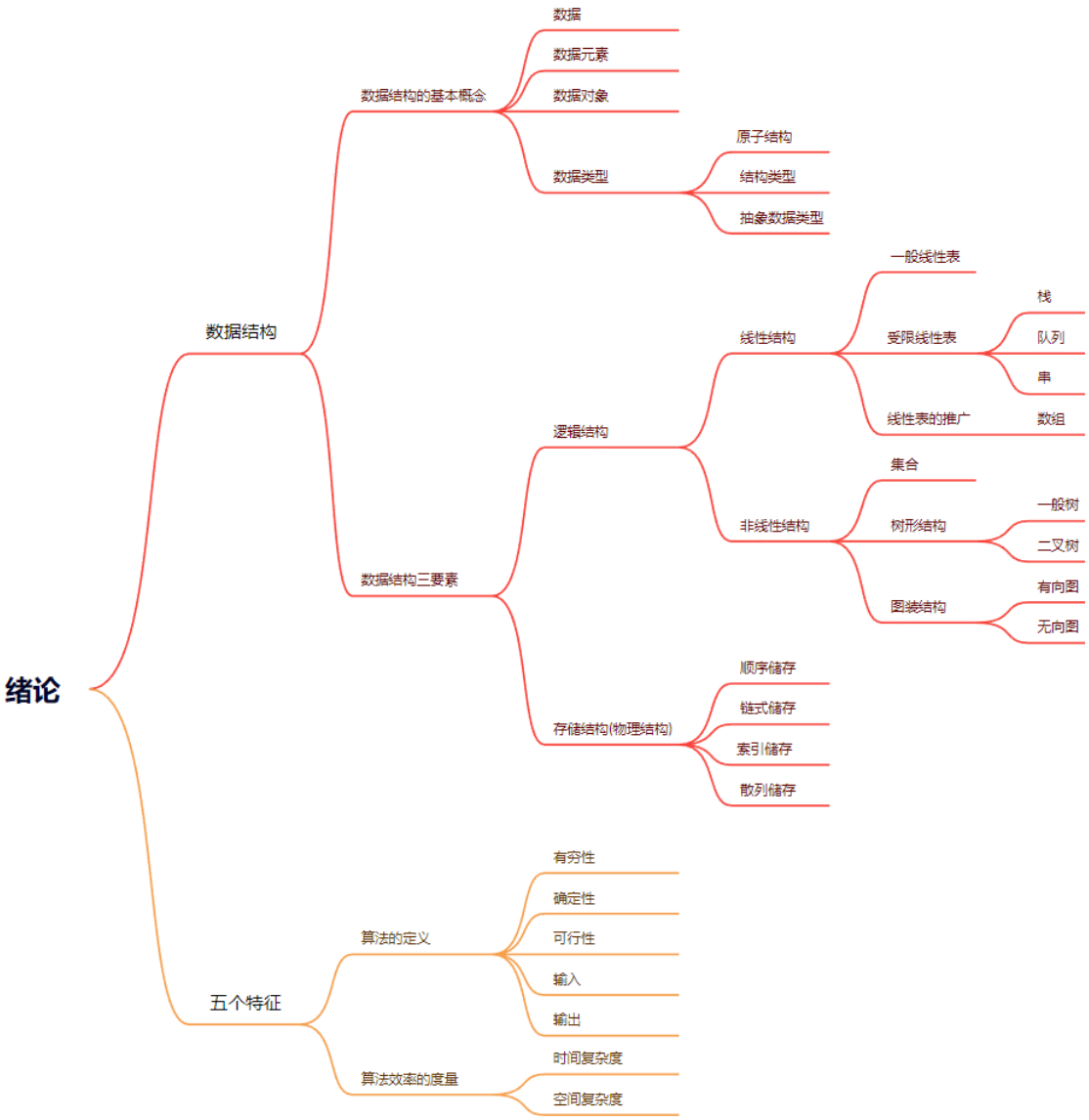
第一章.绪论



1.数据结构的基本概念

1.1基本概念及其术语

数据:信息的载体(所有能输入到计算机中,并能被计算机识别处理的符号集合,如:数,字符等)

