deque

```
#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(isful())
      {
        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");
```

```
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. deleterear
       \n 4. deletefront \n 5. display \n 6. exit \n");
printf ("enter the choic \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);
}
}
```

Multiple queue

```
#include <stdio.h>
#define N 3
int queue[3][N];
int front[3] = {0, 0, 0};
int rear[3] = {-1, -1, -1};
int item, pr;
void main()
{
int ch;
while (1)
{
printf("Priority Queue \n");
printf("*********\n");
printf("1: PQinsert \n");
printf("2: PQdelete \n");
printf("3: PQ display \n");
scanf("%d", &ch);
switch (ch)
{
case 1: printf("\nenterthe priority number \n");
        scanf("%d", &pr);
        if (pr>0 && pr<4)
        PQinsert(pr-1);
        else
        printf("only 3 priorityexists 1.2 3\n");
```

```
case 2 : pqdelete();
         break;
case 3 : display();
         break;
case 4 : exit(0);
}
}
}
pqinsert(int pr)
{
if (rear[pr] = N-1)
printf("\n Queue overflow \n");
else
{
printf("\nenter the item \n");
scanf("%d", &item);
rear[pr]++;
queue[pr][rear[pr]] = item;
}
return;
}
pqdelete()
{
int i;
for(i=0;i<3;i++)
{
if(rear[i]==front[i]-1)
```

```
        printf ("\n queue is empty \n");
        else
        {
        printf ("deleted item is %d of queue %d \n",
                queue [i][front [i]], i+1);
        front [i]++;
        return;
        }
    }
}
display()
{
int i, j;
for (i=0; i<3; i++)
{
if (rear [i] == front [i]-1)
    printf ("queue is empty \n", i+1);
else
    {
    printf ("\n Queue %d :", i+1);
    for (j=front [i]; j<=rear [i]; j++)
    printf ("%d \t", queue [i][j]);
    }
}
return;
}
```