Dequeue

```
#include<stdio.h>
#include<stdlib.h>
#define qsize 5
int f=0, r=-1, ch;
int item, q[10];
int isfull() {
   return (r==qsize-1)?1:0;
}
int isempty(){
   return (f>r)?1:0;
}
void insert_rear() {
 if (isfull()){
 printf("Queue Overflow\n");
    return;
 }
  r+=1;
  q[r] = item;
}
void deletefront() {
 if (isempty()) {
 printf("Queue empty\n");
    return; }
 printf("item deleted is %d\n", q[f++]);
    if (f>r) {
       f=0;
       r=-1;
    }
}
```

```
void insertfront() {
    if (f != 0) {
        f--;
        q[f] = item;
        return;
    }
    else if (f == 0 && r == -1) {
        q[++r] = item;
        return; }
    else
        printf("Insertion not possible\n");
}
void deleterear() {
    if (isempty()) {
        printf("Queue empty\n");
            return;
    }
    printf("item deleted is %d\n", q[r--]);
    if (f > r) {
        f = 0;
        r = -1; }
}
void display() {
    int i;
    if (isempty()) {
        printf("queue empty\n");
            return;
    }
    for (i=f; i<=r; i++)
        printf("%d\n", q[i]);
}
```

```
int main() {
 for (;;) {
printf("\n1. insertrear \n 2. insertfront \n 3. deleterear\n 4. delet
                        front\n 5. display\n 6. exit \n");
  printf(" Enter choice.\n");
   scanf("%d", &ch);
   ~~scanf~~
    switch (ch) {
   case 1: printf("Enter the item\n");
           scanf("%d", &item);
            insertrear();
            break;
   case 2: printf("enter the item\n");
           scanf("%d", &item);
            insert front();
            break;
   case 3: deletefront();
            break;
   case 4: deleterear();
            break;
   case 5: display();
   default: exit(0);
 }
}
```

Multiple priority Queue

```
#include<stdio.h>
#include<stdlib.h>
void pqdelete();
void display();
void pqinsert(int pr);
#define N 3
int queue[3][N];
int front[3] = {0,0,0};
int rear[3] = {-1,-1,-1};
int item, pr;
int main(){
int ch;
while(1){
printf("\n PRIORITY QUEUE\n");
printf("* * * * \n");
printf("\n1: PQinsert\n");
printf("\n2: PQ delete\n");
printf("\n4: Exit\n");
printf("Enter the choice.\n");
  scanf("%d", &ch);
  switch(ch){
case 1: printf("Enter the priority numbe\n");
        scanf("%d", &pr);
        if(pr>0 && pr<4)
         pqinsert(pr-1);
        else
        printf("Only 3 priority exicete\n");
            scanf("%d" break;

case 2: pqdelete();
          break;
case 3: display();
        break;
```

```
    case 4 : exit(0);
  }
 }
}

void insert(int pr) {
 if (rear[pr] == N-1)
   printf("Queue Overflow \n");
   else {
   printf("Enter the item \n");
     scanf("%d", &item);
     rear[pr]++;
     queue[pr][rear[pr]] = item;
  }
}
void 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]++;
 }}
}
void 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("In queue %d : ", i+1);
   for (j=front[i]; j<=rear[i]; j++)
       printf("%d ", q[i][j]);
}}}
```

Ascending Priority Queue :

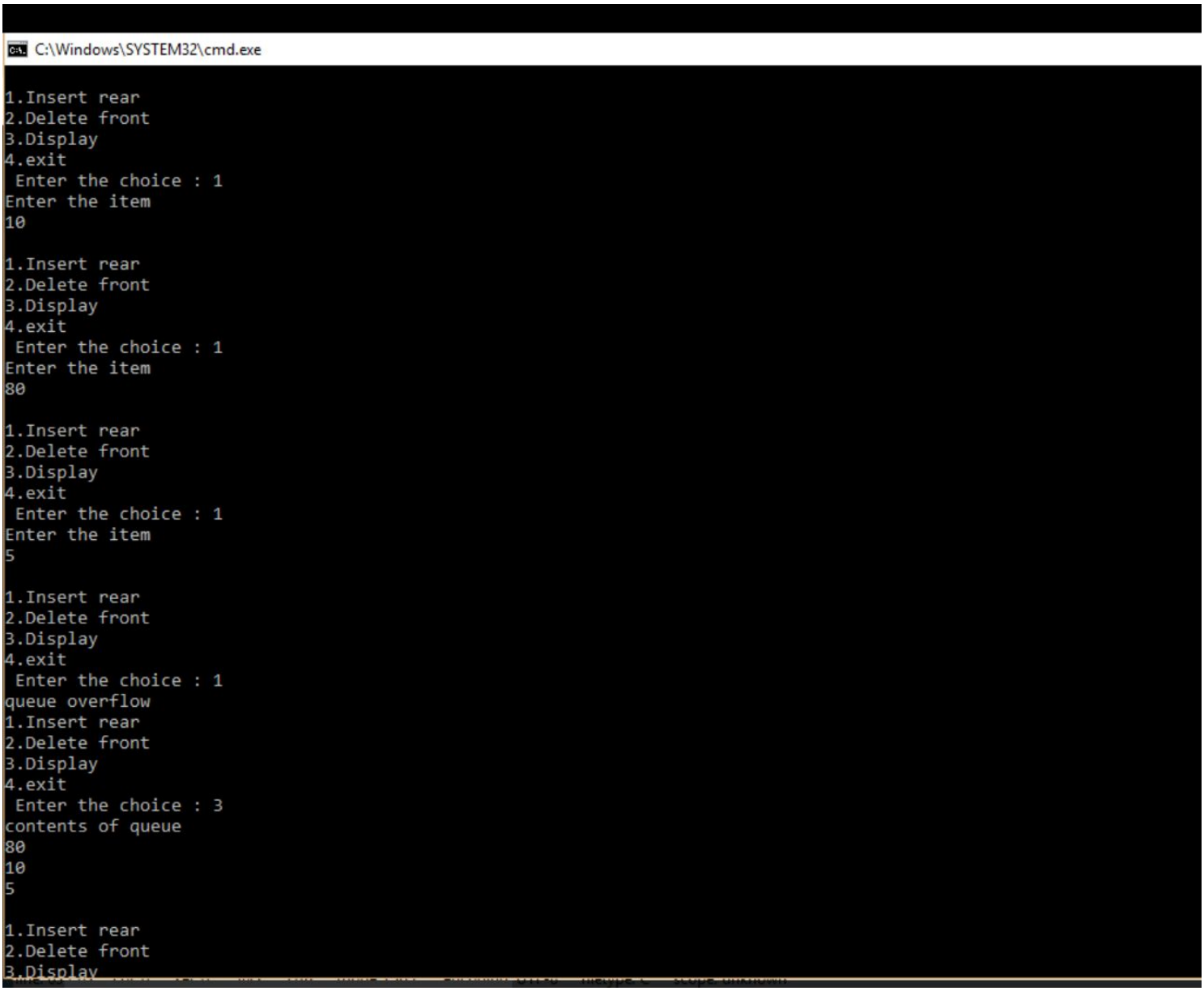
```
#include <stdio.h>
#include <stdlib.h>
#define que_size 3
int item, front=0, rear=-1, q[que_size];
void insertrear() {
  if (rear == que_size-1) {
    printf("Queue Overflow\n");
      return;
  }
  int i,j, min;
  printf("Enter the item\n");
  scanf("%d", &item);
  q[++rear] = item;
  if (rear>=1) {
  for (i=1; i<=rear; i++) {
        min = q[i];
        j = i-1;
      while (j>=0 && q[j]>min) {
          q[j+1] = q[j];
          j--; }
        q[j+1] = min;
} }
}
  int deletefront() {
   if (front > rear) {
       front=0;
       rear=-1;
       return -1;
   }
    return q[front++];
}
 void displayq() {
     int i;
     if (front>rear) {
     printf("queue is empty\n");
         return; }
```

