# 数据结构基本概念和术语

* 1. 数据结构一些概念
     1. 程序设计=数据结构+算法
     2. 数据：是客观事物的符号表示，是所有能输入到计算机中并被计算机程序处理的符号的总称。
     3. 数据元素：是数据的基本单位。数据元素用于完整描述一个对象。
     4. 数据项：是组成数据元素的，有独立含义的，不可分割的最小单位。
     5. 数据对象：是性质相同的数据元素的集合，是数据的一个子集。
  2. 数据结构
     1. 数据结构：是相互之间存在一种或多种特定关系的数据元素的集合。

数据结构分为逻辑结构和存储结构

逻辑结构：逻辑结构是从逻辑关系上描述数据，与存储无关

逻辑结构要素：1.数据元素 2.关系

分为：集合结构，线性结构，树结构，图结构（网状结构）

思维导图：

数据的逻辑结构

集合结构

特殊线性表

一般线性表

树结构

图结构

线性

非线性

线性表

有向图

无向图

二叉树

树

线性表的推广

广义表

数组

字符串

栈与队列

线性表

* + 1. 存储结构：数据对象在计算机中存储的表示称为数据的存储结构，也称物理结构

分为：顺序存储，链式存储

* 1. 数据类型
     1. 数据类型：高级程序语言设计的一个基本概念
     2. 抽象数据类型：由用户定义，表示应用问题的数学模型，以及定义在这个模型上的一组操作的总称。
  2. 算法
     1. 算法：是为了解决某类问题而规定的一个有限长的操作序列
        1. 特性：有穷性，确定性，可行性，输入，输出
        2. 优劣：正确性，可读性，健壮性，高效性

# 二．链表

1. 算法:

遍历 查找 销毁 清空 求长度 排序 删除节点 插入节点

1. 类型：

单链表，双链表，循环链表

# 三．堆与栈

1. 栈 定义：存储静态变量

分类：

1. 动态栈（内核链表）也称链栈
2. 静态栈（内核数组）

算法：

1. 出栈
2. 进栈

2. 堆 定义：存储动态变量

# 四．队列

定义：实现先进先出的存储结构

1. 静态队列(数组队列)
2. 一般是循环的队列
3. 链表队列

# 五．其他线性结构

1. 串：
   1. 匹配算法：KMP算法，BF算法
2. 广义表：

# 六．树

1. 专业名词

* 节点
* 父节点
* 子节点
* 子孙
* 堂兄弟
* 深度：从根节点到最底层节点的层数称之为深度，根节点是第一层
* 叶子节点：没有子节点的的节点
* 非终端节点：实际就是非叶子节点
* 度：子节点的个数称为度

1. 树分类

* 二叉树：任意一个节点的子节点个数最多两个，且子节点的位置不可更改

分类:

* 1. 一般二叉树
  2. 满二叉树：在不增加树的层数的前提下，无法在多添加一个节点的二叉树就是满二叉树
  3. 完全二叉树
* 一般树：任意一个节点的子节点个数都不受限制
* 森林：n个互不相交的树的集合

1. 树的存储

* 二叉树存储
  + 连续存储【完全二叉树】

优点：查找某个节点·的父节点和子节点（也包括判断有没有子节点）速度很快

缺点：耗用内存空间过大

* + 链式存储
    - 二叉链表 结构体：值域，左右孩子指针
    - 三叉链表 结构体：值域，左右孩子指针，指向双亲指针
    - 双亲链表 结构体：数据域，双亲节点下标，char标识左右子树（由T存储根节点位置，根节点双亲为-1，根节点数目）
    - 线索链表（包含线索的链表）
      * 结构为指向节点的lchild,rchild,判断指向孩子节点还是指向后继LTag,RTag,以及值域
* 一般树存储
  + 双亲表示法（一个数组存俩个值：DATA，父节点下标）
  + 孩子表示法（一个节点，一个存子节点）
  + 双亲孩子表示法（一个节点，一个存父节点下标，一个存子节点）
  + 二叉树表示法（孩子兄弟表示法）（一个节点，左指针指向第一个子节点，右指针指向下一个兄弟）
* 森林存储
  + 二叉树表示法（孩子兄弟表示法）（一个节点，左指针指向第一个子节点，右指针指向下一个兄弟）不同树表示下一个节点

