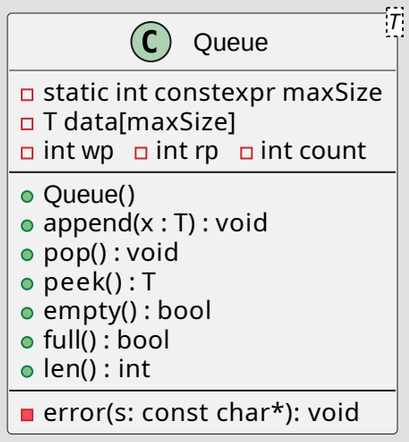
Übung zu Objektorientierter Programmierung

① Es sei die folgende Klassenkarte einer Warteschlangenklasse nach FIFO-Prinzip zu betrachten:



In der modernen Softwareentwicklung werden Tools eingesetzt, die aus UML-Klassenkarten C++-Klassen mit vordeklarierten Attributen und Methodenrümpfen erzeugen. Ein solcher Klassenrumpf sei für die FIFO-Warteschlage gem. Vorgaben zu vervollständigen:

```
template <typename T> class Queue {
private:
    static constexpr int maxSize = MAXSIZE;
    T data[maxSize];
    int wp;     // Write Pointer (WP)
    int rp;     // Read Pointer  (RP)
    int count;  // Elementcount

    void error(const char* s) {
        std::cerr << s << std::endl;
        exit(1);
    }

public:
    Queue() : wp(0), rp(0), count(0) {}

    void append(T x) {
        if (full()) { error("Queue is full!"); }
        data[wp] = x;
        wp = (wp + 1) % maxSize;
        count++;
    }

    void pop() {
        if (empty()) { error("Queue is empty!"); }
        rp = (rp + 1) % maxSize;
        count--;
    }

    T peek() {
        if (empty()) { error("Queue is empty!"); }
        return data[rp];
    }

    bool empty() { return (count == 0); }

    bool full() { return (count == maxSize); }

    int len() const { return count; }

    friend std::ostream& operator<<(std::ostream& os, const Queue<T>& q) {

        int idx = 0;

        for (int i = 0; i < q.count; ++i) {
            idx = (q.rp + i) % maxSize;
            os << q.data[idx];
            if (i < q.count - 1) { os << ", " };
        }

        return os;
    }
};
```