8-1-24

LAB-3

Date
Page

1) Write a program to simulate the working of the queue of integers using an array. Provide the following operations. Insert, delete, display. The program should print appropriate messages.

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 5
int front = -1, rear = -1, int q[SIZE];
void insert();
void delete();
void display();
int main(){
  int choice;
  while (1) {
  printf ("1: Insert \n 2: Delete \n 3: Display \n 4: Exit");
  scanf ("%d", &choice);
  switch (choice) {
  case 1:
      insert();
      break;
  case 2:
      delete();
      break;
case 3:
      display();
      break;
case 4: exit(0);
default: printf ("wrong input \n");
}}}
```

```
printf("The queue is : \n");
for(i=front; i<=rear; i++) {
    printf("%d", q[i]); }
}
```

(d) Write a program to simulate the working of a circular queue using an array. Provide the following operations insert, delete & display. The program should print appropriate message for queue empty & queue overflow conditions.

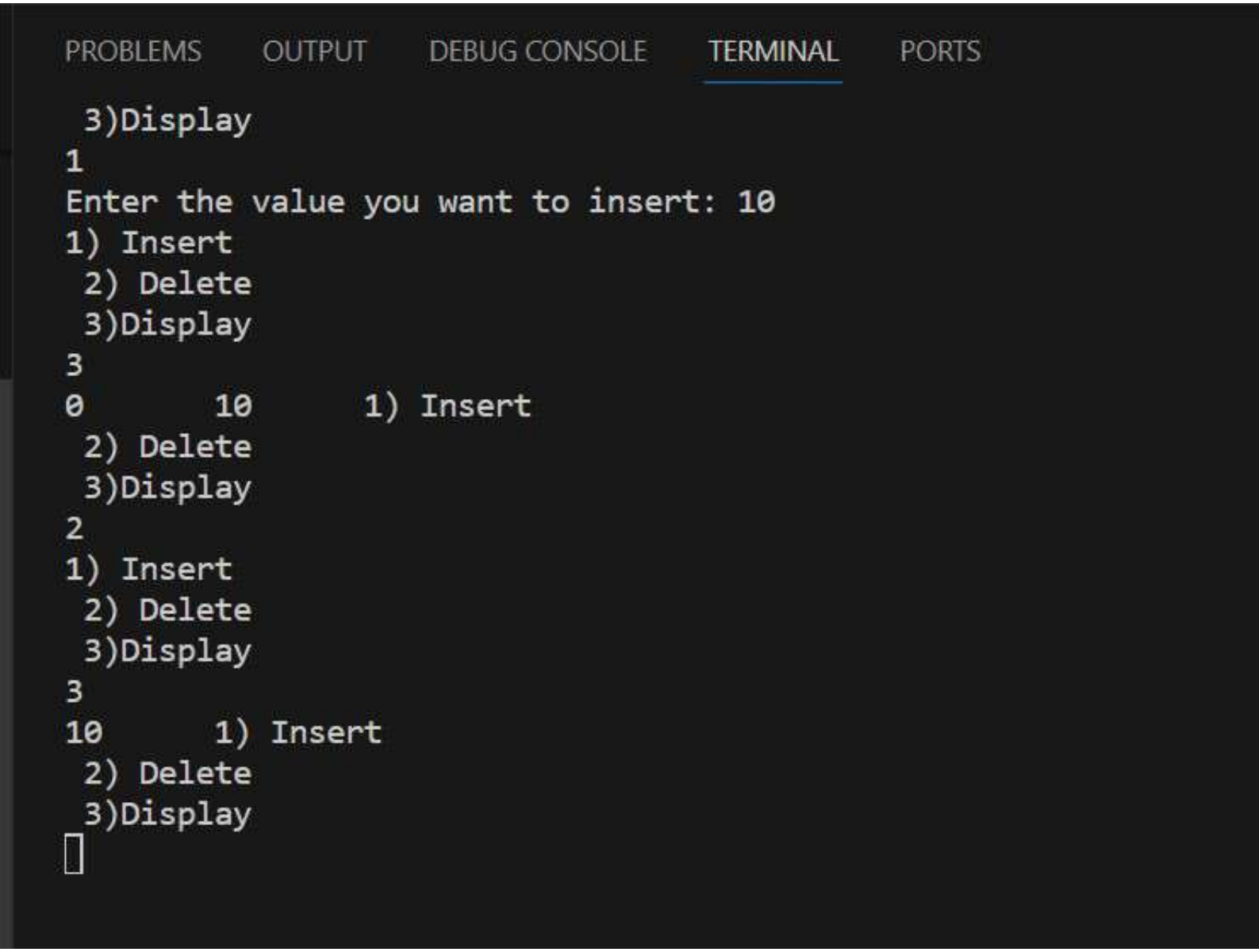
→
```
#include <stdio.h>
#define size 5
int items[size];
int front = -1, rear = -1;
int isFull() {
    if ((front == rear+1) || (front == 0 && rear == size-1))
        return 1;
    return 0;
}
int isEmpty()
{   if (front == -1)
        return 1;
    return 0;
}
void enqueue(int element) {
    if (isFull()) {
        printf("queue is full");
    }
    else {
        if (front == -1)
            front = 0;
```

```
void insert() {
  int add;
  if (rear == SIZE-1) {
     printf("Overflow condition");
  }
  else {
     if (front == -1){
        front = 0;
     printf("Enter a value to insert: ");
     scanf("%d", &add);
     rear ++;
     q[rear] = add;
  }
}

void delete() {
    if (front == rear || front > rear) {
       printf("Underflow condition");
       return;
    }
   else {
      printf("The deleted item will be %d",
            q[front]);
            front ++;
    }
}
void display()
{   int i;
    if (front == -1 || front > rear) {
       printf("The queue is empty");
    }
else {
```

```
void main() {
    int choice, element;
    while (1) {
    printf ("1: insert \n 2: delete \n 3: display \n 4: exit");
    printf ("%d", &choice);
    switch (choice){
        case 1: printf ("\n Enter the element to insert");
                scanf ("%d", &element);
                enqueue(element);
                break;
        case 2: element = dequeue();
                if (element != -1)
                    printf("%d is deleted item", element);
                break;
        case 3: display();
                break;
        case 4: exit(0);
        default : printf("Invalid choice");
    }
}
```

=>

```
    rear = (rear + 1) % size;
    items[rear] = element;
    printf("%d is inserted", element);
}
int deQueue()
{ int value;
    if (isEmpty())
    { printf("\n Queue is empty.\n");
      return -1;
    }
    else {
      value = items[front];
      if (front == rear) {
          front = -1; rear = -1;
      }
      else {
          front = (front + 1) % size;
      }
      return value;
}

→ void display()
{ int i;
    if (isEmpty())
      printf("\n Queue is empty");
    else {
      printf("Front position = %d\n", front);
      for (i = front; i <= rear; i = (i+1) % size) {
          printf("\t%d\t", items[i]); }
      printf("%d\t", items[i]);
    }
}
```


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
3)Display
1
Enter the value you want to insert: 10
1) Insert
2) Delete
3)Display
3
0 10 1) Insert
2) Delete
3)Display
2
1) Insert
2) Delete
3)Display
3
10 1) Insert
2) Delete
3)Display