Multiple Priority

```
#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("\n\t1. PQinsert\n");
   printf("\n\t2: PQ delete\n");
   printf("\n\t3: PQ-display\n");
   printf("\n\t4: Exit\n");
   printf("\nenter the choice\n");
   scanf("%d", &ch);
   switch (ch)
   {
     case 1: printf("\n enter the priority number\n");
             scanf("%d", &pr);
             scanf if(pr>0 && pr<4)
                    PQinsert(pr-1);
                   else
```

```
printf("\n only 3 priority exists 1 2 3 \n");
 break;
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("\n enter 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("\queue 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("\n queue empty %d\n", i+1);
    else
    {
      printf("\nQUEUE %d:", i+1);
      for (j=front[i]; j<=rear[i]; j++)
      printf("%d\t", queue[i][j]);
    }
  }
  return;
}
```

Ascending Priority Queue.

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 3

int pq[MAX];
int count=0;
int d=0;
void insert(int data){
  int i=0;
  if (count==MAX)
  {
    printf("Queue overflow\n");
    return;
  }
  if (count==0){
    pq[count++]=data;
}
  else {
    for(i=count-1; i>=0; i--){
      if(data<pq[i]){
```

```
        pq[i+1] = pq[i];
    }
    else {
        break;
    }
}
    pq[i+1] = data;
    count++;
    }
}
int removeData() {
    return pq[d++];
}
void display()
{
    int i;
    if (count == 0)
    {
        printf("queue is empty\n");
        return;
    }
    printf("Contents of queue: ");
    for(i=d; i<count; i++)
    {
        printf("%d", pq[i]);
    }
    printf("\n");
}
int main() {
    int choice, item;
    for(;;)
    { printf("\n 1: insert \n 2: delete - smallest \n 3: display
                \n 4: exit \n");
        printf("Enter the choice: ");
        scanf("%d", &choice);
        switch(choice)
        {
```

```
case 1: printf("Enter the item to be inse
scanf("%d", &item);
insert(item);
break;
case 2: item = removeData();
if(item == -1)
printf("Queue is empty\n");
else
printf("item deleted = %d\n", item);
break;
case 3: display();
break;
default: exit(0);
}
}
}
```

Descending Priority Queue

```
#include<stdio.h>
#include<stdlib.h>
#define MAX 3

int pq[MAX];
int count=0;
int d=0;

void insert(int data){
int i=0;
if(count == MAX)
{ printf("Queue overflow\n");
return;
}
if(count ==0){
pq[count++] = data;
} else {
```

```
for (i = count - 1; i >= 0; i--) {
    if (data > pq[i]) {
        pq[i+1] = pq[i];
    } else {
        break;
    }
}
pq[i+1] = data;
count++;
}
}
int removeData() {
    return pq[d++];
}
void display()
{ int i;
if (count == 0)
{ printf("queue is empty\n");
    return;
}
printf("Contents of queue: ");
for (i = d; i < count; i++)
{ printf("%d", pq[i]);
}
printf("\n");
}
int main() {
    int choice, item;
    for (;;)
    { printf("\n1: insert\n2: delete_largest\n3: display\n4: exit\n");
        printf("Enter the choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
```

```
case 1 : printf ("Enter the item to be inserted :");
scanf ("%d", &item);
 insert (item);
 break;
case 2: item = removeData();
 if (item == -1)
 printf ("Queue is empty \n");
  else
  printf ("item deleted = %d \n", item);
   break;
   case 3: display();
   break;
   default : exit (0);
   }
  }
}
```