

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 messages for queue empty and queue overflow conditions

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

```
#define MAX 5 //maximum size of the queue
```

```
int queue[MAX];
```

```
int front = -1, rear = -1;
```

```
//function to insert an element in the queue : (ENQUEUE)
```

```
void insert(int value)
```

```
{
```

```
    if(rear == MAX -1)
```

```
    {
```

```
        printf("Queue Overflow! Cannot insert %d\n",value);
```

```
    }
```

```
    else{
```

```
        if(front == -1)
```

```
        {
```

```
            front = 0; //first insertion
```

```
        }
```

```
        rear = rear +1;
```

```
        queue[rear] = value;
```

```
        printf("%d inserted into the queue.\n",value);
```

```
    }
```

```
}
```

```
//function to delete an element from the queue :(DEQUEUE)
```

```
void delete()
```

```

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

    }
    else{
        printf("Deleted element : %d\n",queue[front]);
        front++;
    }
}

```

//FUNCTION TO DISPLAY THE ELEMENTS OF THE QUEUE

```

void display()
{
    if(front == -1 || front > rear)
    {
        printf("Queue is empty.\n");
    }
    else
    {
        printf("Queue elements:\n");
        for(int i = front;i<= rear;i++)
        {
            printf("%d\n",queue[i]);
        }
    }
}

```

//FUNCTION TO GET FRONT ELEMENT:

```

void displayFront()

```

```

{
    printf("The Front element is %d.\n",queue[front]);
}
void displayRear()
{
    printf("The Rear element is %d.\n",queue[rear]);
}
int main()
{
    int choice ,value;
    while(1)
    {
        printf("\nQueue Operations:\n1.Insert\n2.Delete\n3.Display\n4.Exit\n5.Display Front\n6.Display Rear\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 the program!!!");
        return 0;
    case 5:
        displayFront();
        break;
    case 6:
        displayRear();
        break;

    default:
        printf("Invalid choice! Please try again.\n");
    }
}
return 0;
}
```

OUTPUT :

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 1
Enter value to insert: 10
10 inserted into the queue.
```

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 1
Enter value to insert: 20
20 inserted into the queue.
```

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 1
Enter value to insert: 30
30 inserted into the queue.
```

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 3
Queue elements:
10
20
30
```

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 5
The Front element is 10.
```

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 6
The Rear element is 30.
```

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 2
Deleted element : 10
```

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 2
Deleted element : 20
```

```
Queue Operations:
1.Insert
2.Delete
3.Display
4.Exit
5.Display Front
6.Display Rear
Enter your choice: 
```