

线性表特殊例子：栈和队列

Tan Yiqing

2025 年 10 月 31 日



1 栈的逻辑结构

1.1 定义

栈是只允许在固定一端进行插入和删除操作的线性表。允许插入和删除的一端称为栈顶，另一端称为栈底。

栈的插入操作称为进栈 (push)，删除操作称为出栈 (pop)。栈中最后插入的元素最先被删除，称为后进先出 (LIFO, Last In First Out) 原则。

1.2 出栈序列的数量

设栈中有 n 个元素，按照 1 到 n 的顺序进栈 (注：只要求顺序符合，不一定一口气进完)，则所有可能的出栈序列共有 $\frac{1}{n+1} \binom{2n}{n}$ 种。

例如： $n=3$ 时，共有 5 种出栈序列，分别为 123,132,213,231,321。312 没有的原因是要想弹出 3，必须先把 1 和 2 压入栈中，可是要想输出 1，必须输出 2，矛盾。

这一现象与 Catalan 数有关，我们将它与二叉树建立双射关系来证明。

因此递推公式为：

$$C_n = \sum_{i=0}^{n-1} C_i \cdot C_{n-1-i}$$

1.3 常见函数

1.3.1 构造函数

- 前置条件：栈不存在
- 输入：无
- 功能：栈的初始化
- 输出：无
- 后置条件：栈存在且为空

1.3.2 析构函数

- 前置条件：栈存在
- 输入：无
- 功能：销毁栈，释放动态申请的空间
- 输出：无
- 后置条件：栈不存在

1.3.3 判断长度

- 前置条件：栈存在
- 输入：无
- 功能：回传栈的长度
- 输出：栈的长度
- 后置条件：栈没有变化

1.3.4 判空

- 前置条件：栈存在
- 输入：无
- 功能：判断栈是否为空
- 输出：如功能
- 后置条件：栈没有变化

1.3.5 进栈

- 前置条件：栈存在
- 输入：待插入元素 x
- 功能：将元素插入栈顶
- 输出：无
- 后置条件：栈顶指针上移，栈长度加 1

1.3.6 出栈

- 前置条件：栈存在
- 输入：无
- 功能：将栈顶元素弹出。若栈为空则报错
- 输出：栈顶元素
- 后置条件：栈顶指针下移，栈长度减 1，若栈为空则报错

1.3.7 取栈顶元素

- 前置条件：栈存在
- 输入：无
- 功能：获取栈顶元素。若栈为空则报错
- 输出：栈顶元素
- 后置条件：栈没有变化

2 栈的实现

用数组实现，与向量表很像，关键指标 $last$ 指向栈顶元素， top 指向栈顶元素的下一个位置。

2.1 顺序结构

2.1.1 入栈

$last$ 右移，插入数据

2.1.2 出栈

last 左移，回传数据，注意空栈的情况 ($last == -1$)

2.1.3 其他注意点

- 注意栈满的情况，但是时间复杂度有变化。
- 两个栈共享空间，一个栈从数组左端开始，一个栈从数组右端开始。

2.2 链式结构

这次不使用空结点，而是直接使用链表的头结点作为栈顶元素。所有插入都在开始 (头插法)，删除也在开始。操作基本上与单链表的头插法和头删法一样。空表为栈顶指针指向空白。

3 队列的逻辑结构

3.1 定义

队列是只允许在一端进行插入操作，而在另一端进行删除操作的线性表。允许插入的一端称为队尾，允许删除的一端称为队头。

队列的插入操作称为入队 (enqueue)，删除操作称为出队 (dequeue)。队列中最先插入的元素最先被删除，称为先进先出 (FIFO, First In First Out) 原则。

3.2 常见函数

3.2.1 构造函数

- 前置条件：队列不存在
- 输入：无
- 功能：队列的初始化
- 输出：无
- 后置条件：队列存在且为空

3.2.2 析构函数

- 前置条件：队列存在
- 输入：无
- 功能：销毁队列，释放动态申请的空间
- 输出：无
- 后置条件：队列不存在

3.2.3 判断长度

- 前置条件：队列存在
- 输入：无
- 功能：回传队列的长度
- 输出：队列的长度
- 后置条件：队列没有变化

3.2.4 判空

- 前置条件：队列存在
- 输入：无
- 功能：判断队列是否为空
- 输出：如功能
- 后置条件：队列没有变化

3.2.5 入队

- 前置条件：队列存在
- 输入：待插入元素 x
- 功能：将元素插入队尾
- 输出：无
- 后置条件：队尾指针后移，队列长度加 1

3.2.6 出队

- 前置条件：队列存在
- 输入：无
- 功能：将队头元素弹出。若队列为空则报错
- 输出：队头元素
- 后置条件：队头指针后移，队列长度减 1，若队列为空则报错

3.2.7 取队头元素

- 前置条件：队列存在
- 输入：无
- 功能：获取队头元素。若队列为空则报错
- 输出：队头元素
- 后置条件：队列没有变化

4 队列的实现

基于数组实现，关键是有两个指标 `front` 和 `last`，分别指向队头和队尾。

4.0.1 入队

`last` 右移，插入数据

4.0.2 出队

`front` 右移，回传数据，注意空队列的情况 ($front == last + 1$)

4.1 链式结构

这用的更多一些，规避了数组容量的问题。这里还是使用空节点。出队在队头的方向，跟单链表删除首个元素的一样。如果删除唯一存在的数据，尾指针需要调整位置。入队利用尾指针，插入到尾节点的后面，然后更新尾指针。空队就是头指针和尾指针都指向同一个节点空节点。

5 栈的一个简单应用：四则运算计算器

使用 RPN(Reverse Polish Notation) 逆波兰表达式来计算。示例：

中缀表达式： $1+2*3$

逆波兰表达式： $1\ 2\ 3\ *\ +$

计算过程：如果是数字，入栈，如果是运算符，弹栈两次，计算后入栈。

例如计算 $1\ 2\ 3\ *\ +$ 的过程为：

- 读入 1，压栈
- 读入 2，压栈
- 读入 3，压栈
- 读入 *，弹出 3 和 2，计算 $2*3=6$ ，压栈
- 读入 +，弹出 6 和 1，计算 $1+6=7$ ，压栈

最终结果为 7。