Program 3b

Write A Program to simulate the working of a circular queue of integers using an array. Provide the following operations: Insert, Delete & Display

The program should print appropriate messages for queue empty and queue overflow conditions

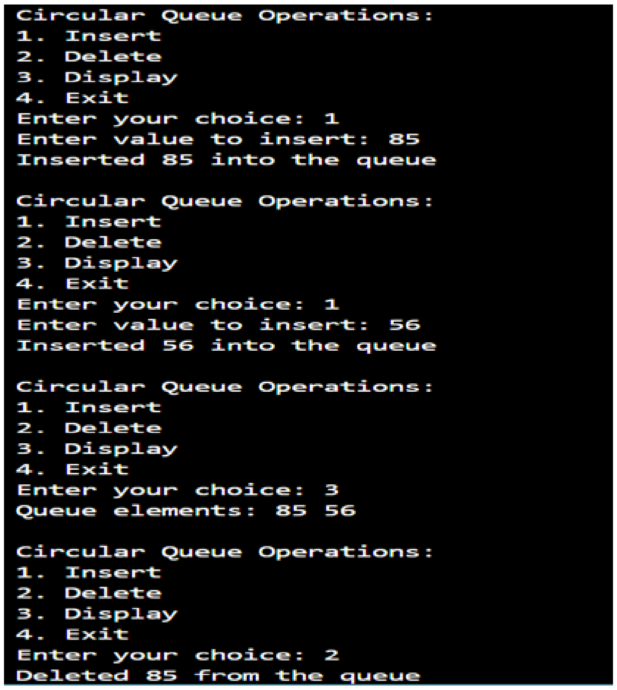
Code:

```
 #include <stdio.h>

#include <stdlib.h>

#define MAX 5



int queue[MAX];

int front = -1;

int rear = -1;



int isFull() {

        return (front == (rear + 1) % MAX);

}



int isEmpty() {

        return (front == -1);

}



void insert(int value) {

        if (isFull()) {
```

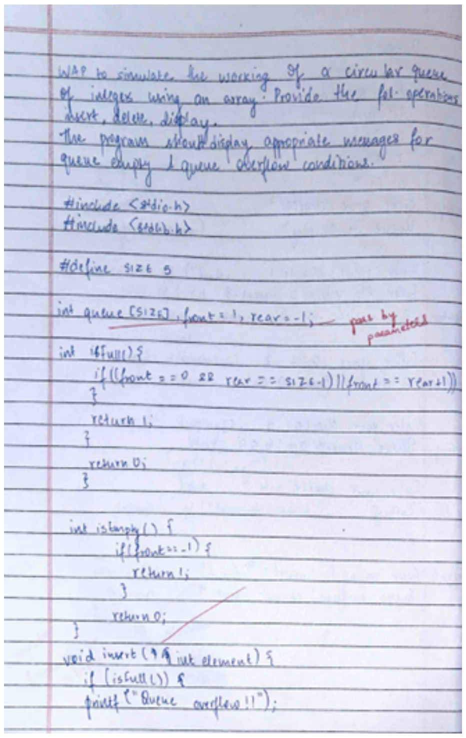
```
        printf("Queue Overflow: Unable to insert %d\n", value);

        return;

            }

            if (isEmpty()) {

        front = 0; // Set front to 0 if the queue is empty

            }

            rear = (rear + 1) % MAX;

    queue[rear] = value;

    printf("Inserted %d into the queue\n", value);

}


void delete() {

            if (isEmpty()) {

        printf("Queue Underflow: Unable to delete from the queue\n");

        return;

            }

            int deletedValue = queue[front];

            if (front == rear) {

        front = -1; // Queue becomes empty

        rear = -1;

            } else {

        front = (front + 1) % MAX;

            }
```

```
   printf("Deleted %d from the queue\n", deletedValue);
}
 void display() {
            if (isEmpty()) {
        printf("Queue is empty\n");
        return;
            }
   printf("Queue elements: ");
            int i = front;
            while (1) {
        printf("%d ", queue[i]);
            if (i == rear) break;
            i = (i + 1) % MAX;
            }
   printf("\n");
}
 int main() {
            int choice, value;

            while (1) {
        printf("\nCircular Queue Operations:\n");
        printf("1. Insert\n");
        printf("2. Delete\n");
printf("3. Display\n");

printf("4. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);


switch (choice) {

   case 1:

      printf("Enter value to insert: ");

      scanf("%d", &value);

      insert(value);

      break;

   case 2:

      delete();

      break;

   case 3:

      display();

      break;

   case 4:

      exit(0);

   default:

      printf("Invalid choice. Please try again.\n");

   }

   }
```

```
        return 0;

}
```

