DATE: PAGE:

POOJA.K
IBMI9CSIII

// Dequeue program

```
#include<stdio.h>
#include<stdlib.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 is 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");
}
```

DATE: PAGE:

```
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]);
 }

int 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");
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);
    }
  }
 return 0;
}
```

```c
#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;
  }
void insert_rear()
  {
   if(isfull())
   {
    printf("queue overflow\n");
    return;
   }
   r=r+1;
   q[r]=item;
  }
void delete_front()
  {
```

```
25    {
26     if(isempty())
27     {
28      printf("queue empty\n");
29      return;
30     }
31     printf("item deleted is %d\n",q[(f)++]);
32     if(f>r)
33     {
34      f=0;
35      r=-1;
36     }
37    }
38   void insert_front()
39    {
40     if(f!=0)
41     {
42      f=f-1;
43      q[f]=item;
44      return;
45     }
46    else if((f==0)&&(r==-1))
47     {
48      q[++(r)]=item;
49      return;
```

```
ain.c
        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");
```

main.c

```
73      printf("queue empty\n");
74      return;
75     }
76     for(i=f;i<=r;i++)
77    printf("%d\n",q[i]);
78    }
79  int main()
80   {
81
82    for(;;)
83    {
84    printf
      ("1.insert_rear\n2.insert_front\n3.delete_rear\n4.
      delete_front\n5.display\n6.exit\n");
85    printf("enter choice\n");
86    scanf("%d",&ch);
87    switch(ch)
88      {
89       case 1:printf("enter the item\n");
90          scanf("%d",&item);
91          insert_rear();
92          break;
93       case 2:printf("enter the item\n");
94          scanf("%d",&item);
           insert_front();
```

Maps

enGranularCl...

Stop

main.c

```
92          break;
93        case 2:printf("enter the item\n");
94          scanf("%d",&item);
95          insert_front();
96          break;
97        case 3:delete_rear();
98          break;
99        case 4:delete_front();
100         break;
101       case 5:display();
102         break;
103
104     }
105   }
106    return 0;
107   }
```

```
$ ./main
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
```

```
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
4
item deleted is 10
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
4
item deleted is 20
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
2
```

```
6.exit
enter choice
2
enter the item
40
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
2
enter the item
50
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
5
50
40
30
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
```

```
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
3
item deleted is 40
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
3
item deleted is 50
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
```

```
5.display
6.exit
enter choice
3
item deleted is 40
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
3
item deleted is 50
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
3
queue is empty
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
```