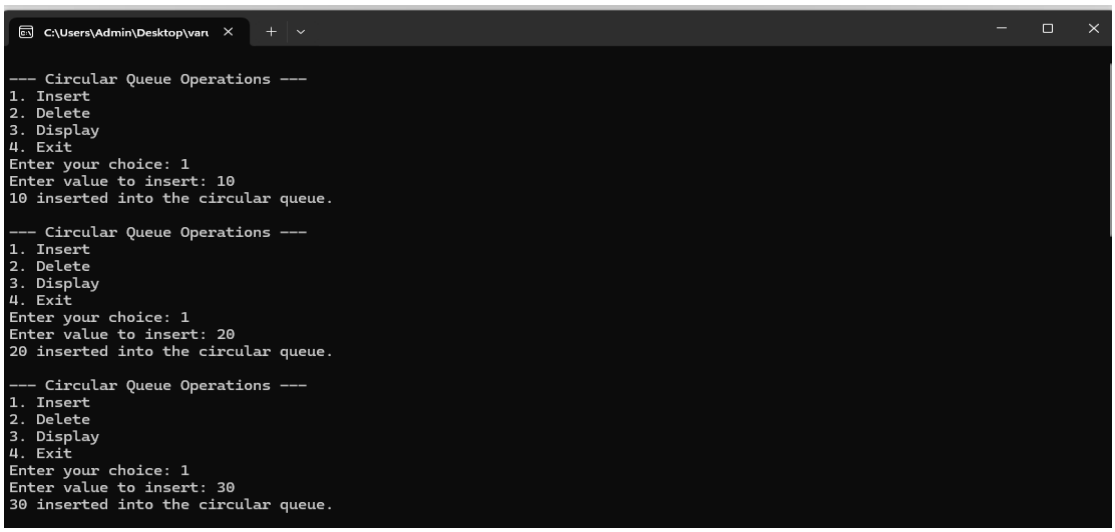
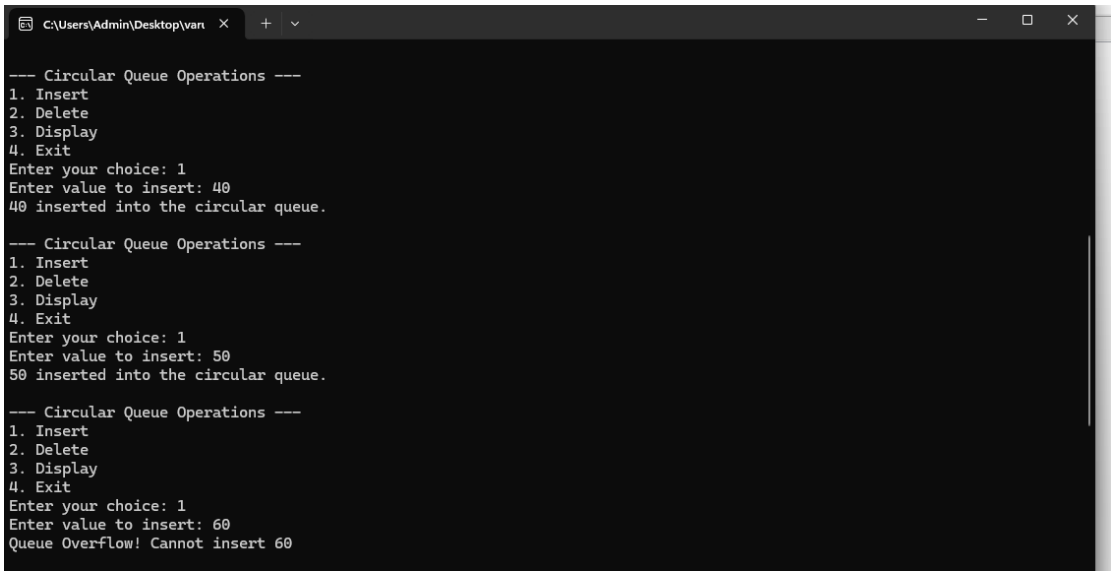
3. b) WAP to simulate the working of a circular queue of integers using an array. Provide the following operations: Insert, Delete & Display.
The program should print appropriate messages for queue empty and queue overflow conditions.

