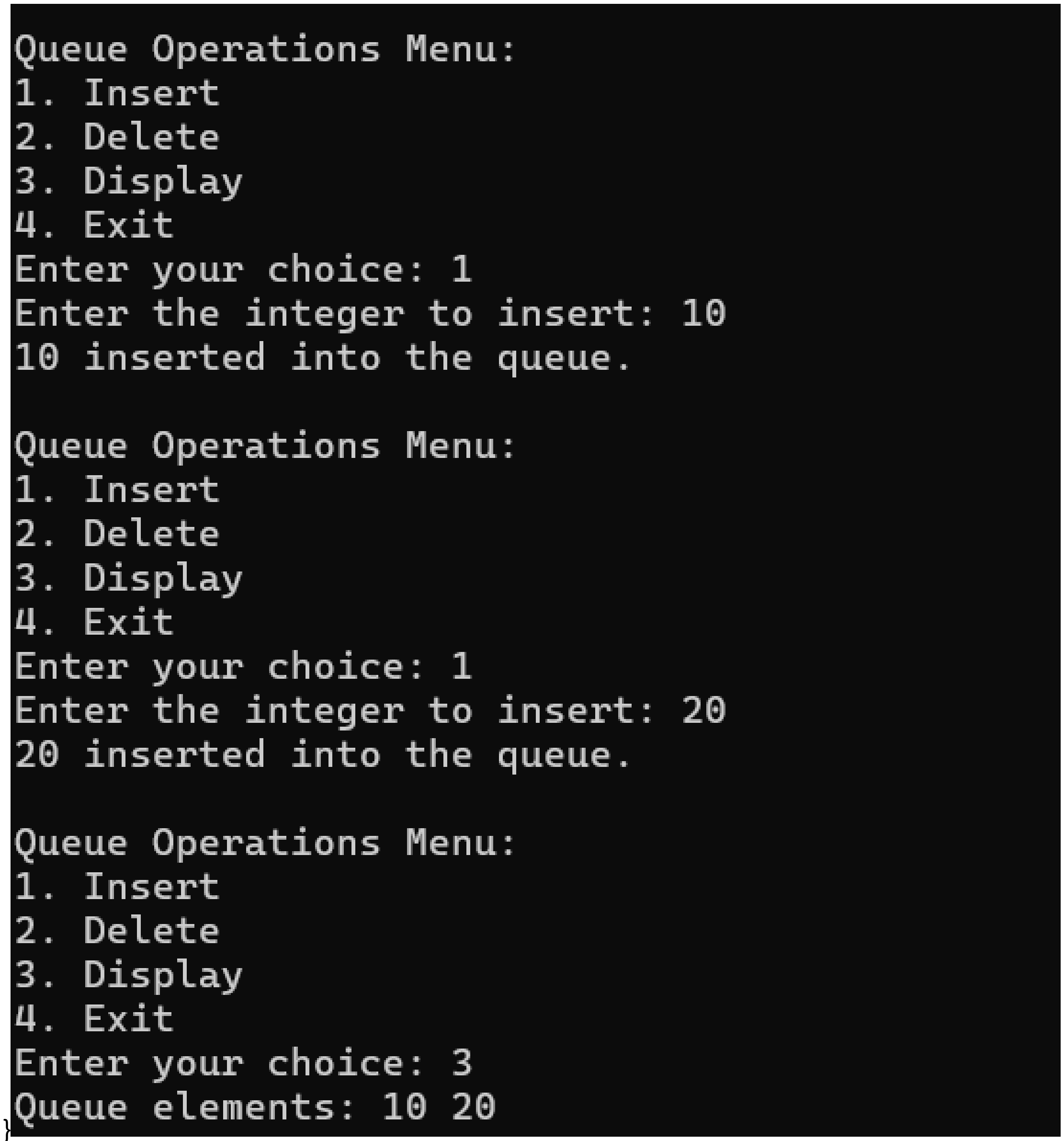
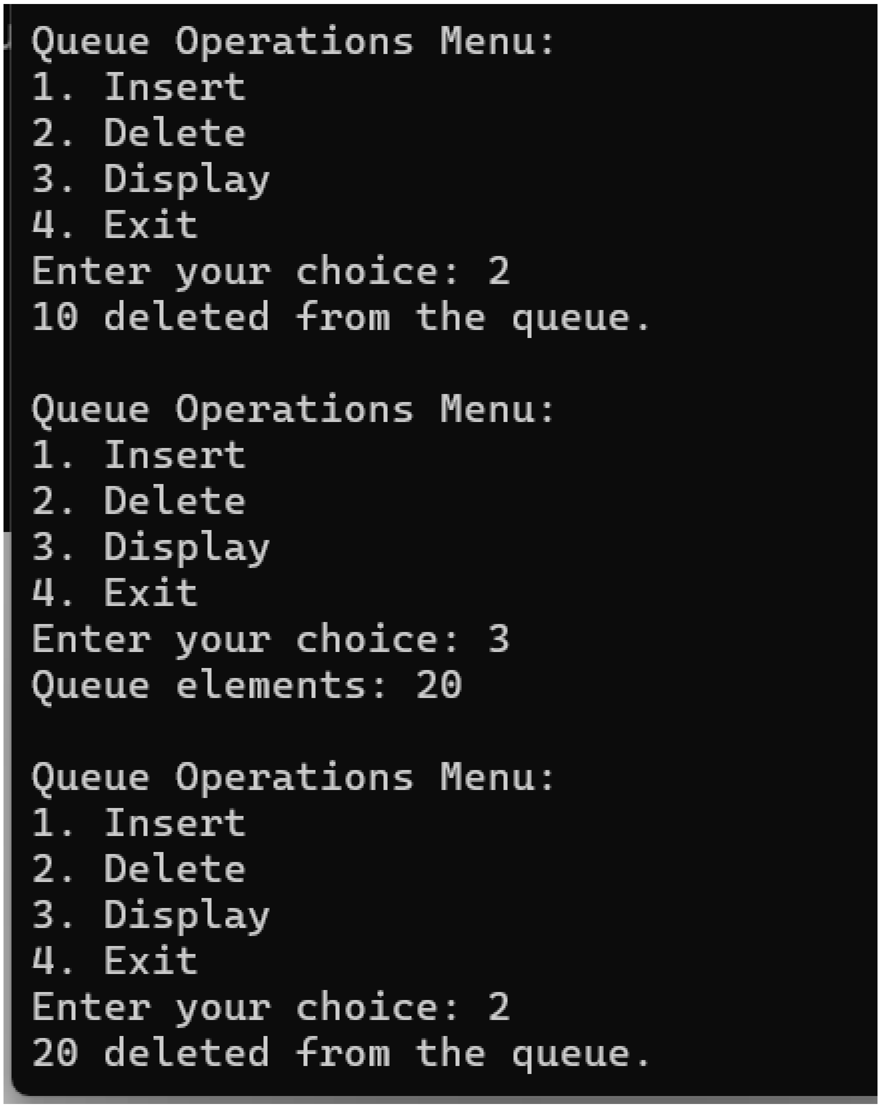
Q_3(a) _WAP to simulate the working of a queue of integers using an array . provide the following operations: Insert,delete,display The program should print appropriate message for queue empty and queue overflow conditionns

```c
#include <stdio.h>
#define MAX 5

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


void insert(int value) {
    if (rear == MAX - 1) {
        printf("Queue Overflow! Cannot insert %d\n", value);
        return;
    }
    if (front == -1) {
        front = 0;
    }
    rear++;
    queue[rear] = value;
    printf("%d inserted into the queue.\n", value);