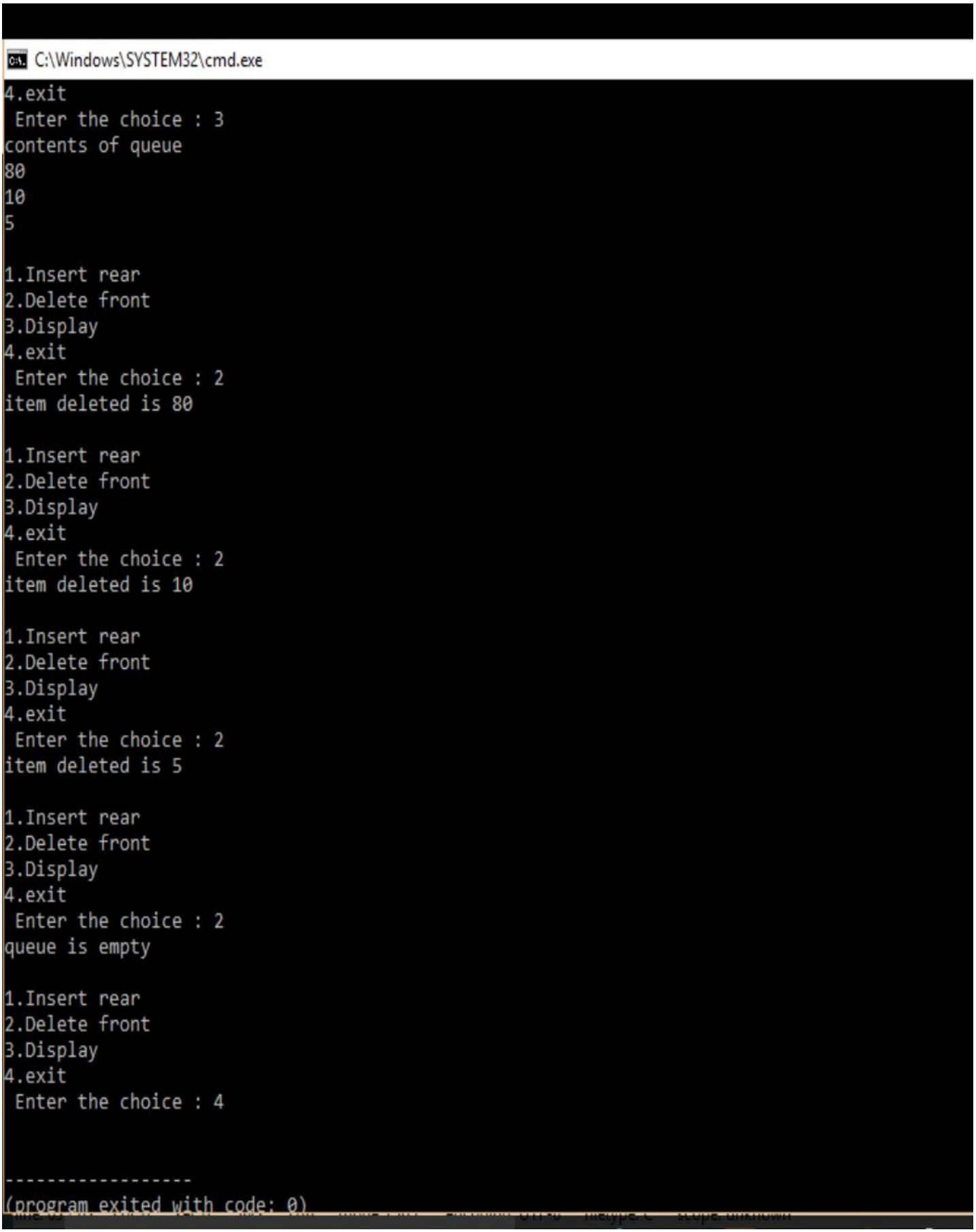
```
printf("Contents of queue \n");
for (i = front; i <= rear; i++) {
    printf("%d \n", q[i]);
}
}
int main() {
  int choice;
  for(;;) {
printf("\n1. Insert rear \n2. Delete front \n3. Display \n4. exit \n");
    printf("Enter the choice :");
    scanf("%d", &choice);
    switch (choice) {
    case 1 : insert rear();
             break;
    case 2 : item = delete front();
             if (item == -1)
              printf("Queue is empty \n");
             else
              printf("item deleted is %d \n", item);
              break;
    case 3 : display q();
              break;
    default : exit(0);
  }
}
}
```

Descending Priority Queue:

```
#include<stdio.h>
#include<stdlib.h>
#define quesize 3
int item, front=0, rear=-1, q[que_size];
void insertrear(){
 if (rear == que_size-1) {
   printf("Queue Overflow\n");
     return;
  }
 int  i,j, max;
 Printf("Enter the item\n");
 scanf("%d", &item);
  q[++rear] = item;
  if (rear>=1) {
 for(j=1 ; i<=rear; i++){
      max = q[i];
       j=i-1;
  while (j>=0 && q[j]<max){
          q[j+1] = q[j];
           j--; }
          q[j+1] = min;
} } }
  int deletefront (){
    if (front > rear) {
     front =0;
     rear = -1;
   }  return -1;
    return q[front++];
}
void displayq (){
   int i;
 if ( front > rear) {
     printf ("Queue is empty\n");
          return ; }
    printf("Contents of queue are\n");
     for(i=front; i<=rear; i++)
     printf("%d\n",
```

```
int 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 : insertrear();
           break;
  case 2: item = deletefront ();
          if (item ==-1)
           printf ("Queue empty \n");
          printf("Deleted item is %d \n", item);
            break;
  case 3:  displayq();
           break;
default:    exit (0);
            break;
```


C:\Windows\SYSTEM32\cmd.exe
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
10
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
80
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
5
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
queue overflow
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 3
contents of queue
80
10
5
1.Insert rear
2.Delete front
3.Display


C:\Windows\SYSTEM32\cmd.exe
4.exit
Enter the choice : 3
contents of queue
80
10
5
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
item deleted is 80
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
item deleted is 10
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
item deleted is 5
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
queue is empty
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 4
(program exited with code: 0)

ueue.c | multiple priorityq.c | ascending priorityq.c | descending pq.c | multiple pq.c

