PAGE NO.
DATE / /

MP - lab 5

① Ascending Order PQ

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define Max 4
int pq[max];
int count = 0;
int d = 0;
Void insert (int data)
{ int i=0;
if (count == = Max)
{ printf (" Queue Overflow \n.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++];
}
```

PAGE NO.
DATE / /

```
void display()
{ int i;
  if (count == 0)
{ printf("queue is empty \n");
  return;
}
Printf("Contents of Queue:");
  for (i=0; i< count; i++)
{ printf("%d", pq[i])
  { printf("\n"); }
  int main()
  {
   int choice, item;
   for(;;)
{
printf("\n 1: insert 2: delete-smallest 3: display
        4: 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); }}}
```

PAGE NO.
DATE / /

② Multiple Priority Queue

```
#include<stdio.h>
#define N 3
int queue[3][N], 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 1: PQ Insert \n 2: PQ delete \n 3: PQ display
          \n 4: exit \n");
printf("enter your choice :\n");
  scanf("%d", &ch);
   switch(ch)
{ case 1: printf("\n enter the priority No : \n");
          scanf("%d", &pr);
    if(pr>0 && pr<4)
      pqinsert(pr-1);
     else
    printf("\n only 3 priority exists 1 2 3 \n");
      break;
  Case 2: pq delete();
       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;
  }
  pq_delete()
 {
  int i;
  for(i=0; i<3; i++)
{
  if (rear[i] = front[i] - 1)
  printf("\n 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("queue empty %d \n", i+1);
  else
{ printf("Queue %d :", i+1);
  for (j=front[i]; j<=rear[i]; j++)
   printf("%d ", queue[i][j]);
} }  return; }
```