数据结构

线性表（一）

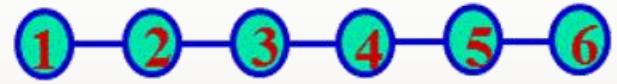
2021年09月

电子与深圳大学信息工程学院 周飞

2 第二章 线性表

第一节 线性表

- 一、线性数据结构的特点



  - 在数据元素的非空有限集中
    - 1、存在唯一的一个被称做“第一个”的数据元素；
    - 2、存在唯一的一个被称做“最后一个”的数据元素；
    - 3、除了“第一个”外，集合中的每个元素均只有一个前驱；
    - 4、除了“最后一个”外，集合中的每个元素均只有一个后继；

3 第二章 线性表

第一节 线性表

- 二、线性表(Linear List)
  - 线性表是最简单的一种线性数据结构
  - 线性表是由n个数据元素组成的有限序列，相邻数据元素之间存在着序偶关系:

    $<a_1, a_2>, <a_2, a_3>,\ldots <a_i, a_{i+1}>,\ldots <a_{n-1}, a_n>$

    $(a_1, a_2, a_3,\ldots a_i, a_{i+1},\ldots a_{n-1}, a_n)$

    其中i表示了元素a_i的位置

第二章 线性表

第一节 线性表

- 二、线性表
  - 线性表中的元素具有相同的特性，属于同一数据对象
    - 比如：
    - 26个字母组成的字母表：（A,B,C,…,Z）
    - 这周每天的平均温度：

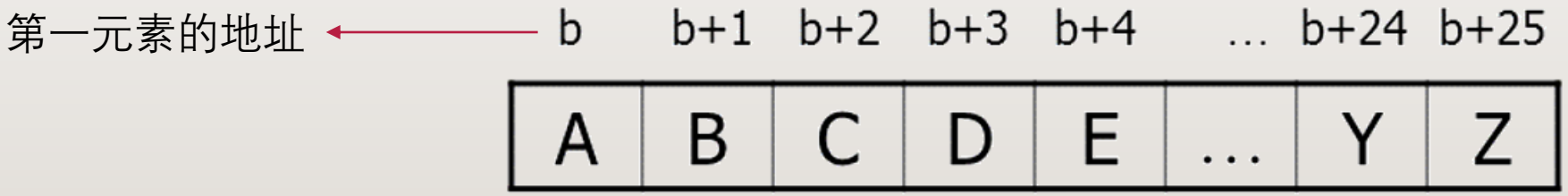
      （28℃, 29℃, 27℃, 30 ℃, 28℃）

不能让奇怪的东西混进来!

5 第二章 线性表

第二节 顺序表

- 一、顺序表
  - 顺序表是线性表的顺序存储表示
  - 采用一组地址连续的存储单元依此存储线性表中的数据元素



每个字母占用一个内存单元

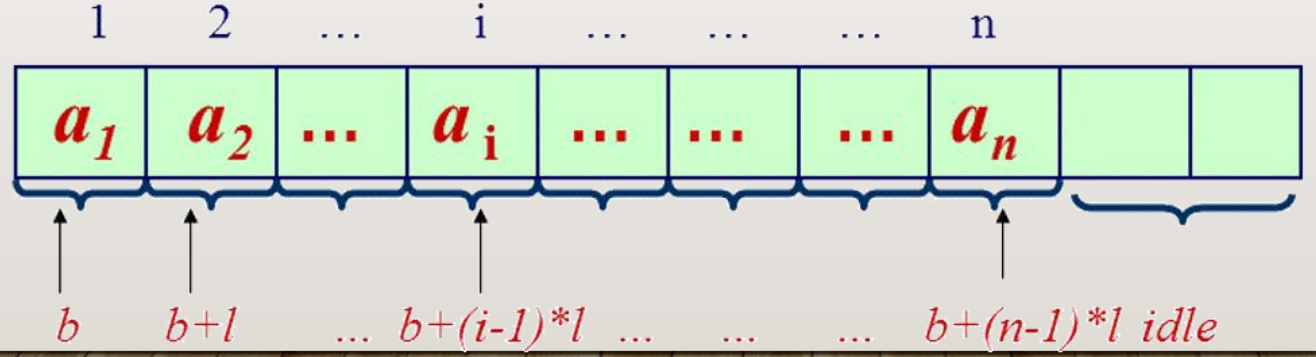
6 第二章 线性表

第二节 顺序表

- 一、顺序表（元素位置）
  - 顺序表中元素的位置:

$$LOC(a_i) = LOC(a_{i-1}) + l$$

$$LOC(a_i) = LOC(a_1) + (i-1)*l$$



l表示元素占用的内存单元 数

第二章 线性表

第二节 顺序表

- 二、顺序表的定义和创建
  - 可以用C语言中的一维数组表示（定义）顺序表

```
#define MAXLISTLEN 100                    //最大表长
int ListLen = 0;                          //当前表长
int SeqList[MAXLISTLEN];                  //顺序表
```

