数据结构与算法

第2节 顺序表

2.1 顺序表的形式

2.1.1 线性表

2.1.1.1 学习目标：

- 了解线性表概念
- 了解线性表的特征
- 掌握顺序表的特征和两种形式

2.1.1.2 什么是线性表？

一个线性表（linear list）是n个数据元素的有限序列：

$$a_0, a_1, ..., a_{i-1}, a_i, a_{i+1}, ..., a_{n-1}.$$

线性表是一种最常用的数据结构,又称线性结构。

分析：

- 数据元素,也称记录，其具体含义在不同的情况下一般不同
- a_0称为起始结点
- a_{n-1}为终止结点
- a_{i-1}称为a_i的前驱
- a_{i+1}称为a_i的后续

线性表的示例：

```
alphabet = (a,b,c,d, ...,z),结点是一个英文字母，  a 就是一个数据元素，这个数据元素只有一个数据项
odd = (1,3,5,7,9)，结点是整型数，3就是一个数据元素
```

学生成绩表

学号	姓名	语文	数学	物理
9001	李梅	91	95	83
9002	张军军	78	89	99
9003	钱龙龙	90	99	80
...	...	...	...	...

上表中每一行作为一个结点.每一个结点有5个域.

2.1.1.3 线性表的特征

在数据元素的集合中:

- 存在唯一的一个首元素。
- 存在唯一的一个末元素 。
- 除首元素外，每个元素均只有唯一的直接前驱元素。
- 除末元素外，每个元素均只有唯一的直接后续元素

2.1.1.4 线性表的基本操作

- 存取 存取表中第i个数据元素a_i
- 插入　在a_i之后插入一个新数据元素
- 删除　删除第i个元素a_i
- 合并　将多个线性表合成一个线性表
- 分解　将一个线性表拆成多个线性表
- 查找　查找满足条件的数据元素
- 排序　对数据元素按照某数据项的值递增或递减的次序重新排列

2.1.2 顺序表

2.1.2.1 什么是顺序表?

在程序中，通常需将一组数据元素作为一个整体来管理和使用;最常用的解决方案是将这一组数据元素看作一个序列。

将线性表中元素依次地存储一组地址**连续**的存储单元中，这种存储结构是顺序结构.　用顺序存储结构的线性表简称为**顺序表**.

分析：

1. 地址连续

2.1.2.2 顺序表的基本形式

1. 数据元素连续存储的顺序表:

数据元素本身连续存储,可用来存储相同类型的数据.

2. 特点：

- 每个数据元素所占的存储空间的大小相同。设每个数据元素需要占用t个存储单元。
- 顺序表存储结构示意：第i个数据元素的存储位置为

$$LOC(a_i) = LOC(a_0) + i * t$$

存储地址	内存状态	元素在顺序表中的位置

存储地址	内存状态	元素在顺序表中的位置
$LOC(a_0)$	a_0	0
$LOC(a_0)+t$	a_1	1
...	...	...
$LOC(a_0)+(i-1)*t$	a_i	i
...	...	...
$LOC(a_0)+(n-1)*t$	a_{n-1}	$n-1$

- 只要有了第一个结点的起始地址，可以按照相同的速度访问线性表中的任意一个结点i.
- 例如： 在32 / 64位系统,整型数据占4个字节， 如果第一个结点的起始地址为0120， 顺序表(100,200,400,800)的存储结构如下：

存储地址	内存状态	元素在顺序表中的位置
0120(地址)	100	0
0124	200	1
0128	400	2
0132	800	3

- 这样的顺序表可用来存储相同类型的数据.

2.1.3 顺序表的形式二：元素外置的顺序表

1. 含义：数据在顺序表之外, 可以存储不同类型的数据

2. 特点：

- 顺序表中各单元保存数据地址,每个地址所需要的存储量相同
- 可以用来保存数据元素的大小不同的数据对象，例如python中列表 li =[1,2,"Mar"]

3. 连续存储形式和元素外置形式的区别：

- 连续存储形式下，每个存储空间所存的是数据本身
- 元素外置形式下，每个存储空间存储的是数据的地址，数据存储在内存的其他位置

2.1.4 为什么需要顺序表？(顺序表的优点)

- 顺序表中的任意数据元素可由公式直接导出，因此顺序表是随机存取的存储结构
- 利用顺序表，无需为表示结点间的逻辑关系增加额外的存储空间；
- 可方便地随机存取表中的任意数据元素(结点)

2.2 顺序表的结构与实现

2.2.0 学习目标

1. 掌握顺序表的结构
2. 了解顺序表的两种实现方式
3. 了解顺序表存储区的扩充方式

2.2.1 顺序表的结构

1. 数据区：用于存储数据本身
2. 表头信息:关于整个顺序表的详情

- 容量
- 元素个数

3.作图：

0100	**8**（容量）
0104	4(元素个数)
0108	20(第1个元素)
0112	40
0116	60
0120	80(第4个元素)
0124	
0128	
0122	
0126	