数据元素: 数据的基本单位

数据项:构成数据元素不可分割的最小单位

一个数据元素由若干个数据项构成

如:学生记录就是一个数据元素,它由学号,性别,姓名等数据项组成

数据对象:具有相同性质的数据元素集合,是数据的一个子集,如:整数的数据对象集合N={...-2,-1,0,1,2...}

数据类型:一个值的集合和定义在此集合上一组操作的总称,如:一个学生的结构体中,除了有学号,姓名等属性变量,还会有所定义的方法

1. **原子类型:其值不能再分的数据类型**,如:int类型
2. **结构类型:其值还能再分解为若干成分的数据类型**,如:有多个属性的结构体(类)
3. **抽象数据类型(ADT):抽象数据组织及与之相关的操作**,如:有多个属性及函数(方法)的结构体(类)

数据结构:相互之间存在一种或多种特定关系的数据元素的集合

这种数据元素之间的关系称为**结构**

数据结构包括三方面的内容:

1. **逻辑结构**
2. **存储结构**
3. **结构的运算**

存储结构与逻辑结构密不可分:算法的设计取决于选定的逻辑结构,而算法的实现依赖于所采用的存储结构

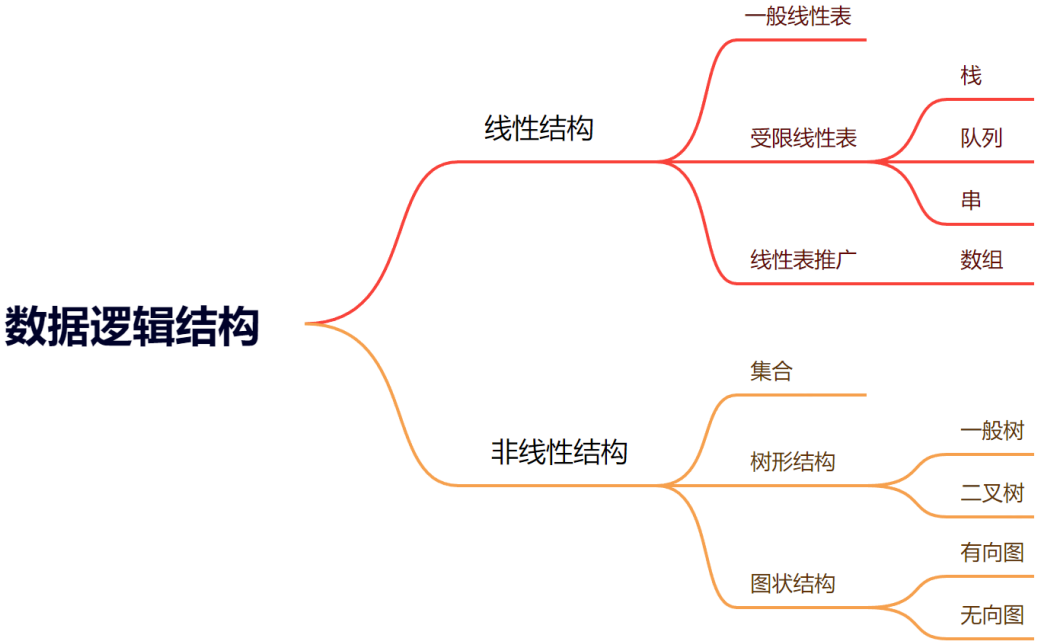
1.2数据结构三要素

(注:之前没有相关基础,推荐优先跳过该节,学完后再回顾这里)

1.2.1逻辑结构

逻辑结构:元素之间的逻辑关系,即从逻辑关系上描述数据,与存储无关

逻辑结构的分类:



1.2.2数据的存储结构

存储结构:数据结构在计算机中的表示,也称物理结构,它包括**数据元素的表示和关系的表示**

主要的存储结构有:顺序存储,链式存储,散列存储

顺序存储:利用邻接关系体现元素之间的关系

- **优点**:可以实现**随机存取**,每个元素占用最少的存储空间
- **缺点**:只能使用相邻的一整块存储单元,因此可能会产生较多的外部碎片

随机存取:当存储器中的数据被读取或写入时，所需要的时间与该数据所在的物理地址无关。例如:数组中我们可以直接通过下标找到相应的数据

链式存储:借助指示元素的指针来表示元素之间的逻辑关系

- **优点**:不会出现碎片现象,能充分利用所有的存储单元
- **缺点**:因为每个元素存储指针需要占用额外的存储空间,因此只能实现顺序存储

索引存储:依靠索引表

- **优点**:检索速度快
- **缺点**:附加的索引表额外占用空间,另外,增加和删除元素需要修改索引表,会花费较多时间

散列存储(哈希存储):根据关键字直接计算出存储地址

- **优点**:检索速度快

- **缺点**:若散列函数不好,则可能出现元素存储单元的冲突,而解决冲突会增加时间和空间的开销

1.3例题

(注:之前没有相关基础,推荐优先跳过该节,学完后再回顾这里)

1.可以用()定义一个完整的数据结构

A.数据元素　　B.数据对象 C.数据关系 D.抽象数据类型

答:D

2.非线性数据结构有那些?

答:看上述1.2.1逻辑结构

3.以下属于逻辑结构的是()

A.顺序表 B.哈希表 C.有序表 D.单链表

答:C,因为A顺序表是属于顺序存储结构,B哈希表属于散列存储结构,D单链表属于链式存储结构

4.以下与存储结构无关的属于是()

A.循环队列 B.链表 C.哈希表 D.栈

答:D,(易错)A循环队列是用顺序表存储的队列是一种数据结构,B是链式存储结构,C是散列存储结构,而D栈是一种逻辑结构,抽象的数据类型,可采用顺序或链式的存储结构

5.在存储数据时,通常不仅要存储各元素之间的值,而且要存储()

答:数据元素之间的关系

2算法和算法评价

2.1算法的基本概念

算法:对特定问题的求解步骤的一种表述,它是指令的**有限序列**,其中的每条指令表示一个或多个操作

一个算法应具有以下5个特征:

1. **有穷性**:每一步都能在有穷的时间内完成
2. **确定性**:每一条指令有确切的含义
3. **可行性**:算法中描述的操作是可以通过基本运算执行有限次实现
4. **输入**:有0或多个输入
5. **输出**:有一个或多个输出

一个优秀的算法应达到:

1. **正确性**:能够正确求解问题
2. **可读性**:算法应能够正确解决问题
3. **健壮性:**输入非法数据,能够做出相应的处理
4. **高效率,低存储性**:算法所需的时间,空间量少

2.2算法效率的度量

2.2.1时间复杂度

频度:该语句在算法中被重复执行的次数

如:i=i+1被执行一次,它的频度便为1

时间复杂度:算法中所有语句的频度之和,记为T(n)=O(f(n))

其中O的含义是T(n)的数量级

在分析时间复杂度时,一般将n看作无穷大,每次分析程序中,只要寻找到并列中的高阶无穷大即可

例如:

例1:

```
void fun(int n){
    int i=1;
    while(i<=n)
        i=i*2;
}
```

在这个函数中,int i=1这个式子只运行一次,与之并列的while语句假设运行t次,即

$$2^t <= n$$

,即

$$t = log_2 n$$

,因为有未知数n看作无穷大,比一次大得多,因此1次可省略,它的时间复杂度为

$$O(log_2 n)$$

例2:

```
void fun(int n){
    int i=0;
    whlie(i*i*i<=n)
        i++;
}
```

与上题同理,我们只需要看while语句中的执行次数,假设执行t次,即

$$t^3 <= n$$

即

$$\sqrt[3]{n}$$

例3:

```
int m=0,i,j
for(i=1;i<=n;i++)
    for(j=1;j<=2*i;j++)
        m++;
```

执行次数有:

$$\sum_{i=1}^{n}\sum_{j=1}^{2i} = \sum_{i=1}^{n} 2i = 2\sum_{i=1}^{n} i = n(n+1)$$

2.2.2空间复杂度

空间复杂度S(n)定义为算法所需要耗费额存储空间S(n)=O(g(n))