}


void delete() {
    if (front == -1 || front > rear) {
        printf("Queue Underflow! Queue is empty.\n");
        return;
    }

    int deletedValue = queue[front];

    front++;

    printf("%d deleted from the queue.\n", deletedValue);

    if (front > rear) { // Reset queue if it becomes empty

        front = rear = -1;

    }

}


void display() {

    if (front == -1 || front > rear) {

        printf("Queue is empty.\n");

        return;

    }

    printf("Queue elements: ");

    for (int i = front; i <= rear; i++) {

        printf("%d ", queue[i]);

    }

    printf("\n");

}


int main() {

    int choice, value;

    do {

        printf("\nQueue Operations Menu:\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 the integer to insert: ");
            scanf("%d", &value);
            insert(value);
            break;
        case 2:
            delete();
            break;
        case 3:
            display();
            break;
        case 4:
            printf("Exiting program.\n");
            break;
        default:
            printf("Invalid choice! Try again.\n");
    }
} while (choice != 4);

return 0;
```

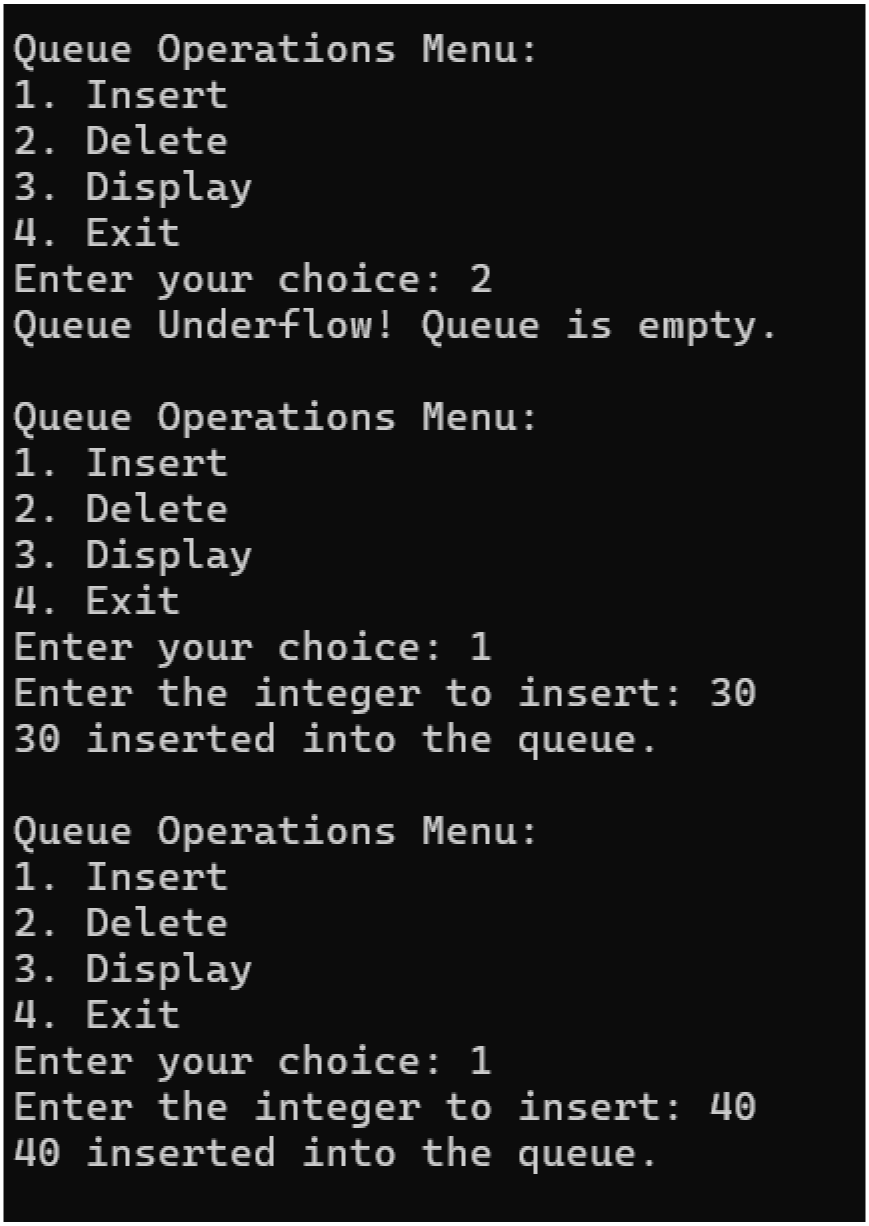


```
Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter the integer to insert: 10
10 inserted into the queue.

