# 第一章 绪论

2022年9月29日 星期四 21:45

## 常考点

| |
|---|
| 1.存储数据时，不仅要存储数据元素的值，还要存储数据元素之间的关系 |
| 2.数据的逻辑结构独立于其存储结构 |
| 3.循环队列是用顺序表表示的队列，是一种具体的数据结构 |
| 4.栈/队列是一种抽象数据类型，可采用顺序存储或链式存储，只表示逻辑结构 |
| 5.有序表只表示逻辑结构 |
| 6.顺序表，哈希表，单链表既描述逻辑结构，又描述存储结构和数据运算 |
| 7.可以用抽象数据类型定义一个完整的数据结构 |

## 数据结构基本概念和术语【了解即可】

| | |
|---|---|
| 数据 | • 信息的载体 |
| 数据元素 | • 数据的基本单位 |
| 数据对象 | • 具有相同性质的数据元素的集合，是数据的一个子集 |
| 数据类型 | • 一个值的集合和定义在此集合上的一系列操作的总称<br>原子类型：其值不可再分的数据类型<br>结构类型：其值可以再分解为若干成分的数据类型<br>抽象数据类型：抽象数据组织及与之相关的操作 |
| 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 |

## 数据结构三要素（之后所有的数据结构笔记从下面👇三方面入手，先看逻辑结构，再看存储结构，最后看数据的运算！！！）

**逻辑结构**

- 数据元素之间的逻辑关系，即从逻辑关系上描述数据
- 它与数据的存储无关，是独立于计算机的
- 逻辑结构主要分为线性结构（线性表，栈，队列）和非线性结构（图、树，集合）

| 常见的逻辑结构 | 数据元素的关系 |
|---|---|
| 集合 | 同属一个集合 |
| 线性结构 | 一对一 |
| 树形结构 | 一对多 |
| 图状结构或网状结构 | 多对多 |

**物理结构**

- 数据结构在计算机中的表示（又称映像），也称存储结构
- 包括数据元素的表示和关系的表示
- 它是用计算机语言实现的逻辑结构，它依赖于计算机语言
- 主要的存储结构

| | 定义 | 优点 | 缺点 |
|---|---|---|---|
| 顺序存储 | • 按顺序 | • 可以实现随机存取<br>• 每个元素占用很少的存储空间 | • 只能使用相邻的一整块存储单元<br>• 可能产生较多的外部碎片 |
| 链式存储 | • 借助指示元素存储地址的指针 | • 不会出现碎片现象<br>• 能充分使用所有存储单元 | • 每个元素因存储指针而占用额外的存储空间<br>• 只能实现顺序存取<br>• 不同节点的存储空间可以不连续<br>• 同一节点内的存储空间要连续 |
| 索引存储 | • 建立附加的索引表 | • 检索速度快 | • 附加的索引表额外占用存储空间<br>• 增加和删除数据时也要修改索引表<br>• 会花费较多的时间 |
| 散列存储 | • 根据元素的关键字直接计算出元素的存储地址<br>• 也叫哈希存储 | • 检索、增加和删除节点的操作都很快 | • 若散列函数不好，则可能出现元素存储单元的冲突<br>• 解决冲突又会增加时空开销 |

