LAB-4;

: Double Ended Queue :

```
#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)? 1:00; }

void inset_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 insert_front()
 {  if (f!=0)
    { f= f-1;  q [f]=item; return;}
    else if ((f==0) && (r==-1))
          q[++(r)]=item; return;
    else
       { 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 main()
{
  for(;;)
  { printf (" 1. inseit_rear \n
              2. inseit_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);
       inseit_rear();
        break;

 case 2: printf ("enter the item \n");
       Scanf ("%d", &item);
       in seit_front();
       break;
 case 3: delete_rear();
        break;
 case 4; delete_front();
        break;
 case 5; display();
        break;
```

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

Practice : Circular Queue;

```
# include <stdio.h>
# include <conio.h>
# define QUE_size 3

int item, front=0, rear = -1, q[Que_size],
    count=0;

void insertrear()
{
  if (count == QUE_SIZE)
  {
    pf ("Queue Overflow \n");
    return;
  }
  rear = (rear+1) %. QUE_SIZE;
   q [rear] = item;
    count ++;
}

int deletefront()
{
  if (count == 0) return -1;
   item = q [front];
   front = (front +1) %. QUE_SIZE;
    count = count-1;
    return item;
}
```

```
void displayQ()
{  int i,f;
   if (count==0)
   {
     pf ("queue is empty \n");
      return;
   }
  f = front;
   printf ("Contents of Queue \n");
    for (i=1; i<=count ; i++)
   {
     printf ("%d\n", q[f];
      f = (f+1) % QUE_SIZE;
   }
}

Void main()
  {
     int choice;
     for (;;)
  {
     printf (" \n1: insertrear
                \n2: delete front
                \n3: display
                \n4: exit \n");
       printf ("enter the choice \n");
        scanf ("%d, &choice);

    switch (choice)

    {  case1 : printf ("enter items to be
                        inserted \n");
              scanf ("%d", &item);
              insertrear ();
               break;
```

```
case 2: item = deletefront();
        if (item == -1)
        printf ("Queue is empty \n");
        else
        printf ("Item deleted =
                %d \n", item);
        break;

case 3: displayQ();
        break;
default : exit(0);

    }
  }
}
```