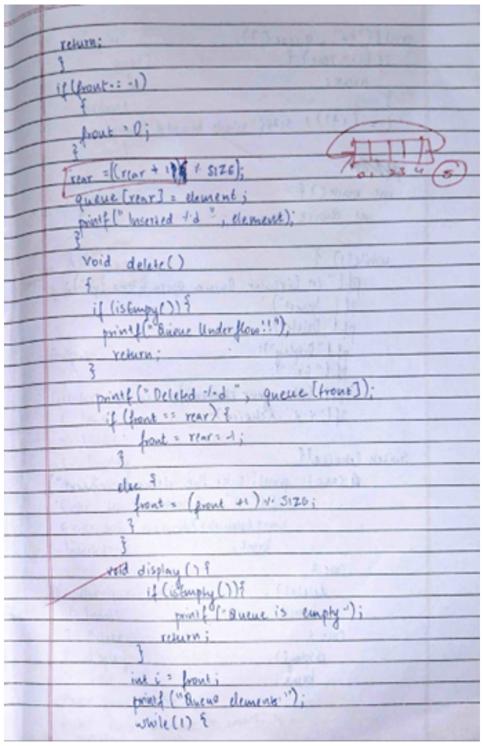


```
Circular Queue Operations:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 85
Inserted 85 into the queue

Circular Queue Operations:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 56
Inserted 56 into the queue

Circular Queue Operations:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Queue elements: 85 56

Circular Queue Operations:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 2
Deleted 85 from the queue
```

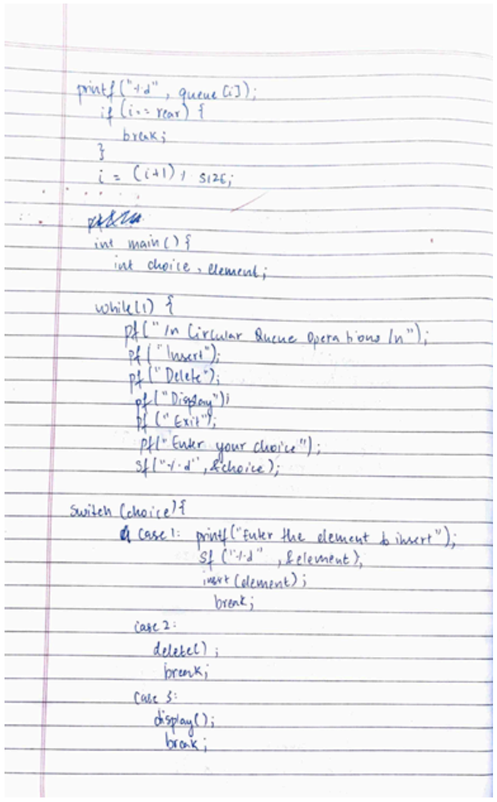


WAP to simulate the working of a circular queue of integers using an array. Provide the fol. operations insert, delete, display.
The program should display appropriate messages for queue empty & queue overflow conditions.

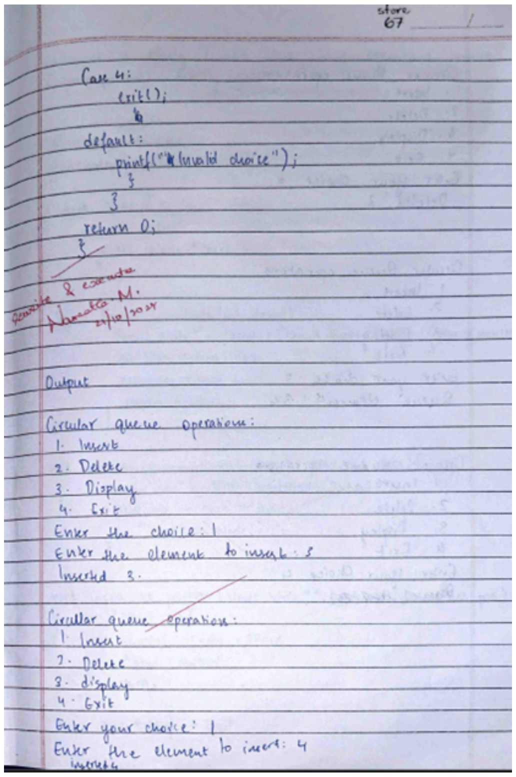
```
#include <stdio.h>
#include <stdlib.h>

#define SIZE 5

int queue [SIZE], front = 1, rear = -1;    pass by parameters

int isFull() {
    if((front == 0 && rear == SIZE-1)||(front == rear+1))
    }
    return 1;
   }
   return 0;
  }

 int isEmpty() {
      if(front==-1) {
        return 1;
      }
      return 0;
 }
void insert (int element) {
   if (isFull()) {
   printf ("Queue overflow !!");
```



```
return;
}
if(front == -1)
  {
  front = 0;
 }
rear = ((rear + 1) % SIZE);
 queue [rear] = element ;
 printf(" Inserted %d ", element);
 }
 Void delete()
  {
   if (isEmpty()){
    printf("Queue Underflow!!");
     return;
  }
   printf (" Deleted %d ", queue [front]);
    if (front == rear) {
        front = rear = -1;
      }
     else {
       front = (front +1) % SIZE;
      }
     }
    void display () {
         if (isEmpty()){
              printf("Queue is empty");
             return;
         }
        int i = front;
        printf ("Queue elements ");
        while(1) {
```



```
printf("%d", queue[i]);
   if (i == rear) {
      break;
   }
   i = (i+1) % size;

[illegible]
 int main() {
    int choice, element;

 while(1) {
      pf("\n Circular Queue Operations \n");
      pf("Insert");
      pf("Delete");
      pf("Display");
      pf("Exit");
       pf("Enter your choice");
      sf("%d", &choice);

switch (choice){
      case 1:  printf("Enter the element to insert");
                sf("%d" , &element);
                insert(element);
                  break;
      case 2:
         delete();
           break;
      case 3:
         display();
           break;
```



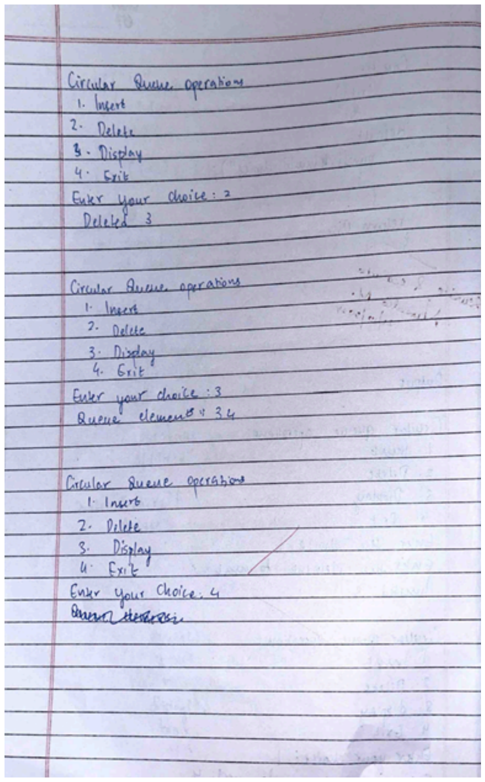
store
67

```
Case 4:
      exit();

default:
      printf("Invalid choice");
         }
      }
return 0;
}
```

Rewrite & execute
[illegible]
21/10/2024

Output

```
Circular queue operations:
 1. Insert
 2. Delete
 3. Display
 4. Exit
 Enter the choice: 1
 Enter the element to insert: 3
 Inserted 3.

Circular queue operation:
 1. Insert
 2. Delete
 3. display
 4. Exit
 Enter your choice: 1
 Enter the element to insert: 4
   inserted 4
```



```
Circular Queue operations
 1. Insert
 2. Delete
 3. Display
 4. Exit
 Enter your choice: 2
  Deleted 3


Circular Queue operations
  1. Insert
  2. Delete
  3. Display
   4. Exit
 Enter your choice: 3
 Queue elements: 3 4


Circular Queue operations
  1. Insert
 2. Delete
 3. Display
 4. Exit
Enter your Choice: 4
[illegible]
```