**数据的运算**

- 包括运算的定义和实现
- 运算的定义是针对逻辑结构的，指出运算的功能
- 运算的实现是针对存储结构的，指出运算的具体操作步骤

## 算法与算法评价

### 算法的基本概念

| | | |
|---|---|---|
| 算法 | • 对特定问题求解步骤的一种描述，是指令的有限序列，其中的每条指令都代表一个或多个操作 | |
| 5个重要特性 | 有穷性 | • 一个算法必须总在有限步骤后结束，每个步骤可在有限时间内完成 |
| | 确定性 | • 算法中每条指令必须有确切的含义，对于相同的输入只能得出相同的输出 |
| | 可行性 | • 算法中描述的操作都可以通过已经实现的基本运算执行有限次来实现 |
| | 输入 | • 一个算法有零个或多个输入，这些输入取自于某个特定的对象的集合 |
| | 输出 | • 一个算法有一个或多个输出，这些输出是与输入有特定关系的量 |
| 预期目标 | 正确性 | • 算法应能正确解决问题 |
| | 可读性 | • 以帮助人们理解 |
| | 健壮性 | • 当输入非法数据时，算法能适当地做出反应或进行处理，而不会产生莫名其妙的输出结果 |
| | 高效率与低存储量的需求 | • 效率指算法执行的时间，存储量需求指算法执行过程中所需要的最大存储空间，两者都与瓦恩替的规模有关 |

### 算法效率的度量

#### 常见算法的时间复杂度

| | |
|---|---|
| 汉诺塔递归算法 | $O(2^n)$ |
| | |

#### 常见的时间复杂度

| 常数阶O(1) | 对数阶O(logN) | 线性阶O(n) | 线性对数阶O(nlogN) | O(n*m) | K次方阶O($n^k$) |
|---|---|---|---|---|---|
| 无复杂结构 | $2^n$次执行后退出循环 | 常有循环结构 | 将时间复杂度为O(logn)的代码循环N遍 | 第一层执行m次，第二层执行n次，共m*n次 | k层嵌套循环 |

```
int i = 1;
int j = 2;
++i;
j++;
int m = i + j;
```

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

```
for (i = 1; i <= n; ++i)
{
    j = i;
    j++;
}
```

```
for (m = 1; m < n; m++)
{
    i = 1;
    while (i < n)
    {
        i = i * 2;
    }
}
```

```
for (x = 1; i <= m; x++)
{
    for (i = 1; i <= n; i++)
    {
        j = i;
        j++;
    }
}
```

```
for (x = 1; i <= n; x++)
{
    for (i = 1; i <= n; i++)
    {
        j = i;
        j++;
    }
}
```

#### 常见的空间复杂度

| O(1) | O(n) |
|---|---|
| 代码中的 i、j、m 所分配的空间都不随着处理数据量变化 | 第一行new了一个数组出来，这个数据占用的大小为n<br>这段代码的2-6行，虽然有循环，但没有再分配新的空间 |

```
int i = 1;
int j = 2;
++i;
j++;
int m = i + j;
```

```
int[] m = new int[n];
for (i = 1; i <= n; ++i)
{
    j = i;
    j++;
}
```

#### 计算方法

- 求执行次数的规模即可,不要想着根据n的值去推公式
- 先判断退出循环的条件，然后构建某语句执行次数和n之间的关系，解出的规模就是时间复杂度

13. 【2019 统考真题】设 n 是描述问题规模的非负整数，下列程序段的时间复杂度是( B )。

```
x=0;
while (n>=(x+1)*(x+1))
x=x+1;
```

A. $O(\log n)$　　B. $O(n^{1/2})$　　C. $O(n)$　　D. $O(n^2)$

设第 k 次循环结束

则第 k 次 $(x+1)^2>n$，x 的初值=0，第 k 次时 $x=k-1$

即 $k^2>n$，$k>\sqrt{n}$

12.【2017 统考真题】下列函数的时间复杂度是( B )。

```
int func(int n){
    int i=0, sum=0;
    while(sum<n) sum += ++i; => i+=i ; sum+=i
    return i;
}
```

A. $O(\log n)$　　B. $O(n^{1/2})$　　C. $O(n)$　　D. $O(n\log n)$

i=1 时，sum=0+1

i=2，sum=0+1+2

i=3，sum=0+1+2+3

sum = 0+1+2+3+…+i = $\frac{i(i+1)}{2}$

则 $\frac{i(i+1)}{2}<n$

=> $i(i+1)<2n$

≈ $i^2<n$ => $i<\sqrt{n}$