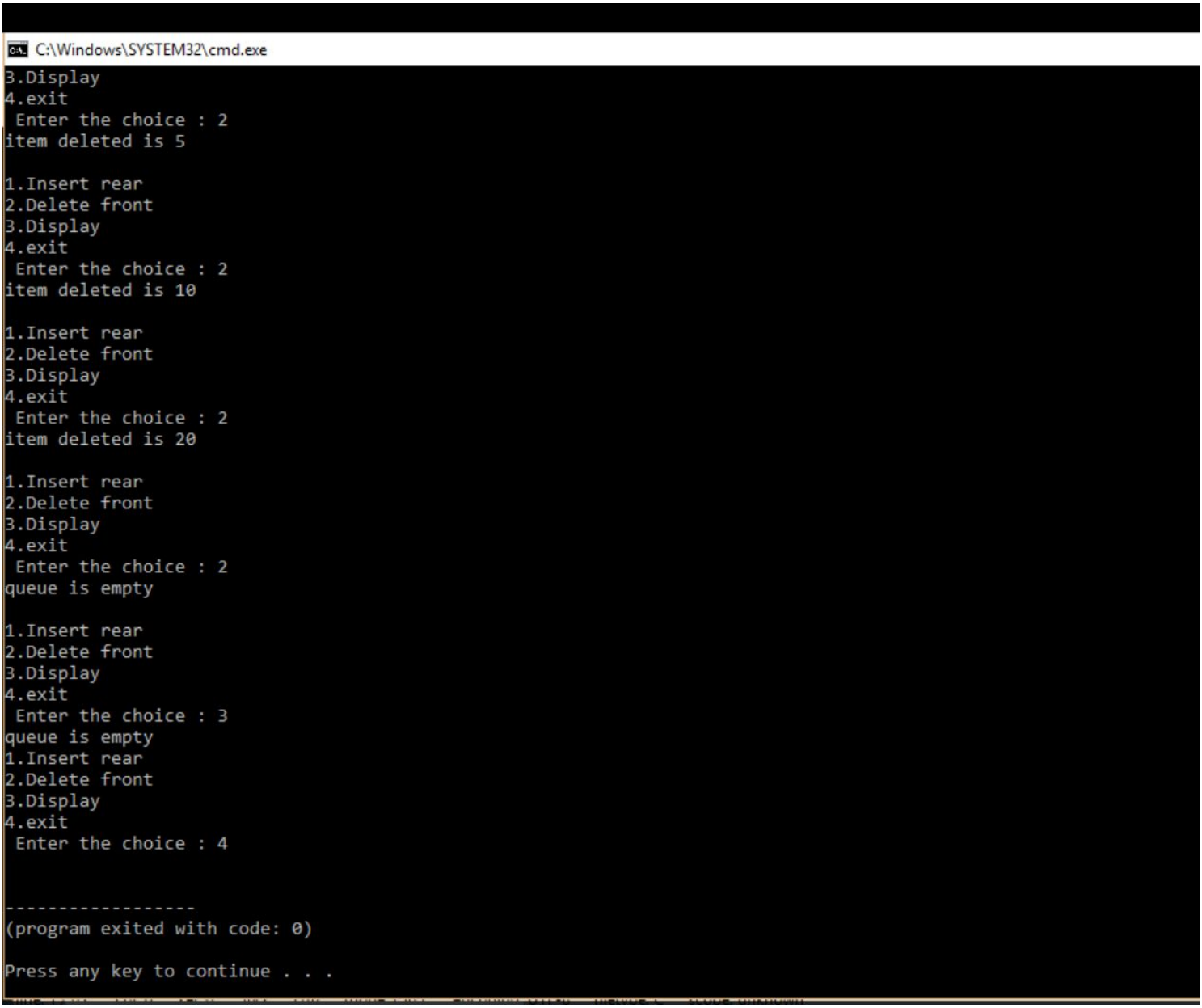
```c
#include<stdio.h>
#include<stdlib.h>
#define que_size 3
int item,front=0,rear=-1,q[que_size];
void insertrear(){
    if(rear==que_size-1){
        printf("queue overflow");
        return;
    }
    int i,j,min;
    printf("Enter the item\n");
    scanf("%d",&item);
    q[++rear]=item;
    if(rear>=1){
        for(i=1;i<=rear;i++){
            min=q[i];
            j=i-1;
            while(j>=0 && q[j]>min){
                q[j+1]=q[j];
                j--;
            }
            q[j+1]=min;
}  }   }
int deletefront(){
    if(front>rear){
        front = 0;
        rear = -1;
         return -1;
     }
    return q[front++];
}
void displayq(){
    int i;
```

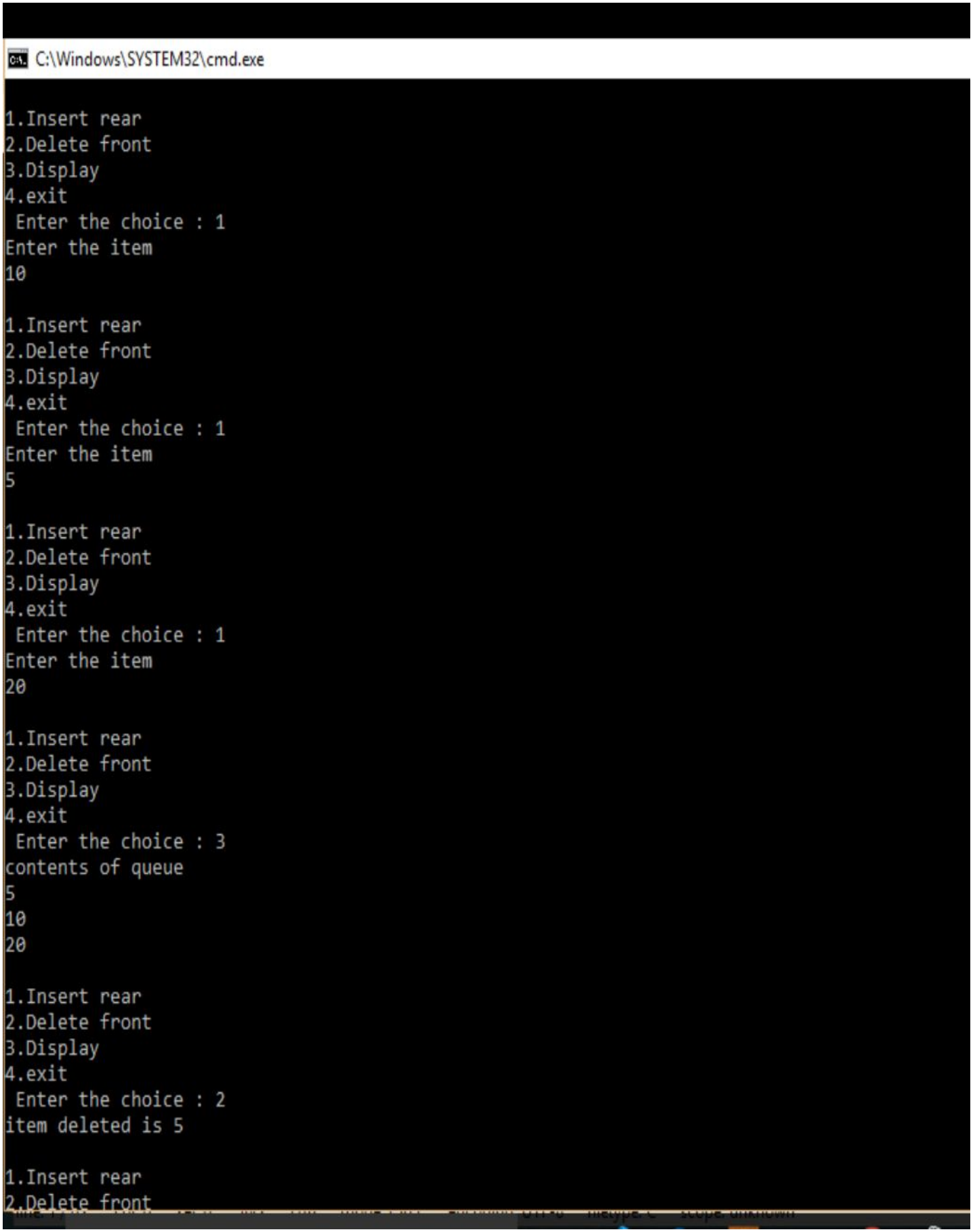
1 / 63 col: 0 sel: 0 INS TAB mode: CRLF encoding: UTF-8 filetype: C scope: unknown

