LAB-5 : CIRCULAR QUEUE:

```
#include <stdlib.h>
#include <stdio.h>
#define SIZE 5

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

void insertrear ()
        if (count == SIZE) {
                printf ("queue overflow");
        }
        printf ("Enter the element");
        scanf ("%d", &item);

        rear = (rear+1) % SIZE;
        q[rear] = item;
        count ++;


void deletfront ()
        if (count == 0) {
                printf ("queue is Empty");
        }
        else {
        item = q[front];
        printf ("\n item deleted is = %d \n",
                        q[item]);
```

```
        front = (front+1) % SIZE;
        count--;
    }
}

void display(){
    int i, f;
    if(count == 0){
        pf("Queue is Empty");
    }
    f = front;
    printf("ITEMS of Queue\n");
    for(i=0; i<count; i++){
        pf("%d\t", q[f]);
        f = (f+1) % SIZE;
    }
}

int main(){
    int choice;
    for(;;){
        pf("\nEnter options\n
            1. Insert\n2. delete\n3. display\n4 Exit");
        scanf("%d", &choice);
```

```
switch (choice) {
        case 1: insertrear()
                break;
        case 2: deletefront()
                break;
        case 3: display()
                break;
        default : exit(0);
}
```

Output:

◎ Enter operations
1. insert
2. Delete
3. Display
→ 1
→ Enter item to be inserted
→ 45
◎ Enter operations
1. insert
2. Delete
3. Display
→ 1.
→ Enter item
→ 20
◎ Enter operations
1. insert → 3.
2. Delete → Elements are:
3. Display 45 20

* PRIORITY QUEUE :

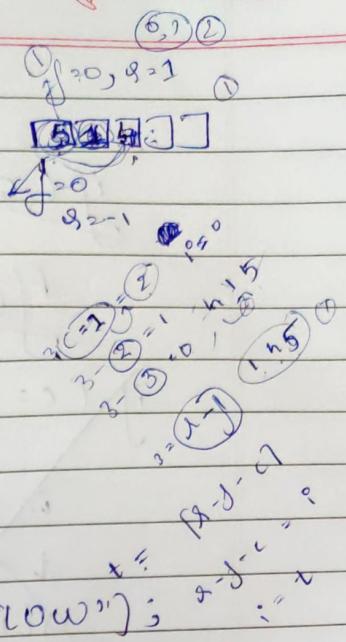
```
#include <stdio.h>
#include <stdlib.h>

#define size 5

int item, q[size], f=0, r=-1;
int c = -1

void insert_rear(){
    if (r=size-1){
        pf("OVERFLOW");
        return;
    }
    pf("Enter the Element to be inserted");
    sf("%d", &item);
    q[++r] = item;
}

→ priority smallest no. = highest priority.
void delet_front(){
    if (r==f);
    pf("Element deleted is %d", q[f]);
    return
}
if(f>r){
(Q is empty)
}
for(i=0; i<r-f; i++)
{
    for(j=0; j<r-f; j++)
    {
        if (q[i] <= q[j])
            c++
```



}

~~if (c == r-f)~~ (3) → find loop out

~~Pf (element deleted is %d, q[i]);~~

if (c == (r-f)) {

int t;
t = a[i];
a[i] = a[j];
a[j] = t;

}

t = q[r-f-c];
q[r-f-c] = q[i];
q[i] = t;

(3) → end loop

}

[1][4][5][ ]
f=0 r=2

0, 1, 2
r-f

Pf ("Element delt is %d, q[f]);
f++;

}

void display() {
if (f > r) {
Pf (QUEUE empty);
return;
}
Pf ("Elements are");
for (int i=0; i < r-f; i++);
{
Pf ("%d", q[i]);
}