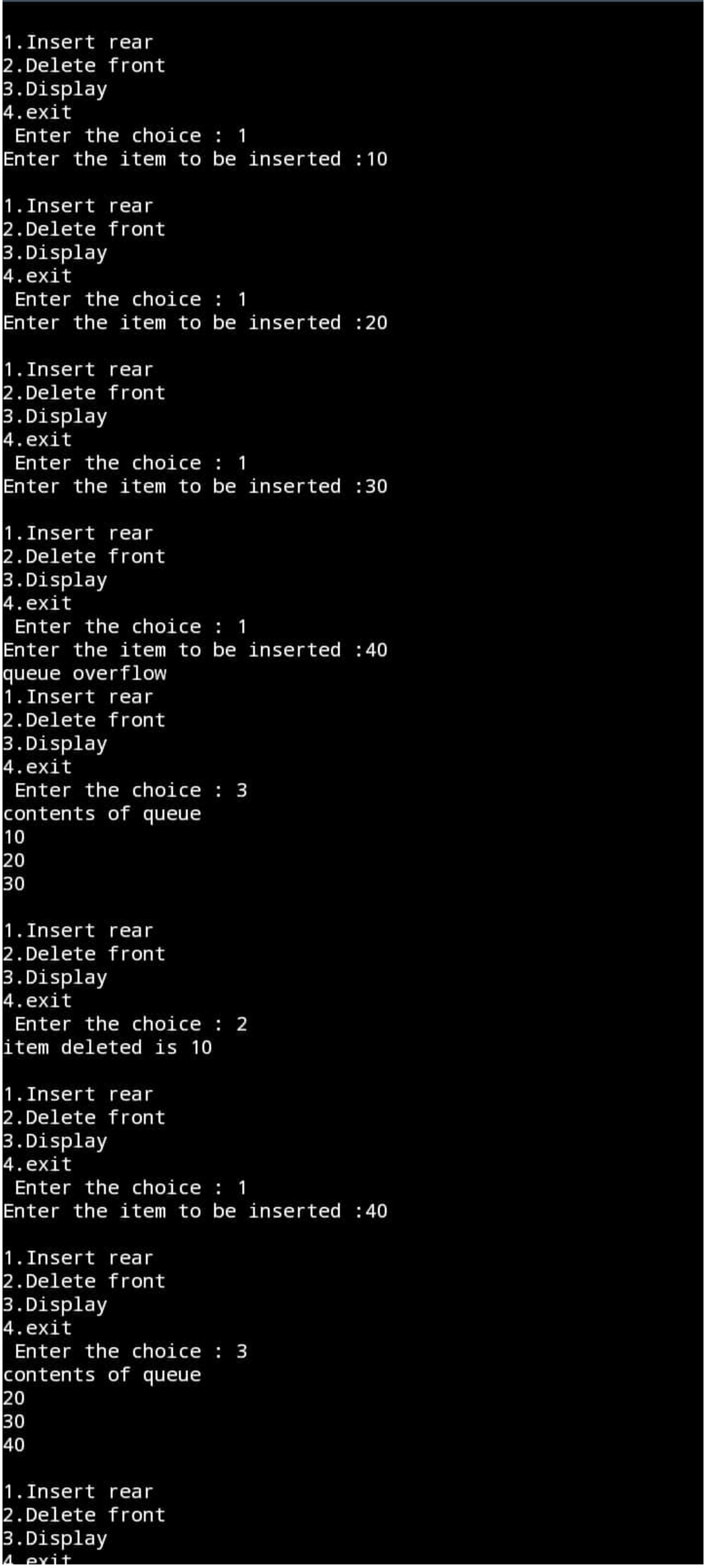
```c
#include<stdio.h>
#include<stdlib.h>
#define que_size 3
int item,front=0,rear=-1,q[que_size],count=0;
void insertrear()
{
   if(count==que_size)
   {
      printf("queue overflow");
      return;
   }
   rear=(rear+1)%que_size;
   q[rear]=item;
   count++;
}
int deletefront()
{
   if(count==0) return -1;
   item = q[front];
   front=(front+1)%que_size;
   count=count-1;
   return item;
}
void displayq()
{
   int i,f;
   if(count==0)
   {
      printf("queue is empty");
      return;
   }
   f=front;
   printf("contents of queue \n");
   for(i=0;i<count;i++)
   {
      printf("%d\n",q[f]);
      f=(f+1)%que_size;
   }
}
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:printf("Enter the item to be inserted :");
                scanf("%d",&item);