Dequeue.c | multiple priorityq.c | ascending priorityq.c | descending pq.c | multiple pq.c

```c
}
void displayq(){
    int i;
    if(front>rear){
        printf("queue is empty");
        return;}
    printf("contents of queue \n");
    for(i=front;i<=rear;i++){
        printf("%d\n",q[i]);
    }
}
int main(){
    int choice;
    for(;;){
        printf("\n1.Insert rear \n2.Delete front \n3.Display \n4.exit \n ");
        printf("Enter the choice : ");
        scanf("%d",&choice);
        switch(choice){
            case 1:insertrear();
                   break;
            case 2:item=deletefront();
                   if(item==-1)
                   printf("queue is empty\n");
                   else
                   printf("item deleted is %d \n",item);
                   break;
            case 3:displayq();
                   break;
            default:exit(0);
        }
    }
}
```

line: 63 / 63 col: 0 sel: 0 INS TAB mode: CRLF encoding: UTF-8 filetype: C scope: unknown


C:\Windows\SYSTEM32\cmd.exe
3.Display
4.exit
Enter the choice : 2
item deleted is 5
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
item deleted is 10
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
item deleted is 20
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
queue is empty
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 3
queue is empty
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 4
(program exited with code: 0)
Press any key to continue . . .


C:\Windows\SYSTEM32\cmd.exe
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
10
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
5
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
20
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 3
contents of queue
5
10
20
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
item deleted is 5
1.Insert rear
2.Delete front

ieue.c | multiple priorityq.c | ascending priorityq.c | descending pq.c | multiple pq.c

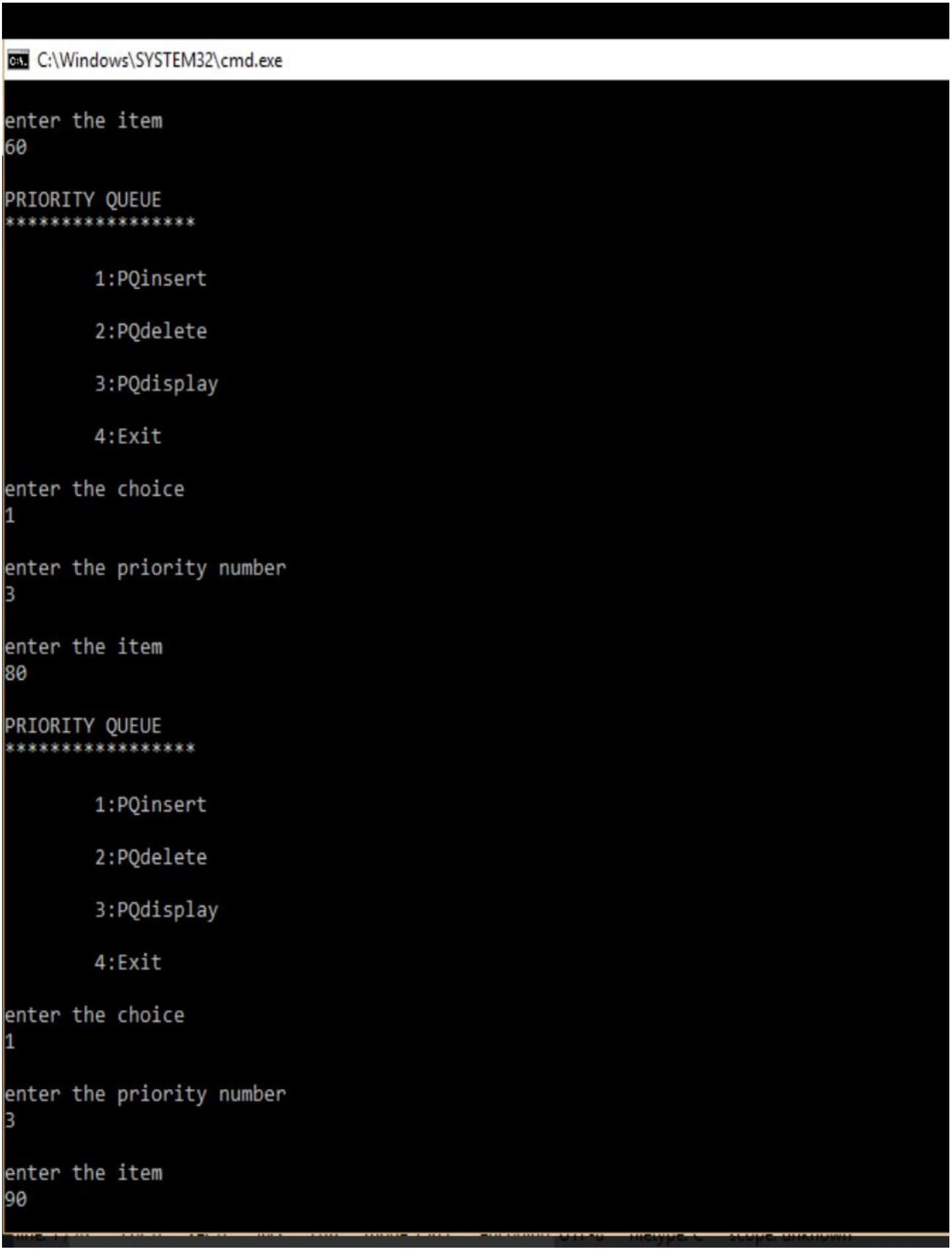
```c
#include<stdio.h>
#include<stdlib.h>
#define que_size 3
int item,front=0,rear=-1,q[que_size];
void insertrear(){
    if(rear==que_size-1){
        printf("queue overflow");
        return;
    }
    int i,j,max;
    printf("Enter the item\n");
    scanf("%d",&item);
    q[++rear]=item;
    if(rear>=1){
        for(i=1;i<=rear;i++){
            max=q[i];
            j=i-1;
            while(j>=0 && q[j]<max){
                q[j+1]=q[j];
                j--;
            }
            q[j+1]=max;
}   }   }
int deletefront(){
    if(front>rear){
        front = 0;
        rear = -1;
         return -1;
     }
    return q[front++];
}
void displayq(){
    int i;
```

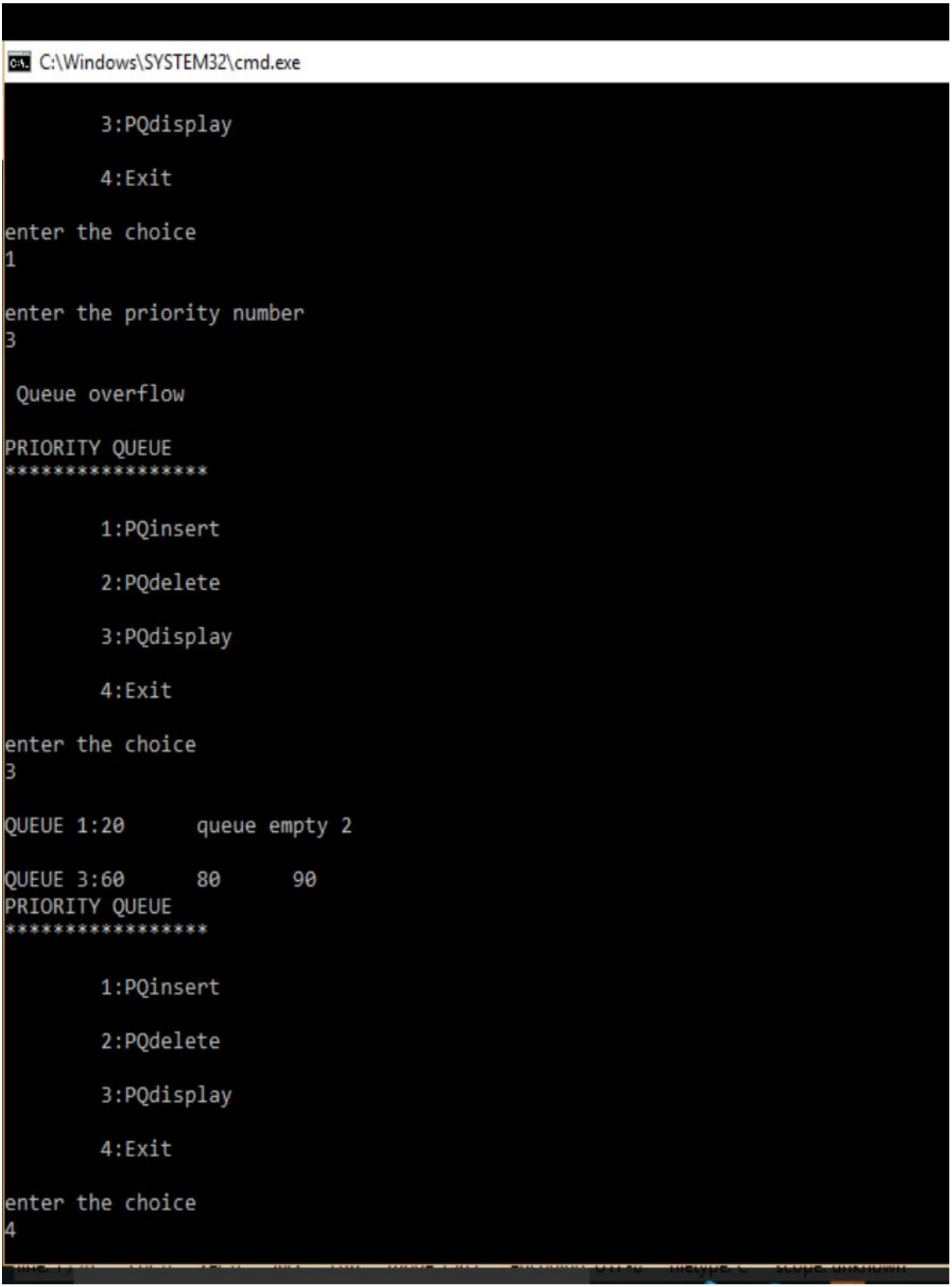
/ 63 col: 0 sel: 0 INS TAB mode: CRLF encoding: UTF-8 filetype: C scope: unknown

Open Save Save All | Revert Close | Back Forward | Compile Build Execute | Color Chooser

ieue.c | multiple priorityq.c | ascending priorityq.c | descending pq.c | multiple pq.c

```
}
void displayq(){
    int i;
    if(front>rear){
        printf("queue is empty");
        return;}
    printf("contents of queue \n");
    for(i=front;i<=rear;i++){
        printf("%d\n",q[i]);
    }
}
int main(){
    int choice;
    for(;;){
        printf("\n1.Insert rear \n2.Delete front \n3.Display \n4.exit \n ");
        printf("Enter the choice : ");
        scanf("%d",&choice);
        switch(choice){
            case 1:insertrear();
                   break;
            case 2:item=deletefront();
                   if(item==-1)
                   printf("queue is empty\n");
                   else
                   printf("item deleted is %d \n",item);
                   break;
            case 3:displayq();
                   break;
            default:exit(0);
        }
    }
}
```


C:\Windows\SYSTEM32\cmd.exe
enter the item
60
PRIORITY QUEUE
****
1:PQinsert
2:PQdelete
3:PQdisplay
4:Exit
enter the choice
1
enter the priority number
3
enter the item
80
PRIORITY QUEUE
****
1:PQinsert
2:PQdelete
3:PQdisplay
4:Exit
enter the choice
1
enter the priority number
3
enter the item
90


C:\Windows\SYSTEM32\cmd.exe
3:PQdisplay
4:Exit
enter the choice
1
enter the priority number
3
Queue overflow
PRIORITY QUEUE
****
1:PQinsert
2:PQdelete
3:PQdisplay
4:Exit
enter the choice
3
QUEUE 1:20 queue empty 2
QUEUE 3:60 80 90
PRIORITY QUEUE
****
1:PQinsert
2:PQdelete
3:PQdisplay
4:Exit
enter the choice
4


C:\Windows\SYSTEM32\cmd.exe
PRIORITY QUEUE
****
1:PQinsert
2:PQdelete
3:PQdisplay
4:Exit
enter the choice
1
enter the priority number
2
enter the item
30
PRIORITY QUEUE
****
1:PQinsert
2:PQdelete
3:PQdisplay
4:Exit
enter the choice
2
deleted item is 10 of queue 1
deleted item is 30 of queue 2
queue empty
PRIORITY QUEUE
****
1:PQinsert
2:PQdelete

C:\Windows\SYSTEM32\cmd.exe

```
PRIORITY QUEUE
******************

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

enter the priority number
1

enter the item
10

PRIORITY QUEUE
******************

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

enter the priority number
1

enter the item
20

PRIORITY QUEUE
******************
```

C:\Windows\SYSTEM32\cmd.exe

```
enter the choice
2
deleted item is 10 of queue 1
deleted item is 30 of queue 2
queue empty

PRIORITY QUEUE
******************

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
40

PRIORITY QUEUE
******************

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

enter the priority number
3

enter the item
60

PRIORITY QUEUE
******************
```

queue.c | multiple priorityq.c | ascending priorityq.c | descending pq.c | multiple pq.c

```c
#include<stdio.h>
#include<stdlib.h>
void pqdelete();
void display();
void pqinsert(int pr);
#define N 3
int queue[3][N];
int front[3]={0,0,0};
int rear[3]={-1,-1,-1};
int item,pr;
int main()
{
int ch;
while(1)
{
printf("\nPRIORITY QUEUE\n");
printf("*****************\n");
printf("\n\t1:PQinsert\n");
printf("\n\t2:PQdelete\n");
printf("\n\t3:PQdisplay\n");
printf("\n\t4: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)
        pqinsert(pr-1);
        else
        printf("only 3 priority exists 1 2 3\n");
        break;
case 2:pqdelete();
```