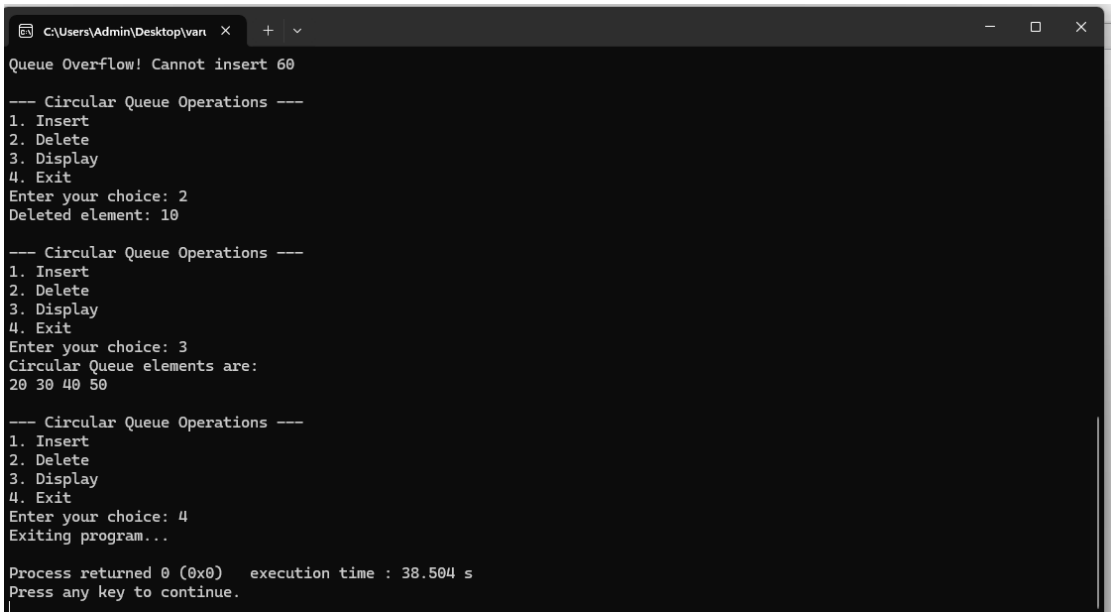
```
#include <stdio.h>
#define MAX 5   // Maximum size of the circular queue

int queue[MAX];
int front = -1, rear = -1;

// Function to insert an element into circular queue
void insert(int value) {
   // Check for overflow condition
   if ((front == 0 && rear == MAX - 1) || (rear + 1) % MAX == front) {
      printf("Queue Overflow! Cannot insert %d\n", value);
   }
   else {
      if (front == -1)  // First element insertion
         front = 0;
      rear = (rear + 1) % MAX;
      queue[rear] = value;
      printf("%d inserted into the circular queue.\n", value);
   }
}

// Function to delete an element from circular queue
void delete() {
   if (front == -1) {
      printf("Queue Underflow! Queue is empty.\n");
   }
   else {
      printf("Deleted element: %d\n", queue[front]);
      if (front == rear) {
         // Queue becomes empty after deletion
         front = rear = -1;
      }
      else {
         front = (front + 1) % MAX;
      }
   }
}

// Function to display all elements of circular queue
void display() {
   if (front == -1) {
      printf("Queue is empty.\n");
```

```
    }
    else {
        printf("Circular Queue elements are:\n");
        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("\n--- Circular Queue Operations ---\n");
        printf("1. Insert\n2. Delete\n3. Display\n4. 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:
                printf("Exiting program...\n");
                return 0;

            default:
                printf("Invalid choice! Try again.\n");
        }
    }
}
```

Output:


C:\Users\Admin\Desktop\var
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 10
10 inserted into the circular queue.
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 20
20 inserted into the circular queue.
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 30
30 inserted into the circular queue.


C:\Users\Admin\Desktop\var
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 40
40 inserted into the circular queue.
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 50
50 inserted into the circular queue.
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter value to insert: 60
Queue Overflow! Cannot insert 60


C:\Users\Admin\Desktop\var
Queue Overflow! Cannot insert 60
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 2
Deleted element: 10
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Circular Queue elements are:
20 30 40 50
--- Circular Queue Operations ---
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 4
Exiting program...
Process returned 0 (0x0) execution time : 38.504 s
Press any key to continue.