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter the integer to insert: 20
20 inserted into the queue.

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Queue elements: 10 20
```

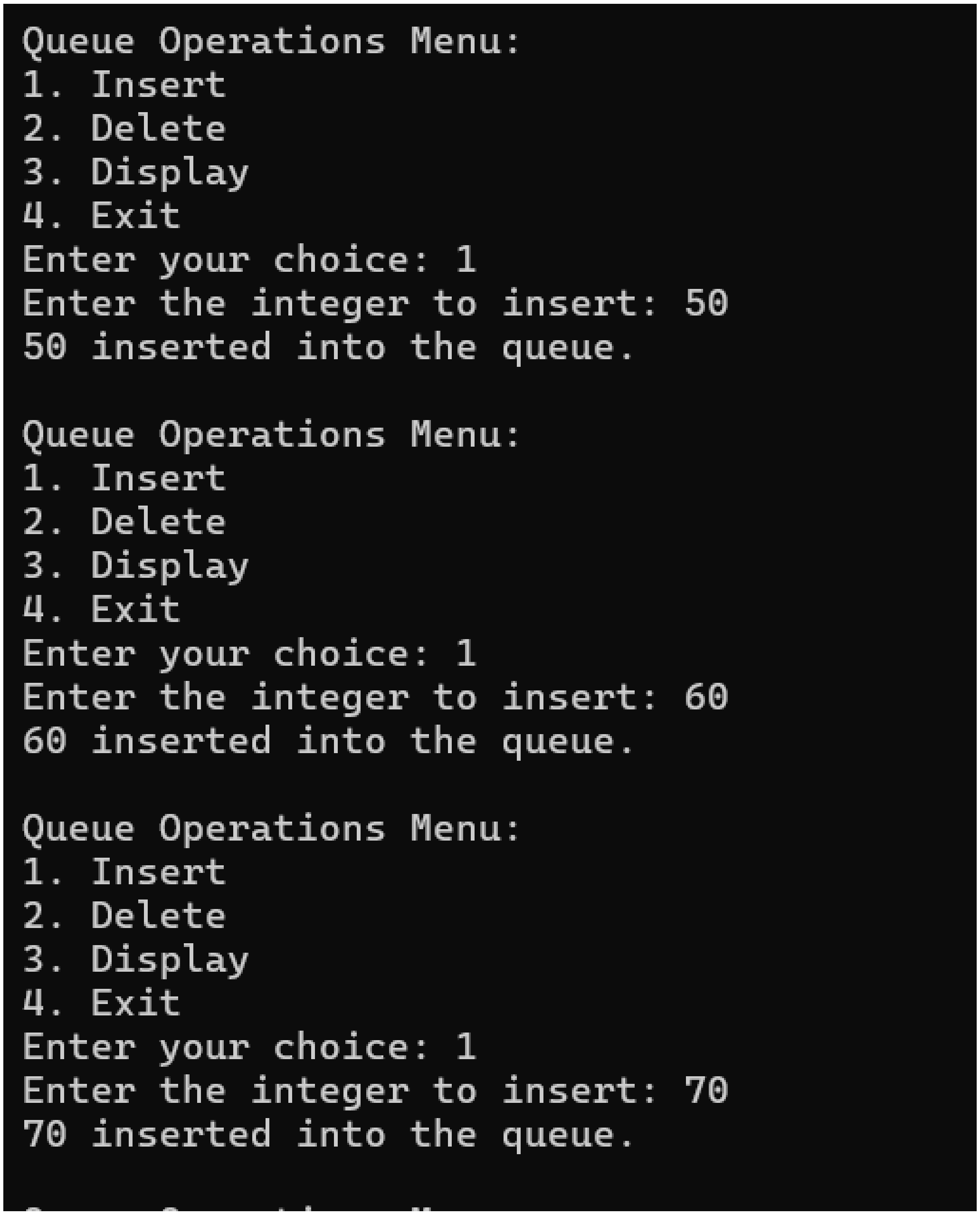
}



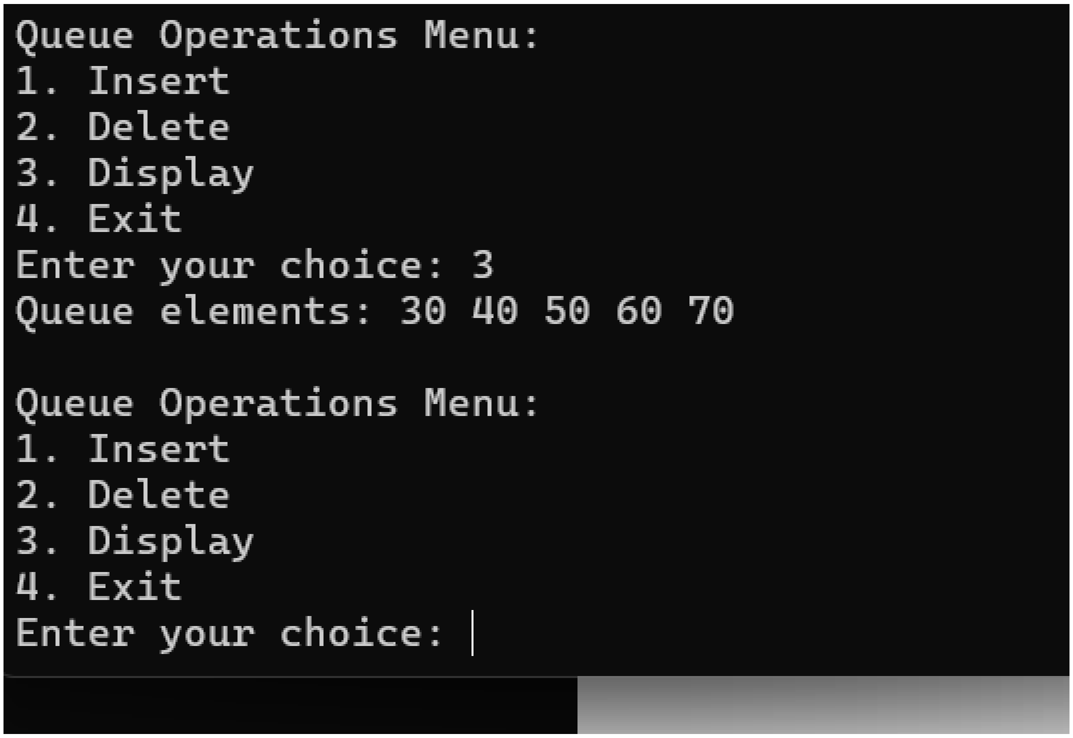
```
Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 2
10 deleted from the queue.

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Queue elements: 20

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 2
20 deleted from the queue.
```



```
Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 2
Queue Underflow! Queue is empty.

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter the integer to insert: 30
30 inserted into the queue.

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter the integer to insert: 40
40 inserted into the queue.
```



```
Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter the integer to insert: 50
50 inserted into the queue.

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter the integer to insert: 60
60 inserted into the queue.

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 1
Enter the integer to insert: 70
70 inserted into the queue.

Queue Operations Menu
```



```
Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice: 3
Queue elements: 30 40 50 60 70

Queue Operations Menu:
1. Insert
2. Delete
3. Display
4. Exit
Enter your choice:
```