Open Save Save All | Revert Close | Back Forward | Compile Build Execute

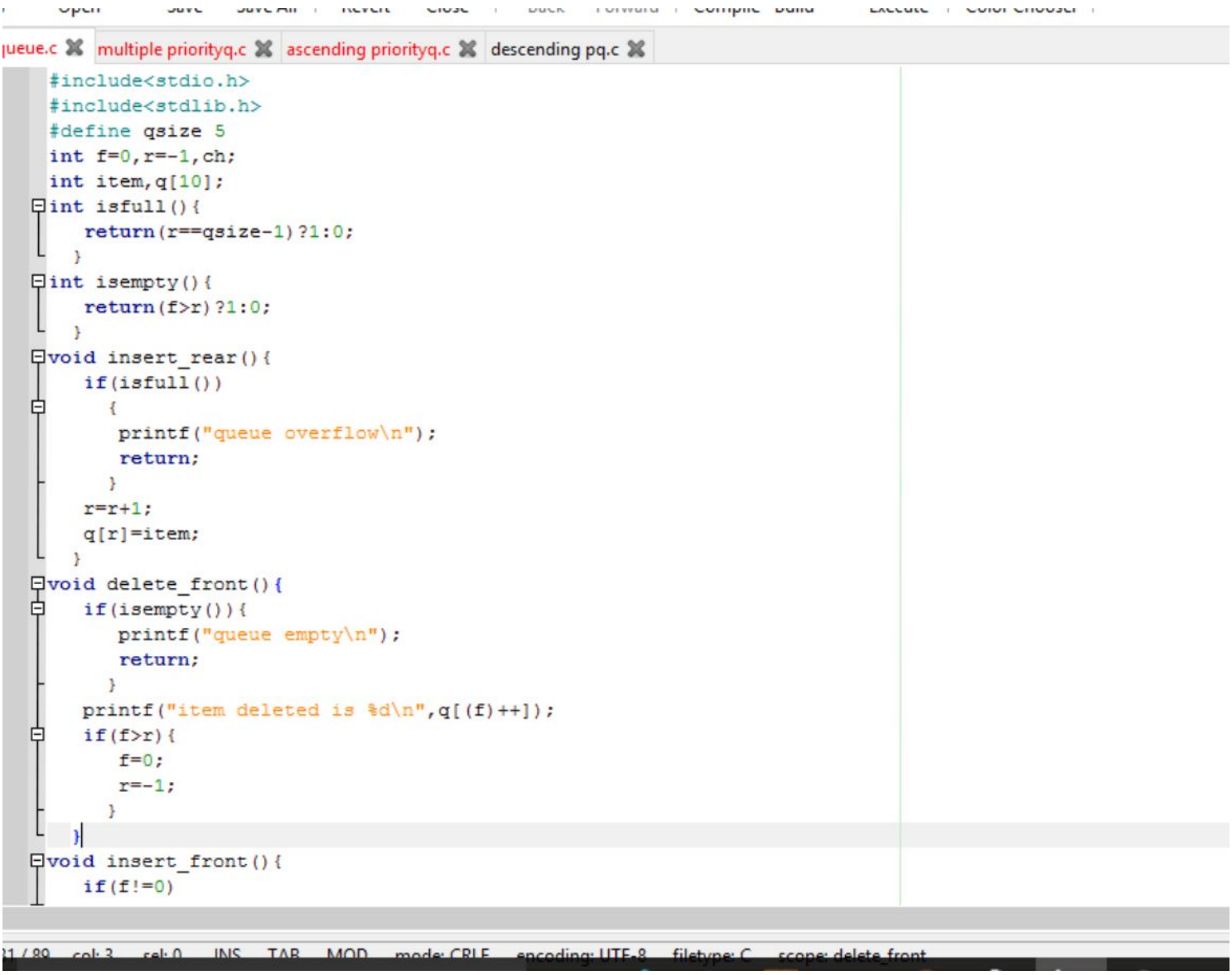
ieue.c | multiple priorityq.c | ascending priorityq.c | descending pq.c | multiple pq.c

```
case 2:pqdelete();
       break;
case 3:display();
       break;
case 4:exit(0);
}
}
}
void pqinsert(int pr)
{
 if(rear[pr]==N-1)
 printf("\n Queue overflow\n");
 else
 {
 printf("\nenter the item\n");
 scanf("%d",&item);
 rear[pr]++;
 queue[pr][rear[pr]]=item;
 }
 }
void 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]++;
  }
 }
```

Open Save Save All | Revert Close | Back Forward | Compile Build Execute | Color Chooser

ueue.c | multiple priorityq.c | ascending priorityq.c | descending pq.c | multiple pq.c

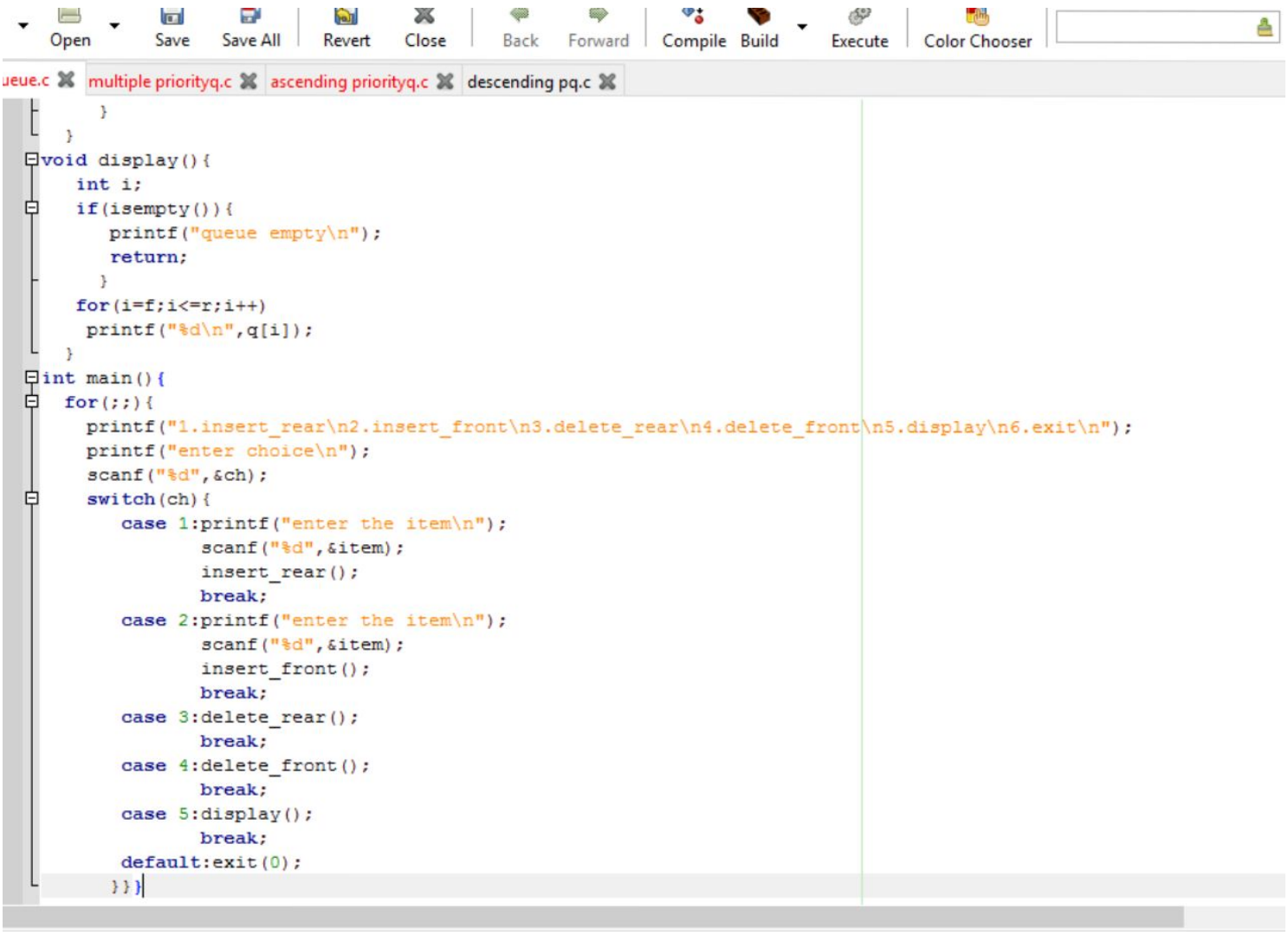
```
 queue[pr][rear[pr]]=item;
 }
 }
void pqdelete()
{
int i;
for(i=0;i<3;i++)
{
  if(rear[i]==front[i]-1)
  printf("queue empty\n");
  else
  {
  printf("deleted item is %d of queue %d\n",queue[i][front[i]],i+1);
  front[i]++;
  }
 }
}
void display()
{
int i,j;
for(i=0;i<3;i++)
{
if(rear[i]==front[i]-1)
  printf("queue empty %d\n",i+1);
else
  {
  printf("\nQUEUE %d:",i+1);
  for(j=front[i];j<=rear[i];j++)
    printf("%d\t",queue[i][j]);
      }
   }
 }
```

