🧑‍💻

Circular Queue

```
#include <iostream>
using namespace std;
#include<stdlib.h>

class queue {
    int cq[5], front = -1, rear = -1;
    public:
    void qInsert();
    void qDelete();
    void qDisplay();
};

void queue::qInsert() {
    int num;
    if ((rear + 1) % 5 == front || (front == 0 && rear == 4))
    cout<<"Circular Queue Overflow\n";
    else {
        cout<<"Enter the number to be inserted: ";
        cin>>num;
        if (front == -1) {
            front = 0;
        }
        rear = (rear + 1) % 5;
        cq[rear] = num;
    }
}

void queue::qDelete() {
    int num;
    if (front == -1)
    cout<<"Circular Queue Underflow\n";
    else {
        num = cq[front];
        if (front == rear)
        front = rear = -1;
        else {
            front = (front + 1) % 5;
        }
        cout<<"The deleted element is: "<<num<<endl;
    }
}
```

```
void queue::qDisplay() {
    int i;
    if (front == -1)
    cout<<"Circular Queue is empty\n";
    else {
        i = front;
        while (i != rear) {
            cout<<cq[i]<<" ";
            i = (i + 1) % 5;
        }
        cout<<cq[i]<<endl;
    }
}

int main() {
    queue Q;
    int ch;
    while (1) {
        cout<<"*****MENU*****\n1. Insert\n2. Delete\n3. Display\n4. Exit\nEnter your choice: ";
        cin>>ch;
        switch (ch) {
            case 1: Q.qInsert();
                    break;
            case 2: Q.qDelete();
                    break;
            case 3: Q.qDisplay();
                    break;
            case 4: exit(0);
            default: cout<<"Invalid choice\n";
        }
    }
}
```