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 delete front()
{
  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: insert rear
              \n2: delete front
              \n3: display
              \n4: exit \n");
      printf (" enter the Choice \n");
        scanf ("%d, &choice);

  switch (choice)

  {   case 1: printf ("enter items to be
                          inserted \n");
               scanf ("%d", &item);
                insertrear ();
                 break;
case2 : 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);

    }
   }
  }
```