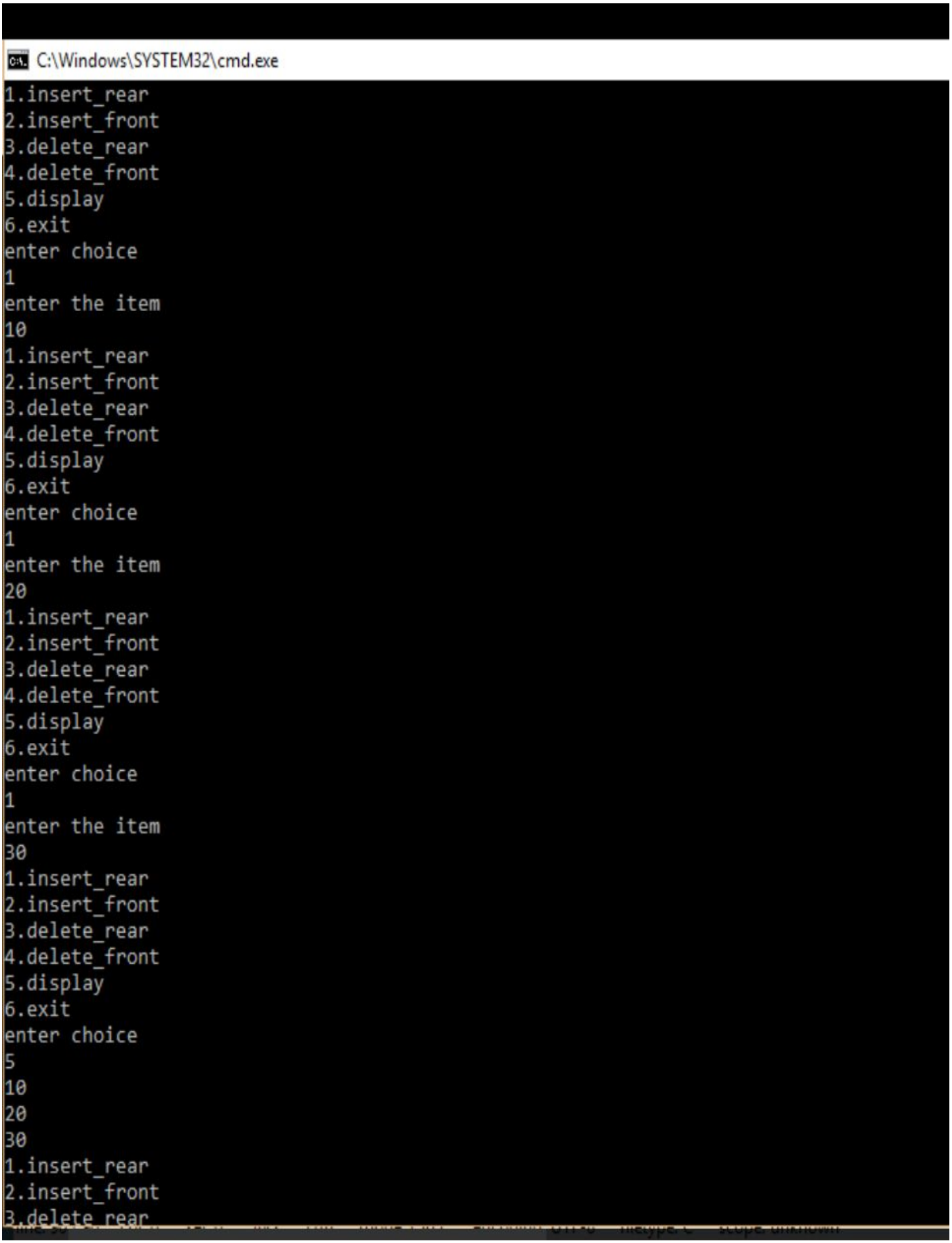

queue.c
multiple priorityq.c
ascending priorityq.c
descending pq.c
#include<stdio.h>
#include<stdlib.h>
#define qsize 5
int f=0,r=-1,ch;
int item,q[10];
int isfull(){
return(r==qsize-1)?1:0;
}
int isempty(){
return(f>r)?1:0;
}
void insert_rear(){
if(isfull())
{
printf("queue overflow\n");
return;
}
r=r+1;
q[r]=item;
}
void delete_front(){
if(isempty()){
printf("queue empty\n");
return;
}
printf("item deleted is %d\n",q[(f)++]);
if(f>r){
f=0;
r=-1;
}
}
void insert_front(){
if(f!=0)

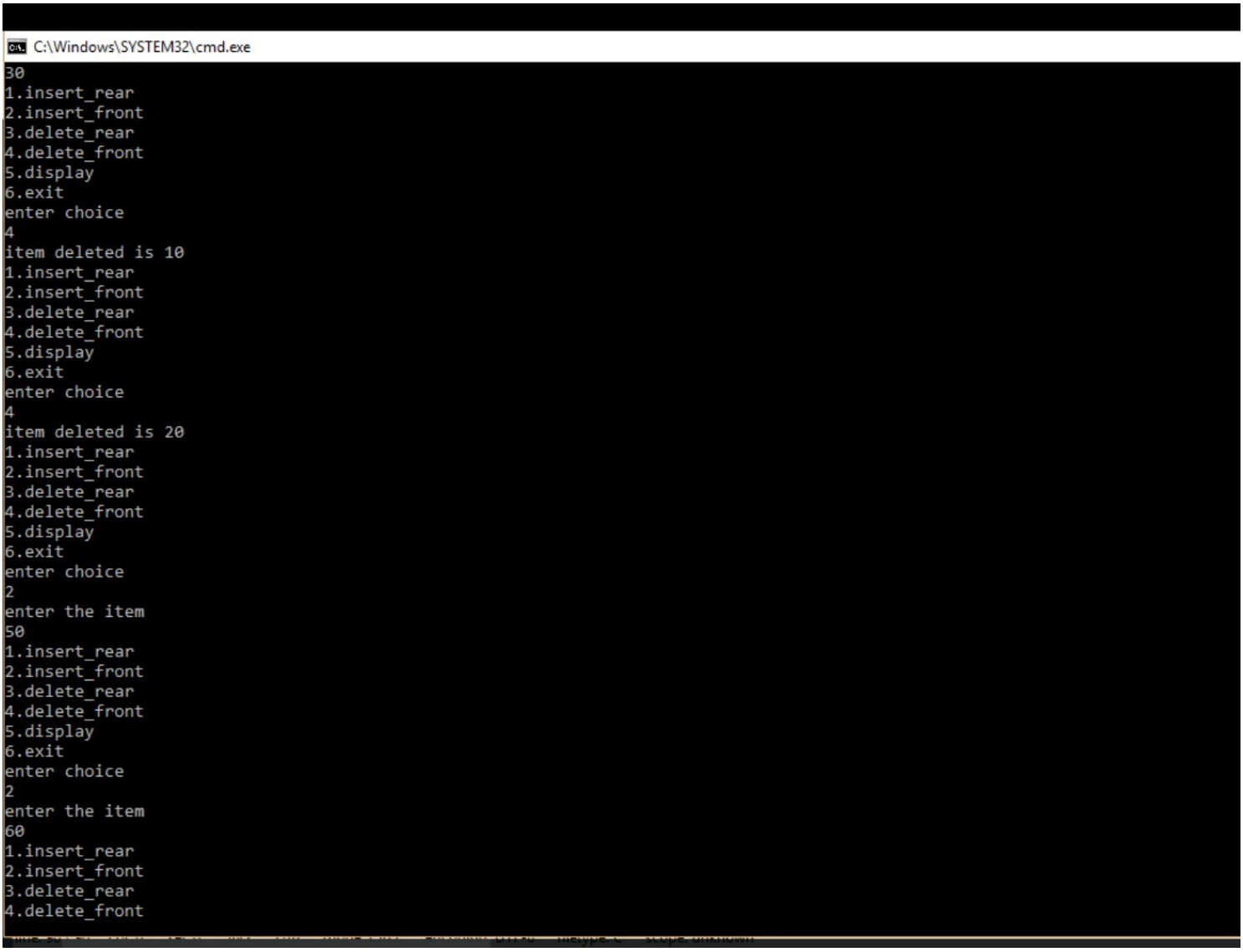
Open Save Save All | Revert Close | Back Forward | Compile Build Execute | Color Chooser