                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);
        }
    }
    return 0;
}
```



```
1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :10

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :20

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :30

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :40
queue overflow
1.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
2.Delete front
3.Display
4.exit
 Enter the choice : 1
Enter the item to be inserted :40

1.Insert rear
2.Delete front
3.Display
4.exit
 Enter the choice : 3
contents of queue
20
30
40

1.Insert rear
2.Delete front
3.Display
4.exit
```

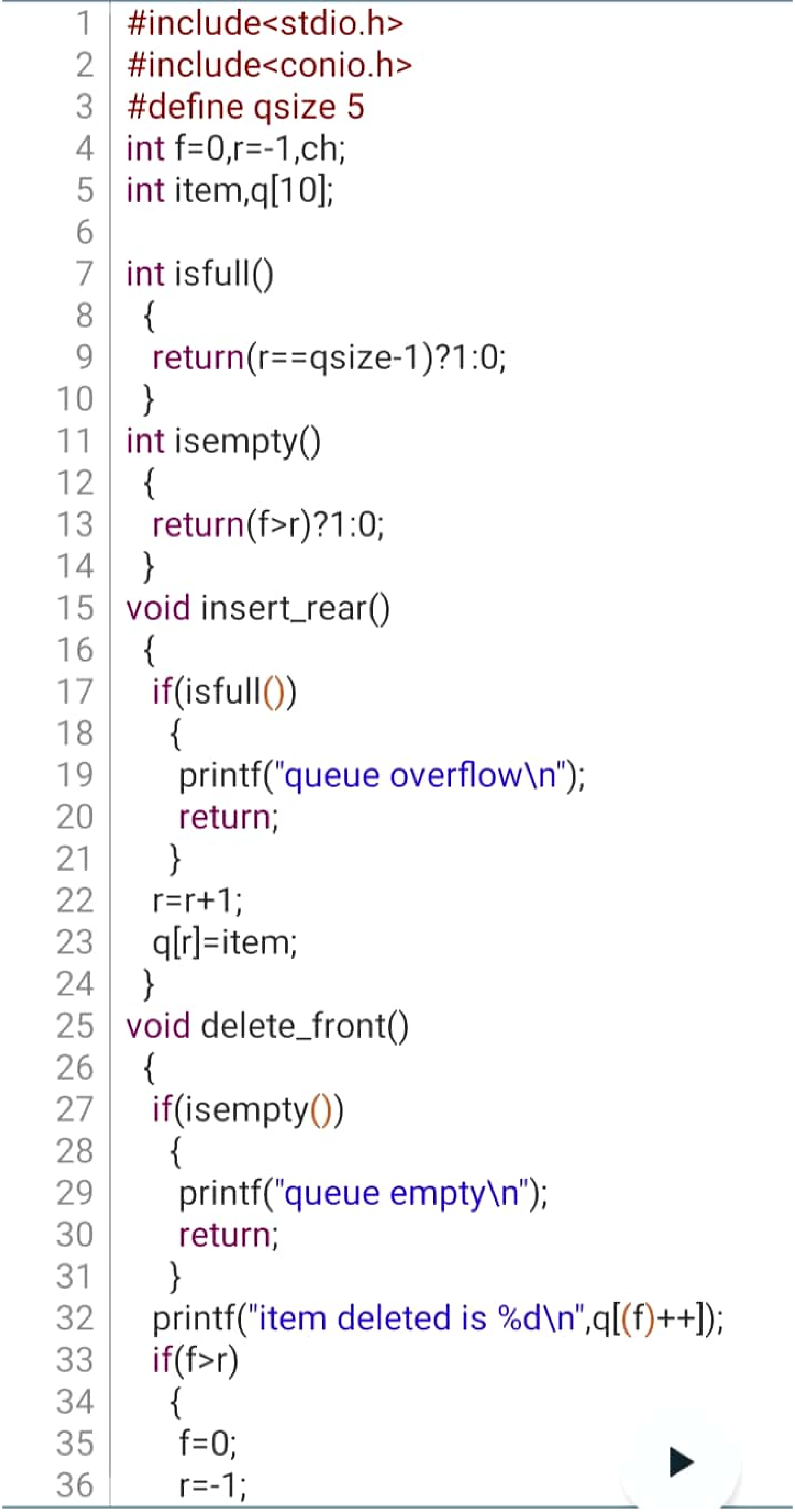
Scanned with CamScanner

CIRCULAR QUEUE

```
# include <STDIO h>
# include <stdlib.h>
# define que-size 3
int item, front = 0, rear=-1, q [que_size], count=0;
