PAGE No.
DATE

PRACTICE PROGRAM – MULTIPLE PRIORITY QUEUE

Q. Practice Program :-
Simulate the working of a multiple priority queue.

```
#include<stdio.h>
#include<stdlib>
#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("\n1: PQ insert\n");
 printf("\n2: PQ delete\n");
  printf("\n3: PQ display\n");
printf("\n4: Exit\n");
printf("\n1-4: Exit\n");
printf("\nenter the choice\n");
scanf("%d", &ch);
Switch(ch)
{
case 1: printf("\nEnter the priority number\n");
        scanf("%d", &pr);
        if(pr>0 && pr<4);
        if(pr>0 && pr<4)
        pqinsert(pr-1);
        else
```

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

```
else
{
printf("deleted item is %od of queue %od\n",
                                queue[i][front[i]]
                                i+1);

front[i]++;
return;
}}
display()
{int i,j;
for(int i=0; i<3; i++)
 {
 if(rear[i]==front[i]-1)
 {
 printf("\queue empty %d\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;
}
```

LAB PROGRAM PRACTICE (PRIORITY QUEUE)

PAGE No.
DATE / /

0. WAP to simulate ascending priority queue.

```
#include<stdio.h>
#include<stdlib.h>
#define MAX 5
void insert()
{
 if(r==MAX-1)
  {
  printf("Queue overflow");
  q[++r]=item;
  sort();
 }
void sort()
 {
 int i, key, j;
for(int i=1; i<=r; i++)
 {
  key=q[i];
  j=i-1;
  while(j>=0 && q[j]>key)
   {
    q[j+1]=q[j];
    j=j-1;
   }
  q[j+1]=key;
 }
}
int delete()
 { if(r==-1)
      printf("Queue Underflow\n");
    else
```

DATE / /

```
void display()
{
 int i;
 if(r==-1)
  {
  printf("\n Queue underflow");
 printf("\n Element of queue are:");
 for(i=f;i<=r;i++)
   {
    printf("%d\n",q[i]);
 }
}
void main()
{
 int ch;
 for(;;)
   {
    printf("Enter\n1.for insertion\n2.deletion
\n3.display\n4.exit\n");
   scanf("%d",&ch);
 switch(ch)
 {
 case 1: printf("Enter the item to be inserted");
          scanf("%d",&item);
          insert(item);
          break;

 case 2: item=delete();
         if(item==-1)
          printf("Queue underflow\n");
        else
         printf("\n Item popped=%d\n",item);
```

PAGE No.
DATE / /

```
break;
case 3: display();
        break;
default: exit(0);
}}
```

Q. WAP to simulate descending queue.

```
#include<stdio.h>
#include<stdlib.h>
#define MAX 5
int f=0, r=-1, item, q[10];
void insert()
{
if(r==MAX-1)
printf("Queue Overflow.\n");
q[++r]=item;
sort();
}
void sort()
{
int i, key, j;
for(i=1; i<=r; i++)
{
key=q[i];
j=i-1;
while(j>=0 && q[j]<key)
{
q[j+1]=q[j];
j=j-1;
}
q[j+1]=key;
}
}
int delete()
{
if(r==-1)
printf("Queue U-deflow\n");
```

```
else
return q[f++];
}
void display()
{
int i;
if(r==-1)
printf("\n Queue underflow!!\n");
printf("\n The element of queue are:\n");
for(i=f;i<=r;i++)
{
printf("%d\n",q[i]);
}
}
void main()
{
int ch;
for(;;)
{
printf("Enter\n1.for insertion\n2.delete
3.display\n4.exit\n");
scanf("%d",&ch);
switch(ch)
{
case 1: printf("Enter the item to be inserted");
scanf("%d",&item);
insert(item);
break;
case 2: item=delete();
if(item==-1)
printf("Queue underflow\n");
else.
```

```
printf("\n Item popped = %d \n", item);
break;
case 3: display();
break;
default: exit(0);
}
}
}
```