C++ 算法讲义：队列

1. 引言

- **问题引入**：
  例 15-7：超市排队。这回小止作为一个收银员在超市打工。收银员会给排在队伍最前面的顾客结账，然后再服务队伍中的下一个顾客。同时，队伍的末尾一直会有新的顾客加入。
  设问：如果我们在超市排队，收银员像处理栈一样，总是先给**最后面**来的人结账（后进先出），会发生什么？
  前面排队的人绝对会掀桌子！这太不公平了！
- **价值阐述**：
  为了维持秩序，我们需要一种公平的规则——**谁先排队，谁先结账。**这就是**队列（Queue）**的核心思想。在计算机世界里，操作系统的任务调度、网络数据的传输等，都需要这种公平的机制来维持运转。
- **学习目标**：
  掌握队列先进先出的运作机制，学会用数组模拟队列，并熟练使用 C++ STL 中的 `queue` 模板解决环形逻辑问题。

---

2. 核心思想

队列（queue）是一种遵循**先进先出（First In, First Out，简称 FIFO）**规则的线性数据结构。顾名思义，它模拟了现实中的排队现象。

我们将队列的核心操作与超市排队完美对应：

- **入队 `push(x)` —— 排队入列**
- **比喻**：新顾客 x 推着购物车，站到了队伍的最末尾。
- **意义**：将元素 x 压入队列的尾部。
- **看队首 `front()` —— 抬头看一眼**
- **比喻**：收银员小止喊：“下一位！”然后抬头看一眼现在站在最前面的人是谁。注意，**只是看一眼**，这个人还没结账走掉。
- **意义**：查询（获取）队首的元素，但不删除它。
- **出队 `pop()` —— 结账走人**
- **比喻**：最前面的人结完账，离开了队伍。排在第二位的人顺理成章地成为了新的队首。
- **意义**：弹出队首的元素（删除最前面的数据）。

---

3. 代码实现

方法一：使用数组模拟队列

在程序里，我们可以用一个固定大小的数组和两个变量来管理队伍。

- **初始化**：
  定义数组 `int q[100];`。
  初始化两个指针索引：`front = 0` 和 `rear = 0`。
- `front`：指向队列的第一个人（队首）。

- `rear`：指向队伍末尾**下一个可插入的空位。**
- **入队（`push`）**：
  将新元素放入 `rear` 指向的位置：`q[rear] = x;`。
  队伍变长了，尾巴往后挪一格：`rear++;`。
- **出队（`pop`）**：
  队首的人走了，前面空出来了，头指针往后挪一格：`front++;`。（我们不需要真的抹除数组里的数据，只要 `front` 越过它，它就相当于出队了）。

方法二：使用 C++ 现成的模板 —— `queue`

实际编程中，我们不需要每次都手写数组，C++ 标准库已经为我们准备好了。

```
#include <iostream>
#include <queue> // 队列头文件
using namespace std;

int main() {
    // 声明语法：queue<数据类型> 队列名;
    queue<int> q; // 存顾客编号
    // queue<string> qs; 存顾客姓名

    q.push(1); // 1号顾客排队
    q.push(2); // 2号顾客排队

    cout << "当前队首是: " << q.front() << endl; // 看一眼，输出 1

    q.pop(); // 1号顾客结账走人

    cout << "新的队首是: " << q.front() << endl; // 看一眼，输出 2

    return 0;
}
```

---

4. 实例

例 15-9 约瑟夫问题 (B0221)
题目：n 个人围成一圈，从第一个人开始报数，数到 k 的人出列。再由下一个人重新从 1 开始报数，数到 k 的人再出圈... 依次类推，直到淘汰 $n-1$ 名小朋友。请按顺序输出每个出圈人的编号。

- **思考**：
  题目说小朋友坐成一个圈，我们的直觉是画一个圆去模拟。但在计算机内存里，数据通常是**排成一排**的（线性）。怎么让这一排数据能转起来呢？
- **算法**：
  想象一个超市收银台的回转传送带。传送带本身是直的，但在尽头会把没被拿走的数据**送回起始点**！
  对于这个传送带而言，先放进去的数据，会先传到尽头，这不就是队列吗！我们只需要加上一个特殊操作：**队首回填队尾。**
- **步骤拆解（数到 k 淘汰）**：
  假设队列里是 1 到 n。我们要数 k 个人：

1. **对于前 $k-1$ 个人（不用出局）**：
   操作：先获取队首元素 `x = q.front()`，把他踢出队首 `q.pop()`，紧接着让他重新在队尾排队 `q.push(x)`。
2. **对于第 k 个人（必须出局）**：
   操作：看一眼队首并输出他的编号，然后直接踢出去 `q.pop()`，**不再**让他回队尾。

队列 + 队首回填队尾 叫做**逻辑环形队列**，在很多需要周期性、环形逻辑的地方都有效！

---

5. 总结

- **要点回顾**：
- **队列 (Queue)** 是一种遵循**先进先出 (FIFO)** 原则的数据结构。
- **入队 (`push`)**：新元素从队尾进入。
- **出队 (`pop`)**：老元素从队首离开。
- **看队首 (`front`)**：获取准备第一个离开的元素。
- **看队尾 (`back`)**：获取最后进入的元素。
- **适用场景**：
  只要问题符合**先产生、先处理**，需要公平排队或**逐层扩散**的逻辑，就适合用队列。例如：数据流动模型（滑动窗口）、资源调度模型（打印机排队）等。