void insert_rear()
{
 if (count == que_size)
  {
   printf ("Queue Overflow \n");
   return;
  }
  rear (rear + 1)% que_size;
  q[rear] = item;
  count ++;
 }
int delete_front()
 {
 if (count = = 0)
   return -1;
  item = q[front];
  front = (front +1) % queue-size;
   count --;
   return item;
  }
 void display()
  {
   int i,f;
   if (count = = 0)
    {
      printf ("Queue is empty \n");
    return;
  }
 f = front;
  printf ("Contents of Queue \n");
   for (i=0; i<=count ; i++)
    {
     printf ("%d \n", q[f]);
     f=(f+1)%que_size;
    }
   }
  ~~void~~ int main()
   {
    int choice ;
     for(;;)
     {
     printf ("\n 1. Insert rear \n 2. Delete front \n 3.
                Display \n 4. Exit \n");
     printf ("Enter choice ");
     scanf ("%d", &choice);
     switch (choice)
     {
    Case 1 : printf ("Enter the item to be inserted :");
             scanf("%d", &item);
             insert_rear();
             break;
     Case 2 : item = delete_front ();
             if (item == -1)
             printf ("Queue is empty \n");
             else
           printf ("item deleted = %d\n", item);
           break;
    Case 3 display()
```

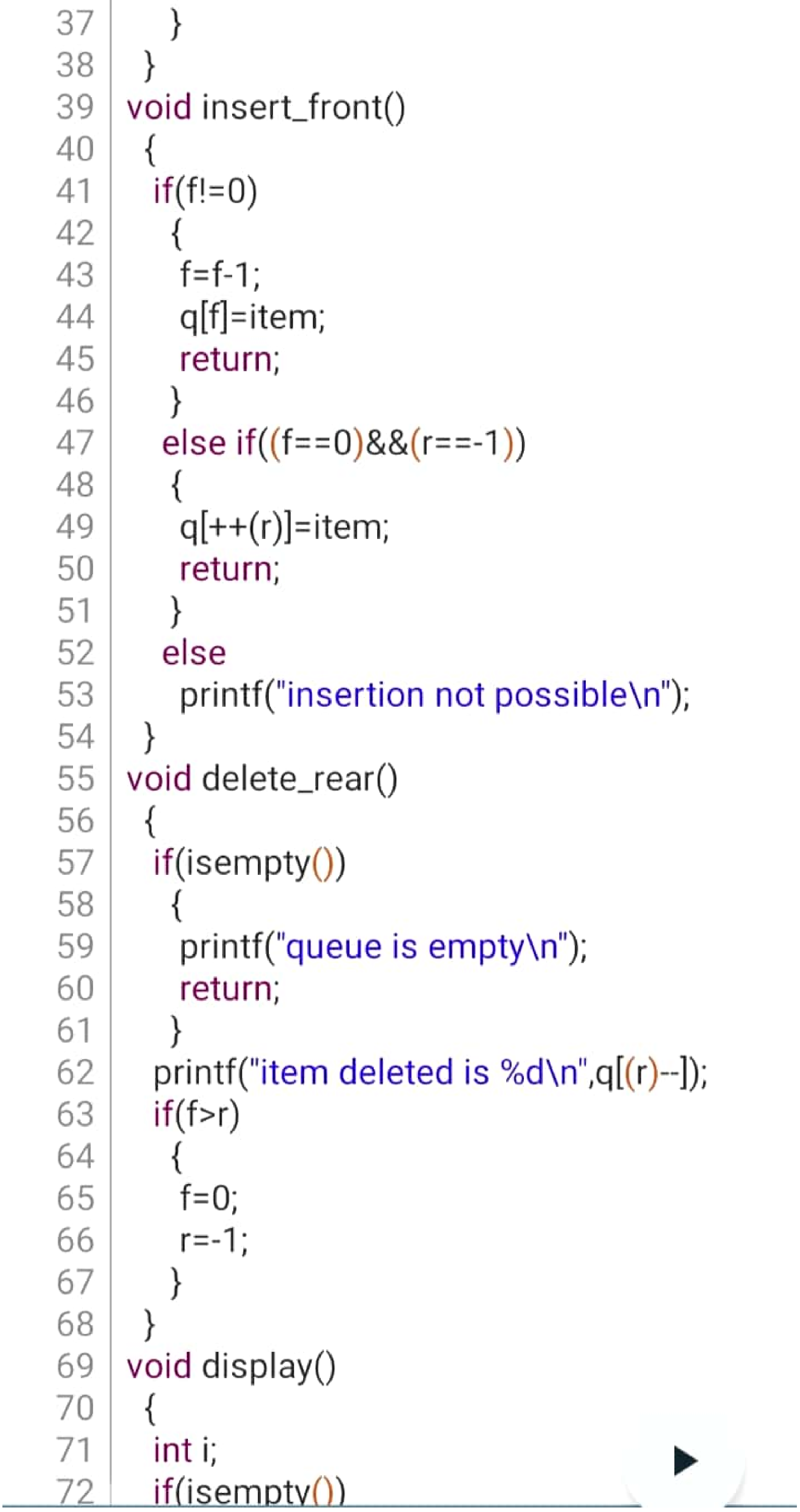
```
break;
~~case 4~~:
default : exit(0);
}
return 0;
}
```

Scanned with CamScanner



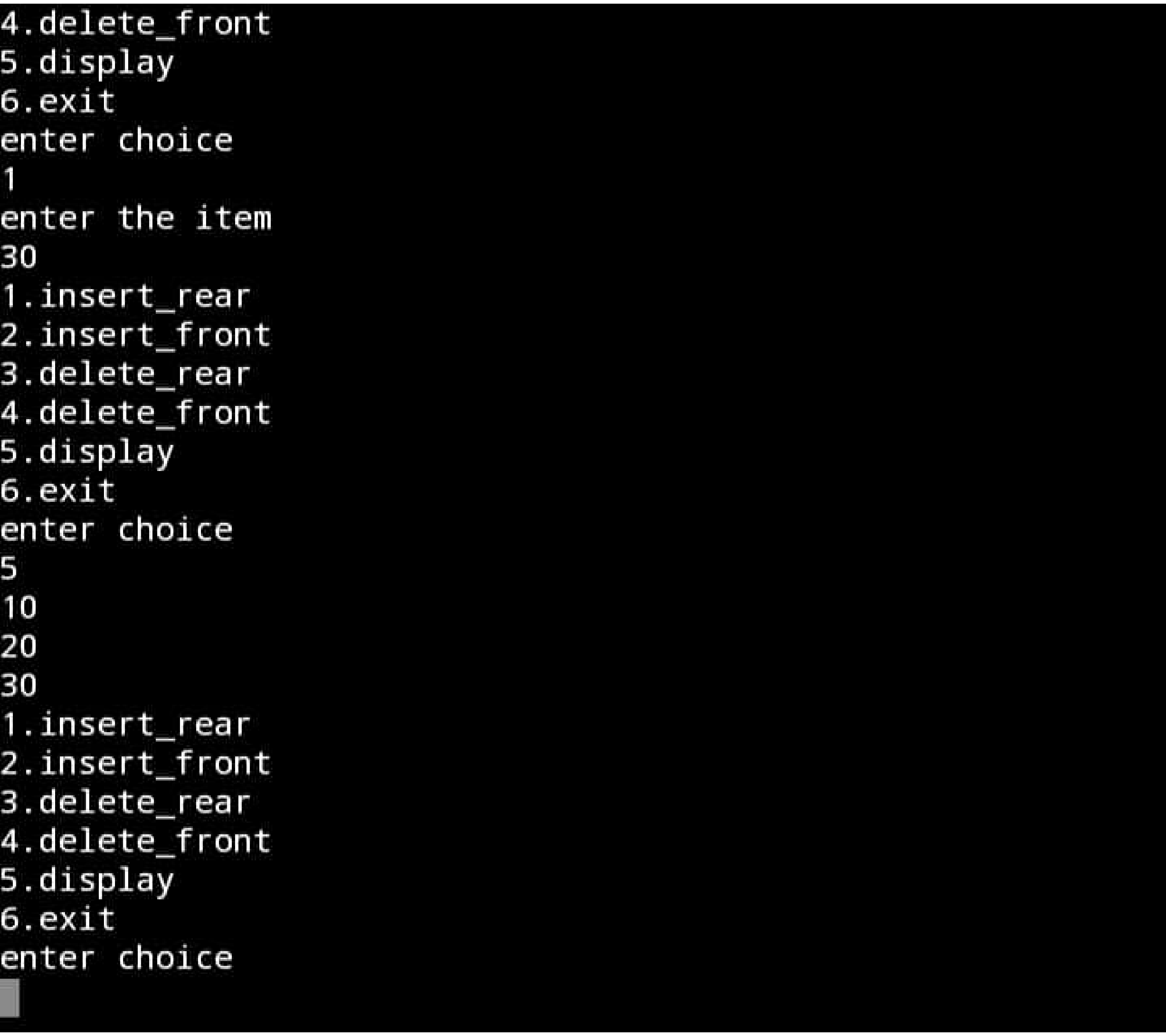
```c
#include<stdio.h>
#include<conio.h>
#define qsize 5
int f=0,r=-1,ch;
int item,q[10];

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;
```

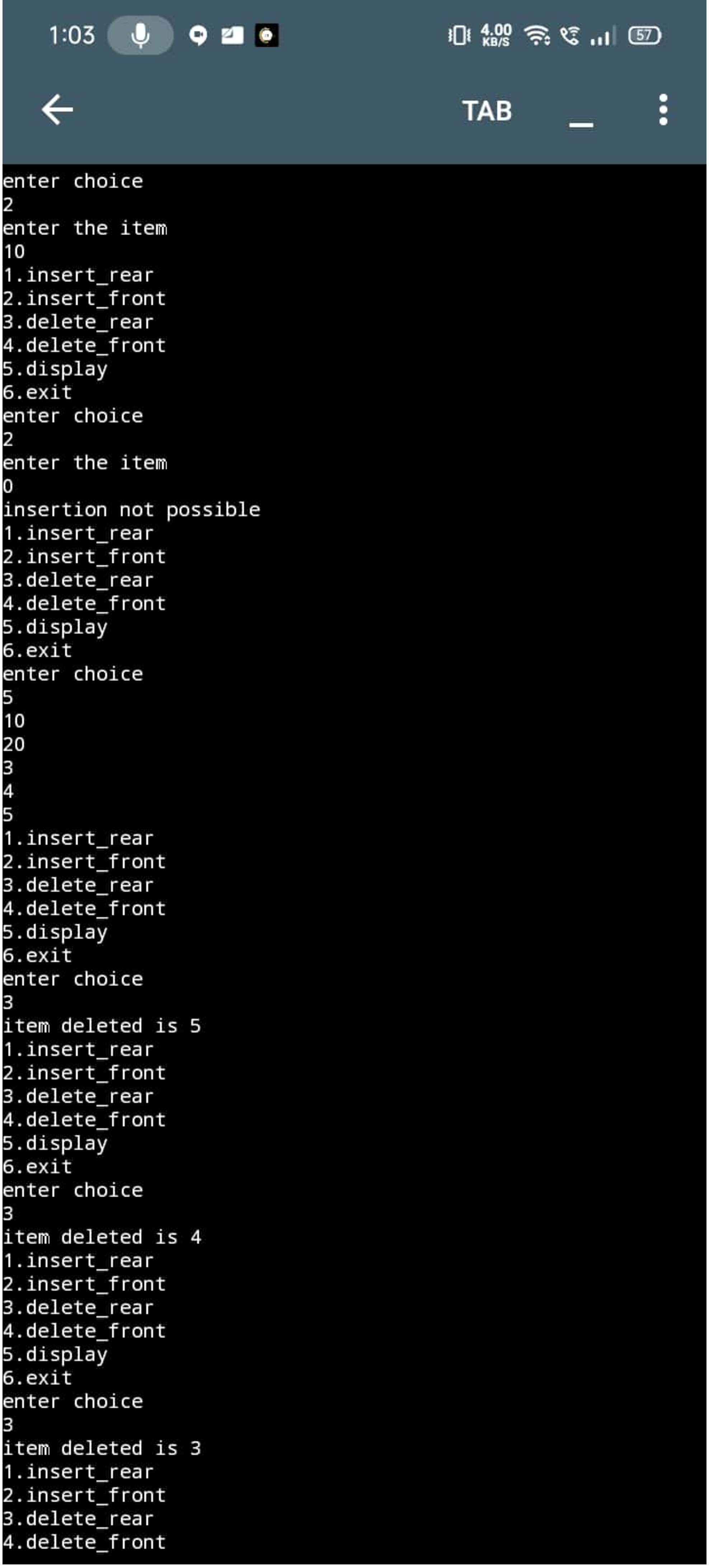


```c
    }
  }
void insert_front()
  {
   if(f!=0)
     {
      f=f-1;
      q[f]=item;