2.2.2　顺序表的两种实现方式

1. 一体式结构:表头信息与数据区连续存储 如上图所示。

2. 分离式结构:表头信息与数据区分别存储 作图讲解与一体式结构的区别。

0100	**8**
0104	4
0108	0200

0200	**20**
0204	40
0208	60

0200	20
0212	80
0216	
0220	
0224	
0228	

3. 两种实现方式的区别

- 为考虑数据的动态变化，往往用分离式结构
- 若更换数据，一体式结构需整体搬迁,而分离式结构中只需改变表头中链接到数据区的地址

2.2.3　顺序表存储区的扩充

1.扩充固定数目的存储空间

- 优点:节省存储空间
- 缺点:操作往往非常频繁

2.倍增其存储空间

- 优点: 减少操作次数
- 缺点: 浪费存储空间

2.3 顺序表的操作

掌握插入操作和删除操作的时间复杂度

2.3.1 插入元素

1. 插入元素: 在顺序表的第i个数据元素a_i之后插入一个新的数据元素x.

- 时间复杂度：$T=O(n)$
- 插入前，顺序表长度为n,顺序表为

$$(a_0,a_1,...,a_i,a_{i+1},...,a_{n-1})$$

- 插入后，顺序表长度为$n+1$,顺序表为

$$(a_0,a_1,...,a_i,x,a_{i+1},...,a_{n-1})$$

- 顺序表进行插入操作后，$a_{i+1},...,a_{n-1}$向后平移一个存储空间

2. 插入元素的效率：

- 最坏情形,$i=0$,顺序表中结点移动次数是n
- 插入算法的最坏时间复杂度为: $O(n)$
- 最好情形,$i=n-1$,无须移动结点

- 在顺序表末尾插入元素的时间复杂度: $O(1)$

2.3.2 删除元素

1. 顺序表中删除第i个元素a_i,删除前顺序表长度为n,顺序表为

$$(a_0, a_1, ..., a_{i-1}, a_i, a_{i+1}, ..., a_{n-1});$$

删除后，顺序表长度为$n-1$,顺序表为

$$(a_0, a_1, ..., a_{i-1}, a_{i+1}, ..., a_{n-1});$$

- 删除a_i,元素时，必须将第a_{i+1}到第a_{n-1}个元素依次向前移动一个位置。

2. 删除元素的效率

- 最坏情形,$i=0$,顺序表中结点移动次数是$n-1$
- 删除算法的最坏时间复杂度：$T=O(n)$
- 最好情形，$i=n-1$,无须移动结点
- 删除表尾元素: $T=O(1)$

2.4 Python中的列表

2.4.0 学习目标

1. 掌握列表的特征
2. 了解列表的基本实现技术
3. 了解列表的扩充策略

2.4.1 Python列表的特征

1. 基于下标,元素访问和更新非常高效：$T=O(1)$ 所以采用顺序表技术
2. 允许任意加入元素，且对象的标识id保持不变

- 为了让列表对象的标识id保持不变，所以采用分离式技术
- 为了允许加入元素，所以采用动态顺序表.

2.4 2 Python列表的基本实现技术

2.4.2.1 Python列表是采用分离式技术实现的动态顺序表

小结：列表是分离式动态顺序表

2.4.2.2 Python列表的扩充策略

1 建立空列表或很小的列表时，系统分配一块能容纳 8 个元素的存储区

2 执行插入操作时，若元素存储区满，换一块 4 倍大的存储区

3 若表已足够大，则改变策略：存储区满时，就换一块 1 倍大的存储区

2.5 附录

a)什么是位(bit)?

信息量的度量单位，是信息量的最小单位。二进制数的一位包含的信息. 例：表现26个拉丁字母必要的信息量是5比特(16<26<32).

b)什么是字节(Byte)?

信息量的度量单位. `1 Byte = 8 bit` . 例： 除拇指外，双手手指弯曲表示0,伸直表示1，那么所有的信息量，就是1字节.

例： CPU的位是指CPU一次性可处理的数据量. 1字节=8位，32位处理器可以一次性处理4个字节的数据量，依次类推。32位操作系统针对的32位的CPU设计。64位操作系统针对的64位的CPU设计.

c).什么是内存?

计算机内暂时存储程序和数据的地方.

d)计算机如何找到内存中的信息?

计算机只能1个字节1个字节地读取内存中的数据；

这样一个字节的内存区域叫一个存储单元;

计算机为每个存储单元分配一个号码，这就是内存的地址。

内存地址的范围，就是存储地址空间。类比: 为一层楼的房间命名，
0001,0002,0003,0004,0005,0006,0007,0008,内存地址的空间为 9999.

注意事项：

- 区分字节和位的概念： `1 Byte = 8 bit`
- 内存读取和定位的最小单位是字节(Byte)
- 顺序表是线性表的一种，顺序表是一种特殊的线性表：顺序表中的元素存储在连续的存储单元中。

习题：

1. 在一个采用顺序存储方式的线性表中，如果表的第一个元素的存储地址为0100,每一个元素的长度为2,则第4个元素的地址是_____. A. 0110 B. 0108 C. 0106 D. 0120

2. 一个有n个元素的线性表，采用顺序存储方式。若删除第$i(1 \leq i \leq n)$个元素，需要向前移动_____个元素；若在第$i(1 \leq i \leq n)$个元素前插入一个元素，需要向后移动_____个元素。