## 1. 概念

### 1.1数据的逻辑结构和物理结构

**数据的逻辑结构是指数据之间的逻辑关系。分为线性结构和非线性结构。**

**线性结构**是指：除第一个数据元素和最后一个数据元素外，每个数据元素有且只有一个前驱元素和一个后驱元素。

一维数组

队列

栈

**非线性结构**是指：一个数据元素可能会有零个或者多个数据前驱，可能会有零个或者多个数据后驱。

树

图

多维数组

集合

**数据的物理结构**是指：数据元素在计算机中的存储方式的表示。分为顺序存储和链式存储

顺序存储是指：把数据元素存储在一块连续地址空间的内存中。特点是，逻辑上相邻的数据元素在物理上也相邻。

链式存储是指：链式存储结构的关键是使用节点，节点是由数据元素域与指针域组合的一个整体，指针将相互关联的节点衔接起来。其特点是逻辑上相邻的元素在物理上不一定相邻，数据间的逻辑关系表现在节点的衔接关系上。

按存储结构分类：

顺序存储结构

链式存储结构

索引存储结构

散列存储结构

总结

数据的逻辑结构是从逻辑关系角度观察数据，它与数据的存储无关，是独立于计算机的。而数据的存储结构是逻辑结构在计算机内存中的实现，它是计算机处理的逻辑。

### 1.2 一维数组

声明：

|  |
| --- |
| // 数组的命名和赋值  **int**[] arr1 = **new** **int**[5];// 方式一  String[] arr2 = **new** String[] { "1", "32", "///" };// 方式二  Object[] arr3 = { 1, 1.22, "asas", **new** Object() };// 方式三  // 数组的取值  **int** i2 = arr1[2];  log.info("i2={}", i2);  // 数组的赋值  arr1[3] = 333;  log.info("i2={}", arr1[3]); |

### 1.3 栈

栈（stack）是一种特殊的表，只在表的一端进行插入和删除操作

### 1.4 队列

队列是一种特殊的线性表，只允许在表的前端（front）进行删除操作，表的前端称为对头。在表的后端（read）进行插入的操作，表的后端称为对尾。

### 1.5 树

树就是树。包括 无序树、有序树、二叉树、B树。一个节点可以有0个或者多个子节点。

节点：每个元素称为节点（NODE）。

根节点：没有父节点的节点称为根节点。

节点的度：一个节点含有的子树的个数。

树的度：一颗树中，最大的节点的度。

节点的层次：从根开始定义起，根为第1层，根的子节点为第二层。以此类推。'

树的高度（或者叫深度）：树中节点的度最大层次。

堂兄弟节点：双亲在同一层的节点互为堂兄弟。

双亲节点或父节点：若一个节点含有子节点，则这个节点称为其子节点的父节点。

#### 1.5.1 无序树和有序树

无序树：树中的任意节点的子节点之间无顺序关系。也称为自由树。

有序树：树中的任意节点的子节点之间有顺序关系。

（不特别指明，一般讨论的树都是有序树。）

#### 1.5.2 二叉树

二叉树通常指：每个节点最多有两个子树的树结构。分为普通二叉树、满二叉树、完全二叉树、平衡二叉树、霍夫曼树等。

二叉树性质：

|  |
| --- |
|  |

二叉树遍历：

|  |
| --- |
|  |

二叉树遍历：

|  |
| --- |
|  |

线索二叉树：

|  |
| --- |
|  |

满二叉树：除最后一层无任何子节点外，每一层上的所有节点都有两个子节点的二叉树。

（国际定义是：如果一颗二叉树的节点，要么是叶子节点，要么它有两个子节点。）

完全二叉树：一棵二叉树至多只有最下面的两层上的结点的度数可以小于2，并且最下层上的结点都集中在该层最左边的若干位置上，而在最后一层上，右边的若干结点缺失的二叉树，则此二叉树成为完全二叉树。

平衡二叉树（AVL二叉树）：它是一 棵空树或它的左右两个子树的高度差的绝对值不超过1，并且左右两个子树都是一棵平衡二叉树。比如红黑树。

红黑树：

#### 1.5.2 B树

B 树（英文：B-tree）：B是balance的意思。多路搜索查找。常用于数据库索引。

#### 1.5.3 霍夫曼树

### 1.6 图

图是多对多的关系，树是一对多的关系，线性表是一对一的关系

### 1.7 多维数组

### 1.8 集合