      return;
     }
    else if((f==0)&&(r==-1))
     {
      q[++(r)]=item;
      return;
     }
    else
      printf("insertion not possible\n");
  }
void delete_rear()
  {
   if(isempty())
     {
      printf("queue 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())
```

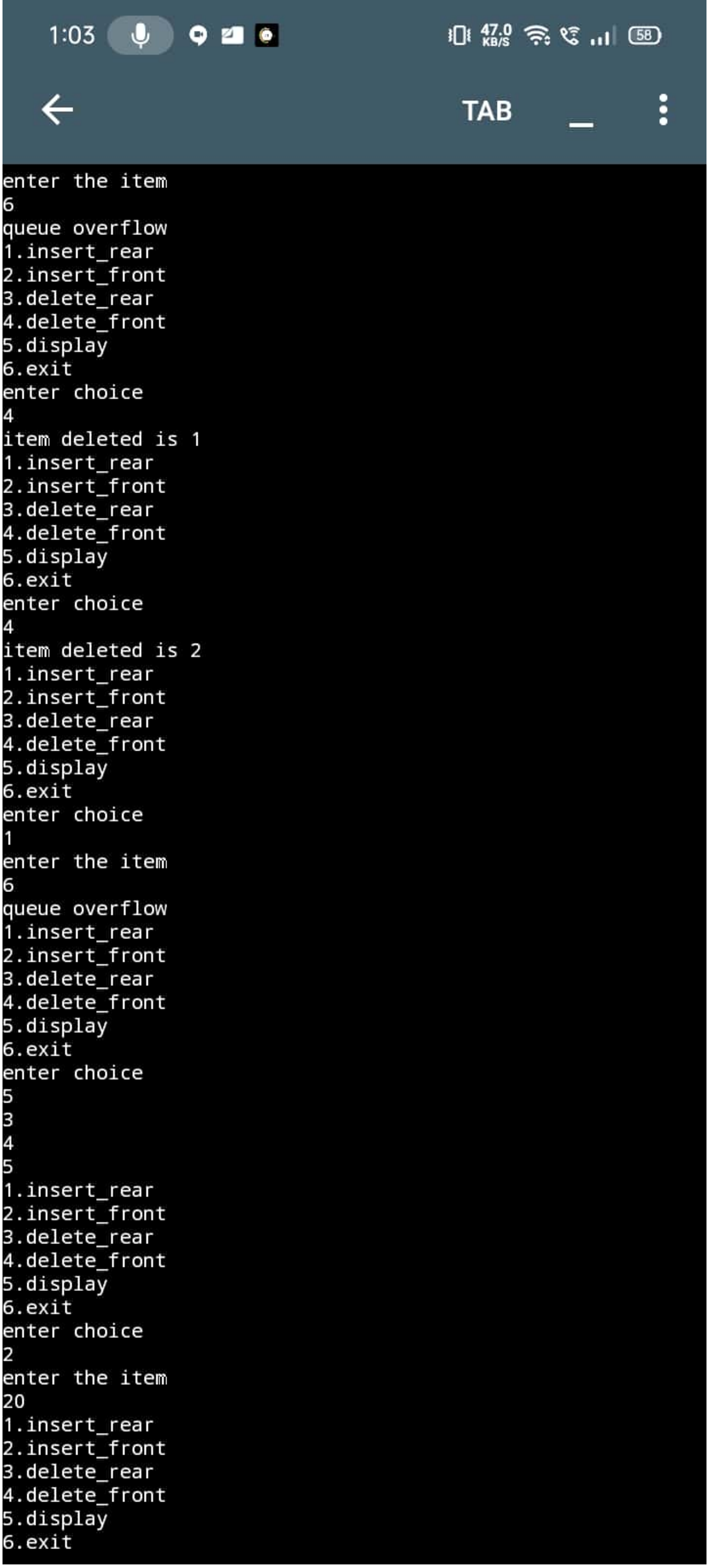
```
      {
       printf("queue empty\n");
       return;
      }
    for(i=f;i<=r;i++)
     printf("%d\n",q[i]);
  }
int main()
 {
  for(;;)
   {
    printf("1.insert_rear\n2.insert_front\n3.delete_rear\n4.delete_front\n5.display\n6.exit\n");
    printf("enter choice\n");
    scanf("%d",&ch);
    switch(ch)
     {
      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);
     }
   }
  return 0;
}
```



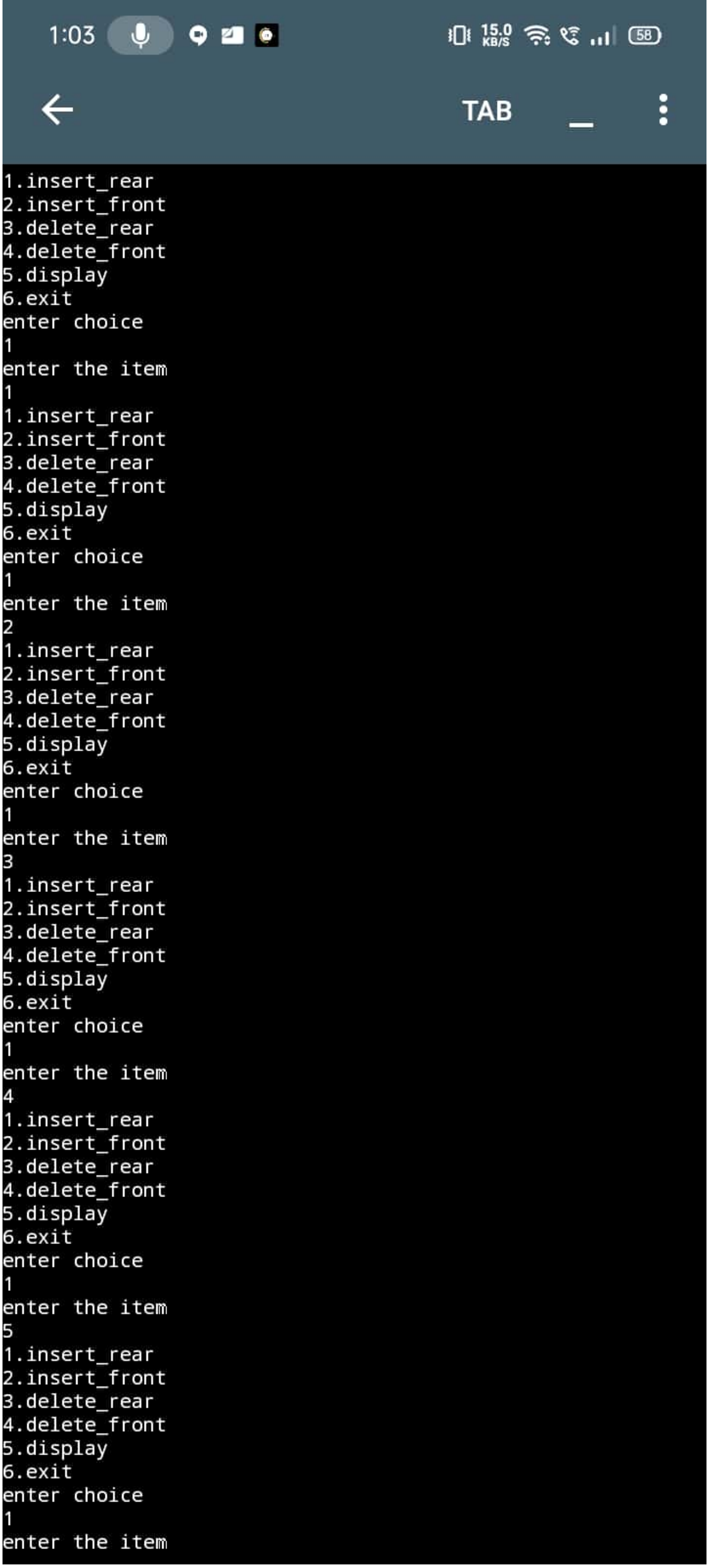
```
4.delete_front
5.display
6.exit