8 第二章 线性表

第二节 顺序表

- 三、顺序表的访问
  - 访问并输出指定位置（第i个）的数据元素值

```
int AcquireSeqList(int i)                    //返回指定位置的元素值
{
    return(SeqList[i-1]);                    //i从1开始计数
}
```

健壮性差!

第二章 线性表

第二节 顺序表

- 三、顺序表的存取
  - 访问并输出指定位置（第i个）的数据元素值

```
int AcquireSeqList(int i)
{
    if (i > ListLen)
        return -1;
    else
        return(SeqList[i-1]);
}
```

健壮性好!

时间复杂度是?

O(1)

第二章 线性表

第二节 顺序表

- 四、顺序表的插入
  - 顺序表的插入是指在顺序表的第i-1个数据元素和第i个数据元素之间插入一个新的数据元素（记为e）
  - 将长度为n的顺序表

$$(a_1, \ldots a_{i-1}, \ a_i, \ \ldots, \ a_n)$$

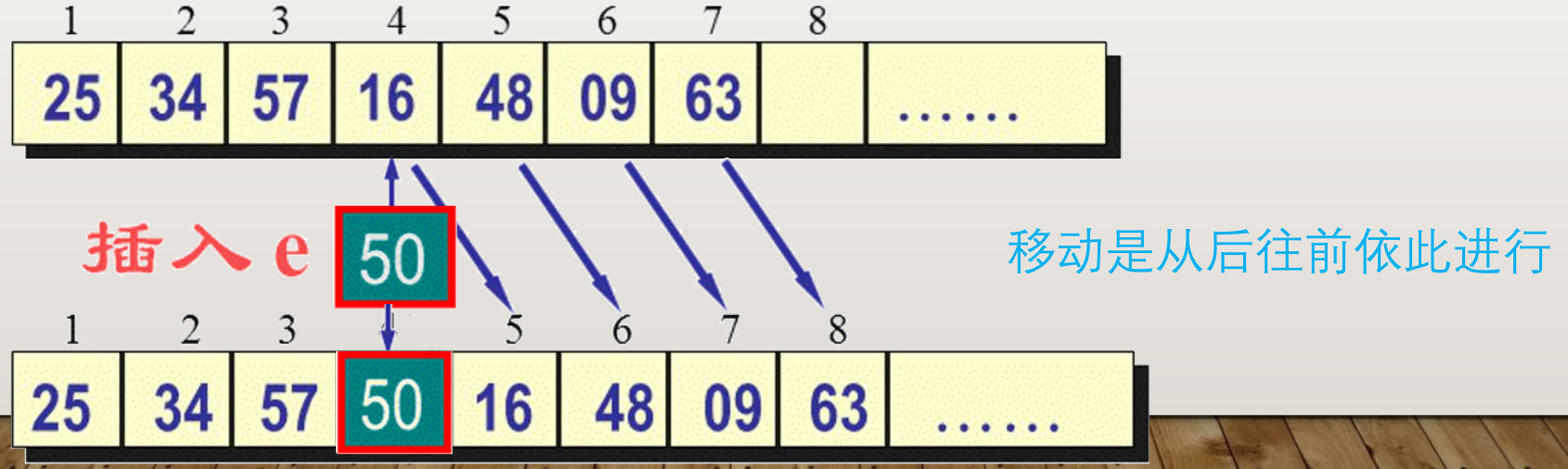
变成长度为n+1的顺序表

$$(a_1, \ldots a_{i-1}, \ e, \ a_i, \ \ldots, \ a_n)$$

第二章 线性表

第二节 顺序表

- 四、顺序表的插入（操作举例）
  - 在第3个元素和第4个元素之间插入新元素e
  - 需要将最后n至第4个元素（共7-4+1）都向后移动一位



第二章 线性表

第二节 顺序表

- 四、顺序表的插入
  - 在顺序表中，第i个位置插入一个元素，需要向后移动的元素个数为

    n-i+1
  - 平均移动元素的操作次数为

$$E_{is} = \sum_{i=1}^{n+1} p_i \times (n-i+1)$$

在第i个位置插入元素的概率

第二章 线性表

第二节 顺序表

- 四、顺序表的插入
  - 当插入位置等概率时，$p_i=1/(n+1)$
  - 因此，

$$E_{is} = \sum_{i=1}^{n+1} [1/(n+1)] \times (n-i+1) = n/2$$

顺序表插入操作的时间复杂度为O(n)

