目录

1 数据结构概念

- 在数据结构中，从逻辑上可以把数据结构分为线性结构和非线性结构，非线性结构又可至少被分为树形结构，图状结构与集合。
- 形式参数指在被调用时才会被分配内存的变量，一般指函数后括号内的参数
- 实际参数一般指有明确定义的变量，一般已经被赋好初值，一般形容出现在调用函数时括号内用到的变量
- 评价算法两个重要指标是时间复杂度与空间复杂度。
- 选取数据结构时一般不考虑存储数据的数值。
- 一个数据结构在计算机中的表示(又称映像)称为存储结构。
- 一个算法具有五个特性：有穷性；确定性；可行性；有零个或多个输入；有一个或多个输入.
- 数据结构包括数据的逻辑结构与物理结构.
- 一个抽象数据包括数据类型和数据操作两个部分.
-

1.1 线性数据结构

线性结构是一个有序数据元素的集合。
特征:

1. 集合中必存在唯一的一个 “第一个元素”
2. 集合中必存在唯一的一个”最后一个元素”
3. 每一个元素都存在唯一的一个”前驱”以及存在唯一的一个”后驱”

Eg:字符串、栈与队列为线性结构，树不是

1.2 时间复杂度

时间复杂度T(n)表示执行规模.

$\Theta(g(n)) = \{f(n)$:存在正常量 c_1, c_2, n_0，使得对所有 $n \geq n_0$，有$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$

$\Theta(n)$ 相当于扔掉低阶项，并忽略最高阶项前的系数。用来描述程序的平均运行时间

$O(g(n) = \{f(n)$:存在正常量c和n_0，使得对所有$n \geq n_0$,有 $0 \leq f(n) \leq cg(n)\}$

$O(n))$用来确认时间复杂度的上限，比如插入排序的时间复杂度为$O(n^2$，而实际上因为特定输入的变化，实际变化并不一定是n^2的规模，所以描述的是无论输入为何，都不会超过n^2的规模，一般从程序结构上来分析。要注意，形容的是执行时间与括号内成正比。

$\Omega(g(n) = \{f(n)$:存在正常量c和n_0，使得对所有$n \geq n_0$,有 $0 \leq cg(n) \leq f(n)\}$

$\Omega(g(n))$用来描述一个函数的下限，例如插入排序，若达到最优输入，那么规模仅为n。

2 线性表

线性表的特点:

- 集合必存在一个唯一的”第一元素“
- 集合比存在一个唯一的”最后元素“
- 除最后一个元素外，均存在唯一的后序
- 除第一个元素外，均存在唯一的前驱

2.1 顺序表

2.2 基本概念

顺序表是在计算机内存中以数组的形式保存的线性表，线性表的顺序存储是指用一组地址连续的存储单元依次存储线性表中的各个元素、使得线性表中在逻辑结构上相邻的数据元素存储在相邻的物理存储单元中，即通过数据元素物理存储的相邻关系来反映数据元素之间逻辑上的相邻关系，采用顺序存储结构的线性表通常称为顺序表。顺序表是将表中的结点依次存放在计算机内存中一组地址连续的存储单元中。

2.3 特点

- 随机访问，可在O(1)的时间内找到第i各元素
- 存储密度高，每个节点只存储数据元素本身
- 拓展容量不方便
- 插入、删除操作不方便，需要移动大量元素。

2.4 链表

2.5 基本概念

链表是一种物理存储单元上非连续、非顺序的存储结构，数据元素的逻辑顺序是通过链表中的指针链接次序实现的。链表由一系列结点（链表中每一个元素称为结点）组成，结点可以在运行时动态生成。每个结点包括两个部分：一个是存储数据元素的数据域，另一个是存储下一个结点地址的指针域。相比于线性表顺序结构，操作复杂。由于不必须按顺序存储，链表在插入的时候可以达到O(1)的复杂度，比另一种线性表顺序表快得多，但是查找一个节点或者访问特定编号的节点则需要O(n)的时间，而线性表和顺序表相应的时间复杂度分别是O(logn)和O(1)。

2.6 特点

- 存储方式为非连续，大小不固定.
- 插入与删除处理快，但是查询速度慢.

静态链表与动态聊表的区别

- 静态链表需要预先分配一段存储空间(又是需要额外分配一个数组来存储空闲空间的位置),想必动态链表则是动态申请空间，不需要考虑规模。
- 操作上，删除添加等都不需要做元素的移动，但是动态数组可以使用申请或删除来直接取消，相比于静态数组会比较方便。
-

2.7 附件

- 带头结点的链表表示头节点本身不存储任何内容，但是头结点的下一项存储着链表内的第一个元素，所以判断链表状态的方法为头结点的下一项是否为空；

3 栈与队列

3.1 特点

- 顺序存储结构和链式存储结构是栈结构通常采用的存储结构。
- 都是限制存储点的线性结构，不可以随机存取栈或队列中的值。
- 栈为先入后出，队列为先入先出
- 插入与删除复杂度均为O(1).
- 由两个栈共享一篇来内需的的内存空间时，应将两栈栈底分别设在这片内存的两端，这样只有当两个栈顶相邻时才会产生上溢。

- 都有可能出现假溢出.

3.2 队列的应用

- 非线性数据结构的搜索
- 信息缓冲器
- 操作系统的各种资源管理

3.3 栈的应用

- 判断左右括号配对的算法，采用栈结构。

3.4 队列的应用

避免队列出现假溢出现象可用一下集中方法解决:

- 采用移动元素的方法，每当由一个新元素入队，就将队列中已有的元素向队头移动一个位置，假定空余空间足够。
- 每当删去一个队头元素，则可依次移动队列中的元素总是使front指针指向队列中的第一个位置。
- 采用循环队列方式，将队头队尾看作是一个首尾相接的循环队列，即用循环数组实现，此时队首仍在队尾之前，作插入和删除运算时仍遵循”先进先出”的原则，

- 循环队列计算元素个数的方法为(尾指针-头指针+元素数)%元素数

4 数组与广义表

4.1 特殊矩阵

定义: 具有很高的阶数，矩阵中有许多值相同的元素或者是零元素，且它们的分布有一定规律，

问题: 存储全部的元素，造成空间浪费

- 多维数组可以看作数据元素也是基本线性表的基本线性表
- 以行为主序或以列为主序对于多维数组的存储没有影响
- 对于不同的特殊矩阵应该采用不同的存储方式。
- 特殊矩阵的主要形式有:
    1. 对称矩阵
    2. 上三角矩阵\下三角矩阵
    3. 对角矩阵
- 广义表的表头可以是广义表，也可以是单个元素.
- 广义表的表尾一定是广义表。
- 广义表的元素可以是子表，也可以是单元素。