enter choice
1
enter the item
30
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
5
10
20
30
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
```



```
enter choice
2
enter the item
10
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
2
enter the item
0
insertion not possible
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
5
10
20
3
4
5
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
3
item deleted is 5
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
3
item deleted is 4
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
3
item deleted is 3
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
```



```
enter the item
6
queue overflow
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
4
item deleted is 1
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
4
item deleted is 2
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
6
queue overflow
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
5
3
4
5
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
2
enter the item
20
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
```



```
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
1
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
2
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
3
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
4
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
5
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
```

Double Ended Queue

```
#include<stdio.h>
#include< process.h>
# define qsize 5
int f=0, r=-1, ch, 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)
    {
     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++)
     printf ("%d \n", q[i]);
   }
 void main ()
 {
  for (;;)
   {
    printf ("1. insert_rear \n 2. insert_front \n 3.
             Delete Rear \n 4. Delete Front \n 5. Display \n
             6. Exit \n);
    printf ("Enter choice \n);
    scanf ("%d", &ch);
    switch (ch)
    {
     case 1 : printf ("enter the item \n");
              scanf ("%d", &item);
              insert_front ();
              break;
     case 2 : printf ("enter the item \n");
              scanf ("%d", &item);
              insert_front ();
              break;
     case 3 : delete_rear ();
              break;
case    case 4: delete_front ()
                break;
        case 5: display ()
                break;
          defaut : exit (0);
          }
          }
          return 0;
          }
```