CIRCULAR QUEUE

Q3. WAP to simulate the working of a circular queue of integers using an array.

Provide the following operations.

a) Insert

b) Delete

c) Display

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

```
#include <stdio.h>
int rear = -1;
int front = -1;
int max = 5;

void Enqueue(int arr[], int value) {
   if (rear == -1 || front == -1) {
      rear++;
      front++;
      arr[rear] = value;
      rear++;
   }
   else if (rear == max) {
      if(front!=0){
         rear=0;
         arr[rear]=value;
         rear++;
      }
      else{
      printf("Overflow\n");
      }
   }
```

```
    else if(rear==(front)){
            printf("overflow");
            }
    else {
       arr[rear] = value;
       rear++;
    }
}

void Dequeue(int arr[]) {
    if (front == -1 || rear == -1) {
       printf("Underflow\n");
    } else if (front == (rear - 1)) {
       printf("Deleted element = %d\n", arr[front]);
       rear = -1;
       front = -1;
    } else {
       int temp = arr[front];
       front++;
       printf("Deleted element = %d\n", temp);
    }
}

void display(int arr[]) {
    for (int i=0; i <max; i++) {
       printf("%d\t", arr[i]);
    }
    printf("\n");
}

int main() {
```

```
    int choice;
    int arr[5];
    int value;

    void operations() {
        printf("Enter appropriate number to perform operations: \n1. Enqueue \n2. Dequeue \n3. Display \n4. Exit\n");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                printf("Enter the value to insert\n");
                scanf("%d", &value);
                Enqueue(arr, value);
                operations();
                break;
            case 2:
                Dequeue(arr);
                operations();
                break;
            case 3:
                display(arr);
                operations();
                break;
            case 4:
                printf("Exited\n");
                break;
            default:
                printf("Invalid choice\n");
                operations();
                break;
        }
```

```
    }


    operations();
    return 0;
}
```

OUTPUT: