day6-8: 数据结构

**********day_1**********
**********基本概念**********
【1】数据结构
研究数据和数据之间的关系
【2】程序 = 数据结构 + 算法
【3】数据结构到底有什么用？
存储数据及其操作
【4】数据（Data)
数据即信息的载体，是能够输入到计算机中并且能被计算机识别、存储和处理的符号总称。
【5】数据元素（Data Element)
数据元素是数据的基本单位，又称之为记录（Record)
【6】数据项
数据元素由若干基本项（或称字段、域、属性）组成，称之为数据项，数据项是数据的最小单位
【7】数据结构概念：
数据结构指的是数据的逻辑结构和存储结构及其操作
逻辑结构：表示数据运算之间的抽象关系（如邻接关系、从属关系等），
按每个元素可能具有的直接前趋数和直接后继数将逻辑结构分为“线性结
构”和“非线性结构”两大类。
线性关系：线性表 栈 队列
树形关系：树
网状关系：图
存储结构：逻辑结构在计算机中的具体实现方法，分为顺序存储方法、
链接存储方法、索引存储方法、散列存储方法。
顺序存储结构：借助元素在存储器中的相对位置来表示数据元素间的逻辑关系
需要在内存中开辟一块连续的空间，使用定义的数组下标去操作数据
链式存储结构：借助指示元素存储地址的指针表示数据元素间的逻辑关系
不需要开辟一块连续的空间，使用指针建立连接
数据运算：对数据进行的操作，如插入、删除、查找、排序等 。
增、删、改、查
【8】算法的定义
算法（Algorithm）是一个有穷规则（或语句、指令）的有序集合,函数的实现
算法设计: 取决于选定的逻辑结构
算法实现: 依赖于采用的存储结构
【9】算法的特性
(1) 有穷性 —— 算法执行的步骤（或规则）是有限的；
(2) 确定性 —— 每个计算步骤无二义性；
(3) 可行性 —— 每个计算步骤能够在有限的时间内完成；
(4) 输入 —— 算法有一个或多个外部输入；
(5) 输出 —— 算法有一个或多个输出。
【10】语句的频度(Frequency Count)
语句频度定义为可执行语句在算法（或程序）中重复执行的次数。
【11】算法的时间复杂度(Time Complexity)
算法的时间复杂度定义为算法中可执行语句的频度之和，记为T(n)。

```
void  MATRIXM(A,B,C)
{   float A[n][n] , B[n][n] , C[n][n];
      { int i, j, k;                                                    语句频度
          for (i=0; i<n; i++)                                  n+1
            for (j=0; j<n; j++)                                  n(n+1)
```

```
    {  C[i][j] = 0;                                    n~2
  for (k=0; k<n; k++)                          n~2(n+1)
   C[i][j] = C[i][j]+A[i][k] * B[k][j];}}            n~3
```

T(n)=(n+1)+n(n+1)+n~2(n+1)+n~3=2n~3+3n~2+2n+1
T(n) = O (n~3)

**********线性表**********

【1】定义

线性表是信息表的一种形式，表中数据元素之间满足线性关系（或线性结构），是一种最基本、最简单的数据结构类型。

【2】线性表的特征：

1) 对非空表,a0是表头,无前驱；
2) an-1是表尾,无后继；
3) 其它的每个元素ai有且仅有一个直接前驱(ai-1)和一个直接后继(ai+1)。

【3】线性表的顺序存储结构有存储密度高及能够随机存取等优点，但存在以下不足：

(1)要求系统提供一片较大的连续存储空间。
(2)插入、删除等运算耗时，且存在元素在存储器中成片移动的现象；

作业：

按照位置插入数据
按照顺序插入数据（自带排序功能）
linklist_sort(h, 10);
10 20 50 40 30
10 20 30 40 50
实现链表的翻转

**********day_2**********

**********栈和队列**********

【1】栈

定义：栈是限制在一端进行插入操作和删除操作的线性表（俗称堆栈），允许进行操作的一端称为“栈顶”，另一固定端称为“栈底”，当栈中没有元素时称为“空栈”。

特点：后进先出（LIFO）。

定义数据类型
定义结构体
创建一个空的栈
判断为空
判断为满
入栈（压栈）
出栈（弹栈）
打印

**********day_3**********

**********树和二叉树**********

【1】树的概念

树（Tree）是n（n≥0）个节点的有限集合T，它满足两个条件：
有且仅有一个特定的称为根（Root）的节点；
其余的节点可以分为m（m≥0）个互不相交的有限集合T1、T2、……、Tm，
其中每一个集合又是一棵树，并称为其根的子树（Subtree）。

【2】度数
一个节点的子树的个数称为该节点的度数，
一棵树的度数是指该树中节点的最大度数。
【3】路径
一个节点系列k1,k2，……,ki,ki+1，……,kj,并满足ki是ki+1的父节点，
就称为一条从k1到kj的路径，路径的长度为j-1,即路径中的边数。
【4】层数
节点的层数等于父节点的层数加一，根节点的层数定义为一。
树中节点层数的最大值称为该树的高度或深度。
【5】树的逻辑结构
树中任何节点都可以有零个或多个直接后继节点（子节点），
但至多只有一个直接前趋节点（父节点），根节点没有前趋节
点，叶节点没有后继节点。
【6】二叉树的定义
二叉树（Binary Tree）是n（n≥0）个节点的有限集合，它或者是空集（n＝0），
或者是由一个根节点以及两棵互不相交的、分别称为左子树和右子树的二叉树组
成。
二叉树与普通有序树不同，二叉树严格区分左孩子和右孩子，即使只有一个子节点也要区分左右。
【7】二叉树的性质
二叉树第i（i≥1）层上的节点最多为2~（i-1）个。
深度为k（k≥1）的二叉树最多有2~k－1个节点
在任意一棵二叉树中，树叶的数目比度数为2的节点的数目多一。
总节点数为各类节点之和：n = n0 + n1 + n2
总节点数为所有子节点数加一：n = n1 + 2*n2 + 1
故得：n0 = n2 + 1；
满二叉树 ：深度为k（k≥1）时有2k－1个节点的二叉树。
完全二叉树 ：只有最下面两层有度数小于2的节点，且最下面一层的叶节点集中在最左边的若干位置上
【8】二叉树的存储
完全二叉树节点的编号方法是从上到下，从左到右，根节点为1号节点。设完全二叉树的节点数为n，某节点编号为i
当2*i≤n时，有左孩子，其编号为2*i ,否则没有左孩子，本身是叶节点;
当2*i＋1≤n时，有右孩子，其编号为2*i+1 ,否则没有右孩子；

如果根节点为0，则用2*i＋1≤n判断左 用2*i＋2≤n判断右

【9】二叉树的遍历
先序遍历: 先访问树根，再访问左子树，最后访问右子树；根 左 右
中序遍历：先访问左子树，再访问树根，最后访问右子树；左 根 右
后序遍历：先访问左子树，再访问右子树，最后访问树根；左 右 根