第二章 线性表

第二节 顺序表

- 五、顺序表的删除
  - 顺序表的删除是指将顺序表中第i个数据元素删除
  - 将长度为n的顺序表

$$(a_1, \ldots a_{i-1}, \ a_i, \ \ldots, \ a_n)$$

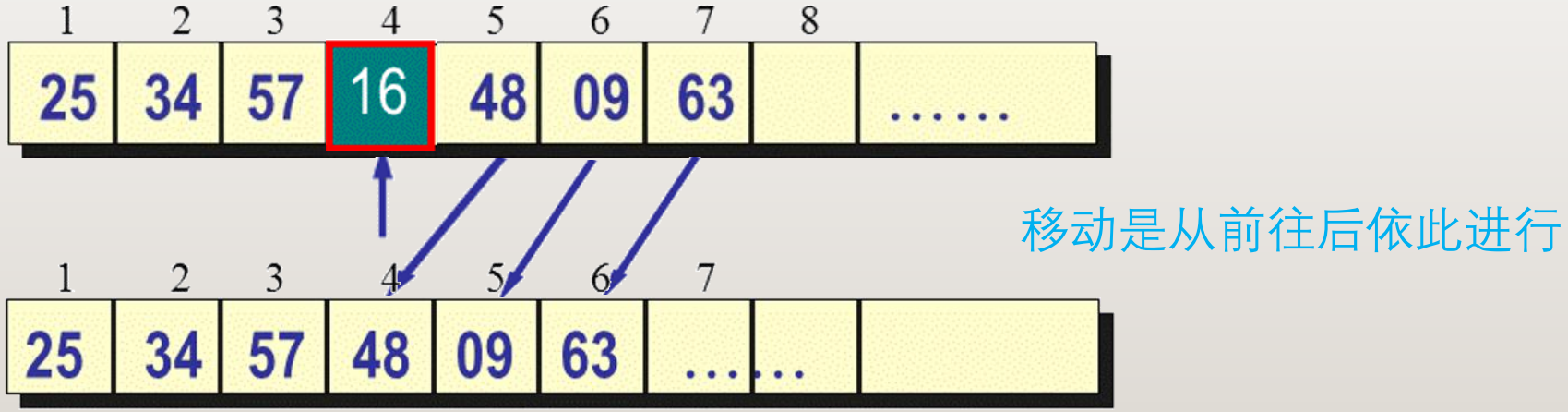
变成长度为n-1的顺序表

$$(a_1, \ldots a_{i-1}, \ a_{i+1}, \ \ldots, \ a_n)$$

第二章 线性表

第二节 顺序表

- 五、顺序表的删除（操作举例）
  - 将第4个元素删除
  - 需要将第5个元素至第n个元素（共7-4）都向前移动一位



第二章 线性表

第二节 顺序表

- 五、顺序表的删除
  - 在顺序表中，删除第i个元素，需要向前移动的元素个数为

    n-i

  - 平均移动元素的操作次数为

$$E_{dl} = \sum_{i=1}^{n} q_i \times (n-i)$$

删除第i个元素的概率

第二章 线性表

第二节 顺序表

- 五、顺序表的删除
  - 当删除位置等概率时，$q_i=1/n$
  - 因此，

$$E_{dl} = \sum_{i=1}^{n} [1/n] \times (n-i) = (n-1)/2$$

顺序表删除操作的时间复杂度为O(n)

第二章 线性表

第二节 顺序表

- 六、顺序表的特点

优点:
访问元素时，可以随机存取
时间复杂度是O(1)

缺点:
插入或删除元素时，需要进行大量的移动操作
时间复杂度是O(n)

顺序存取：就是存取第N个数据时，必须先访问前(N-1)个数据

随机存取：就是存取第N个数据时，不需要访问前(N-1)个数据,直接就可以对第N个数据操作