1. 二叉树遍历：顺着某一条路径寻访二叉树中的节点，使得每个节点均被访问一次，而且仅被访问一次
   1. 先序遍历
      1. 先访问根节点
      2. 在先序访问左子树
      3. 在先序访问右子树
   2. 中序遍历
      1. 先中序遍历左子树
      2. 在访问根节点
      3. 在中序遍历右子树
   3. 后序遍历
      1. 先后序遍历左子树
      2. 在后序遍历右子树
      3. 在访问根节点
2. 二叉树算法
   1. 已知两种遍历序列求原始二叉树
      1. 通过先序和中序 或中序和后序（先序后序无法还原）

采用分治的思想

* 1. 二叉树复制（使用后序遍历）
  2. 先序遍历建立二叉树用‘#’表示空树

1. 树遍历
   1. 先序
   2. 中序
   3. 后序
   4. 层级遍历
2. 哈夫曼树
   1. 哈夫曼树：一类带权路径长度最短的树
   2. 路径：从树中一个节点到另一个节点之间分支构成的这俩个节点之间的路径
   3. 路径长度：路径上分支数目
   4. 树的路径长度

# 七．图(Graph)

1. 图定义：图G是由俩个集合组成，记做G=(V,E),V是顶点的有穷非空集合，E是V中顶点偶对的有穷集合（或称为边）。V(G)，E(G)分别表示图G的顶点集合和边集合，E(G)可为空。

对于图G若边集合E(G)为有向边的集合，则图为有向图；若图G边集合为无向边集合，则该图称为无向图

1. 图的基本术语
   1. 子图：若有俩个图G=(V,E)，G’=(V’,E’)，且V’含与V,E’含于E，则G’称为G的子图
   2. 无向完全图和有向完全图：对于无向图，有n(n-1)/2边称为无向完全图；对于有向图，有n(n-1)条边称为有向完全图
   3. 稀疏图和稠密图：对边少于e<nlog2n的图称为稀疏图，反之称为稠密图
   4. 权和网：对于图的每条边可以标注某种含义的值，该值称为改边上权；带权的图称为网。
   5. 邻接点：对与无向图G，如果图的边(V,V’)属于E则称V,V’为邻接点
   6. 度，入度和出度：顶点v的度是值和v相关联的边的数目，记做TD(v)。对与有向图，顶点的度分为入度和出度。入度是以顶点v为头的弧的数目，记做ID(v)；出度是以顶点v为尾的弧的数目，记做OD(v)。
   7. 路径和路径长度：在图中，顶点到顶点的顶点序列称为路径，路径长度是一条路径上经过的边或弧的数目。
   8. 回路或环：第一个顶点和最后一个顶点相同的路径称为回路或者环
   9. 简单路径，简单回路或简单环：序列中不重复出现的路径称为简单路径，除第一个和最后一个顶点，其余顶点不重复出现的回路，称为简单回路或简单环
   10. 连通、连通图和连通分量：在无向图中，顶点v到顶点v’有路径，则v与v’是连通的；对于图任意连顶点都是连通则称图为连通图；连通分量，指的是无向图中的极大连通子图数。
   11. 强连通图和强连通分量：在有向图中，对每对顶点都存在路径，则称图为强连通图。在有向图中的极大强连通子图称作图的强连通分量。
   12. 连通图的生成树：一个极小连通子图，他包含图中全部顶点，但只有足以构成一颗树的n-1条边，这样的连通子图称为连通图的生成树。

（一有n个顶点的生成树有且仅有n-1条边。如果一个图有n个顶点和小于n-1条边，则是非连通图。如果它多余n-1条边，则一定有环。但是有n-1条边的图也不一定是生成树（例：该图不是连通图，连通分量为2或以上））

* 1. 有向树和生成森林：有一个顶点的入度为0，其余顶点入度均为1的有向图称为有向树。一个有向图的生成森林是由若干棵有向树组合。

1. 图的存储结构
   1. 邻接矩阵
      1. 时间复杂度：O(n2)
   2. 邻接表
      1. 表头节点表
      2. 边表
      3. 创建无向图O(n+e)
   3. 十字链表

有向图的结构

弧头，弧尾，指向弧头相同的下一条边，指向弧尾相同的下一条边，权值

可以理解成邻接表和逆邻接表的结合

* 1. 邻接多重表

无向图·存储结构

1. 图的遍历
   1. 深度优先搜索

深度优先生成树

回边：生成树间接得到，非生成树上的边

* 1. 广度优先探索

相当于按层搜索

对于邻接表可用队列

1. 图的应用