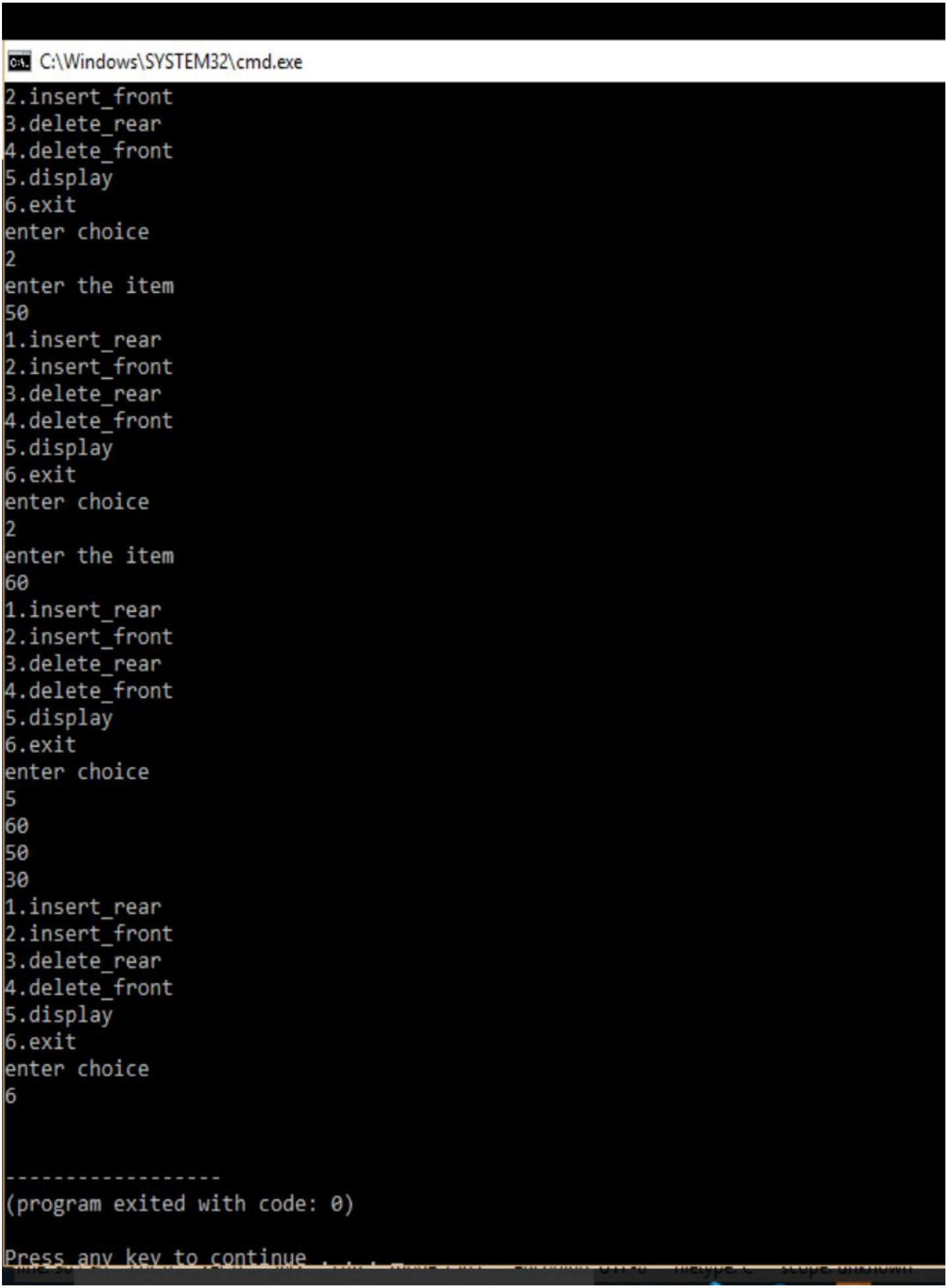
eue.c | multiple priorityq.c | ascending priorityq.c | descending pq.c

```
    if(f!=0)
      {
       f=f-1;
       q[f]=item;
       return;
      }
     else if((f==0)&&(r==-1))
      {
       q[++(r)]=item;
       return;
      }
     else
       printf("insertion not possible\n");
  }
void delete_rear()
  {
   if(isempty()){
      printf("queue is empty\n");
      return;
     }
   printf("item deleted is %d\n",q[(r)--]);
   if(f>r){
      f=0;
      r=-1;
     }
  }
void display(){
   int i;
   if(isempty()){
      printf("queue empty\n");
      return;
     }
   for(i=f;i<=r;i++)
```


Open
Save
Save All
Revert
Close
Back
Forward
Compile
Build
Execute
Color Chooser
ueue.c
multiple priorityq.c
ascending priorityq.c
descending pq.c
}
}
void display(){
int i;
if(isempty()){
printf("queue empty\n");
return;
}
for(i=f;i<=r;i++)
printf("%d\n",q[i]);
}
int main(){
for(;;){
printf("1.insert_rear\n2.insert_front\n3.delete_rear\n4.delete_front\n5.display\n6.exit\n");
printf("enter choice\n");
scanf("%d",&ch);
switch(ch){
case 1:printf("enter the item\n");
scanf("%d",&item);
insert_rear();
break;
case 2:printf("enter the item\n");
scanf("%d",&item);
insert_front();
break;
case 3:delete_rear();
break;
case 4:delete_front();
break;
case 5:display();
break;
default:exit(0);
}}}


C:\Windows\SYSTEM32\cmd.exe
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
10
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
20
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
30
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
5
10
20
30
1.insert_rear
2.insert_front
3.delete_rear


C:\Windows\SYSTEM32\cmd.exe
30
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
4
item deleted is 10
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
4
item deleted is 20
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
2
enter the item
50
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
2
enter the item
60
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front


C:\Windows\SYSTEM32\cmd.exe
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
2
enter the item
50
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
2
enter the item
60
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
5
60
50
30
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
6

(program exited with code: 0)
Press any key to continue


C:\Windows\SYSTEM32\cmd.exe
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
10
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
20
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
30
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item
40
queue overflow1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 3
contents of queue
10
20
30
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
item deleted is 10
1.Insert rear