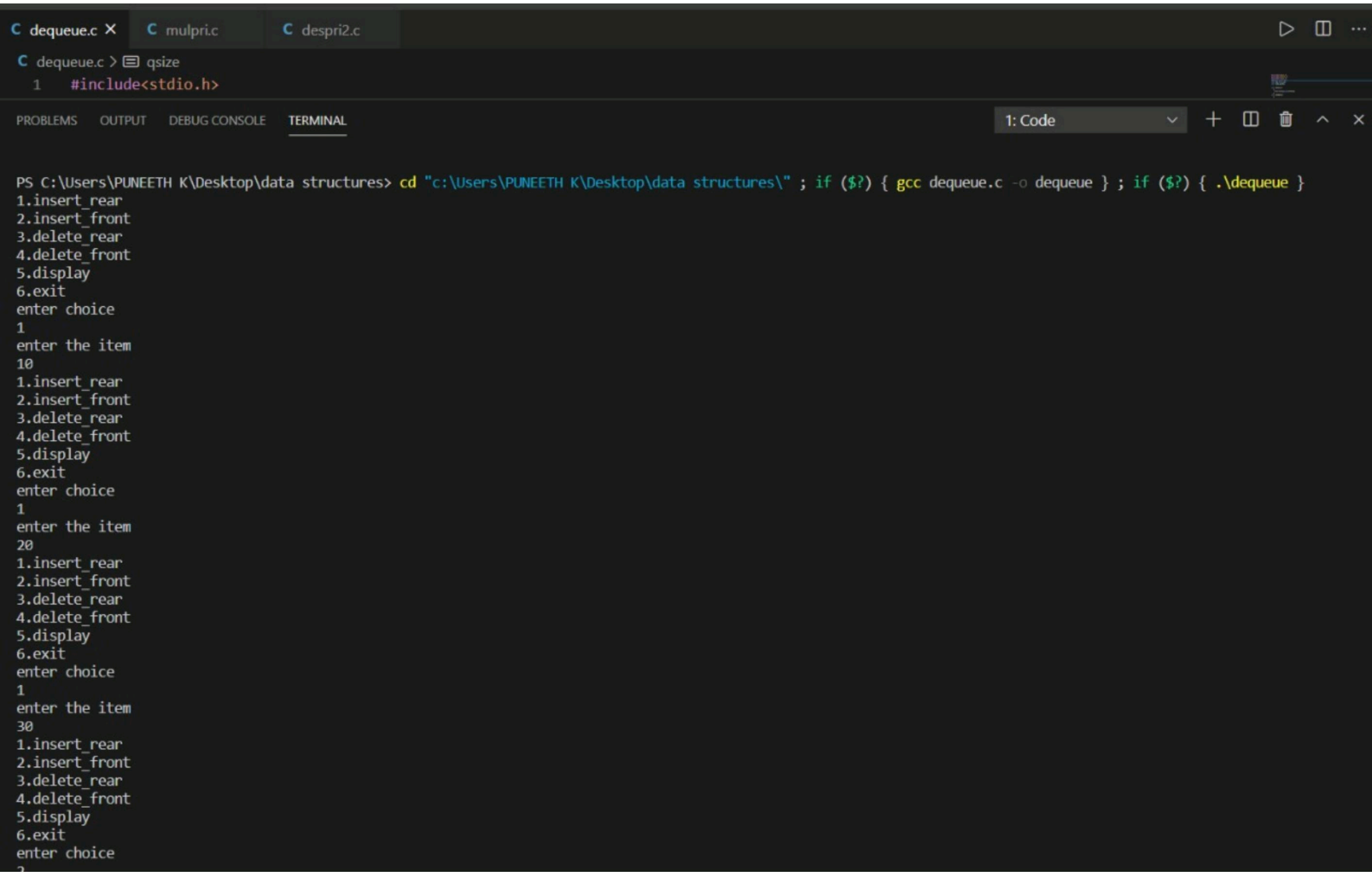
dequeue.c > main()

```
33      printf("item deleted is %d\n",q[(f)++]);
34      if(f>r)
35        {
36         f=0;
37         r=-1;
38        }
39     }
40   void insert_front()
41     {
42      if(f!=0)
43        {
44         f=f-1;
45         q[f]=item;
46         return;
47        }
48       else if((f==0)&&(r==-1))
49        {
50         q[++(r)]=item;
51         return;
52        }
53       else
54         printf("insertion not possible\n");
55     }
56   void delete_rear()
57     {
58      if(isempty())
59        {
60         printf("queue is empty\n");
61         return;
62        }
63      printf("item deleted is %d\n",q[(r)--]);
64      if(f>r)
65        {
66         f=0;
67         r=-1;
68        }
69     }
70   void display()
```

dequeue.c | mulpri.c | despri2.c

dequeue.c > main()

```
70  void display()
71    {
72     int i;
73     if(isempty())
74       {
75        printf("queue empty\n");
76        return;
77       }
78     for(i=f;i<=r;i++)
79      printf("%d\n",q[i]);
80    }
81  void main()
82   {
83
84
85    for(;;)
86     {
87      printf("1.insert_rear\n2.insert_front\n3.delete_rear\n4.delete_front\n5.display\n6.exit\n");
88      printf("enter choice\n");
89      scanf("%d",&ch);
90      switch(ch)
91        {
92         case 1:printf("enter the item\n");
93                scanf("%d",&item);
94                insert_rear();
95                break;
96         case 2:printf("enter the item\n");
97                scanf("%d",&item);
98                insert_front();
99                break;
100        case 3:delete_rear();
101               break;
102        case 4:delete_front();
103               break;
104        case 5:display();
105               break;
106        default:exit(0);
```


dequeue.c
mulpri.c
despri2.c
dequeue.c > qsize
1 #include<stdio.h>
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: Code
PS C:\Users\PUNEETH K\Desktop\data structures> cd "c:\Users\PUNEETH K\Desktop\data structures\" ; if ($?) { gcc dequeue.c -o dequeue } ; if ($?) { .\dequeue }
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

dequeue.c | mulpri.c | despri2.c

dequeue.c > qsize

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

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Code

```
2
enter the item
10
insertion not possible
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
1
enter the item
45
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
5
10
20
45
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
```

DEQUEUES

```
#include <stdio.h>
#include <conio.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;
}
void 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 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;
 }
```

```
for (i=f ; i<=r ; i++)
printf ("%d \n", q[i]);
}
void main ()
{
for (;;)
{
printf ("1. insert_rear \n 2. insert_front \n 3. delete_rear \n 4. delete_
front \n 5. display \n 6. exit \n");
printf ("enter choice \n");
scanf ("%d", &ch);
switch(ch)
{
case 1: printf("enter the item \n");
scanf ("%d", &item);
insert_rear ();
break;
case 2: printf ("enter the item \n");
scanf ("%d", &item);
insert_front();
break;
case 3: delete_rear ();
break;
case 4: delete_front ();
break;
case 5: display ();
break;
default: exit (0);
}
}
}
```