

实验 1（线性表的物理实现）日志

一、线性表

线性表是由 n ($n \geq 0$) 个数据元素（结点） $a[0]$, $a[1]$, $a[2]$..., $a[n-1]$ 组成的有限序列。

其中：

- 数据元素的个数 n 定义为表的长度 $n = \text{"list"}.length()$ ($\text{"list"}.length()=0$ (表里没有一个元素) 时称为空表)。
- 将非空的线性表 ($n \geq 1$) 记作: ($a[0]$, $a[1]$, $a[2]$, ..., $a[n-1]$)。
- 数据元素 $a[i]$ ($0 \leq i \leq n-1$) 只是个抽象符号, 其具体含义在不同情况下可以不同。

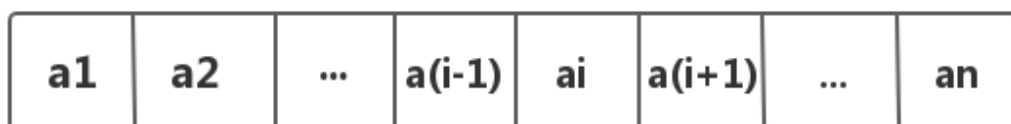
一个数据元素可以由若干个数据项组成。数据元素称为记录, 含有大量记录的线性表又称为文件。这种结构具有下列特点: 存在一个唯一的没有前驱的(头)数据元素; 存在一个唯一的没有后继的(尾)数据元素; 此外, 每一个数据元素均有一个直接前驱和一个直接后继数据元素。

线性表的存储结构分为顺序表和链表。

二、顺序表

顺序存储结构: 用一段地址连续的存储单元依次存储线性表的数据元素。

从定义中我们可以知道这种存储方式存储的数据是连续的, 而且数据类型相同, 所以每一个数据元素占据的存储空间是一致的。



顺序存储结构

顺序表的优缺点:

优点:

1. 逻辑与物理顺序一致, 顺序表能够按照下标直接快速的存取元素。
2. 无须为了表示表中元素之间的逻辑关系而增加额外的存储空间。

缺点:

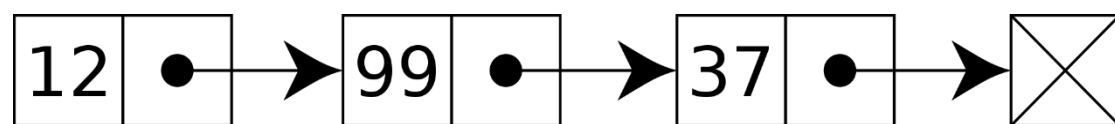
1. 线性表长度需要初始定义, 常常难以确定存储空间的容量, 所以只能以降低效率的代价使用扩容机制。
2. 插入和删除操作需要移动大量的元素, 效率较低。

三、链表

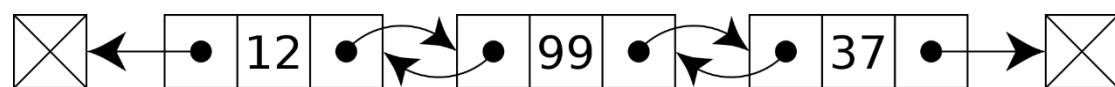
链表（Linked list）是一种常见的基础数据结构，是一种线性表，但是并不会按线性的顺序存储数据，而是在每一个节点里存到下一个节点的指针(Pointer)。由于不必按顺序存储，链表在插入的时候可以达到 $O(1)$ 的复杂度，比另一种线性表顺序表快得多，但是查找一个节点或者访问特定编号的节点则需要 $O(n)$ 的时间，而顺序表相应的时间复杂度分别是 $O(\log n)$ 和 $O(1)$ 。

使用链表结构可以克服数组需要预先知道数据大小的缺点，链表结构可以充分利用计算机内存空间，实现灵活的内存动态管理。但是链表失去了数组随机读取的优点，同时链表由于增加了结点的指针域，空间开销比较大。

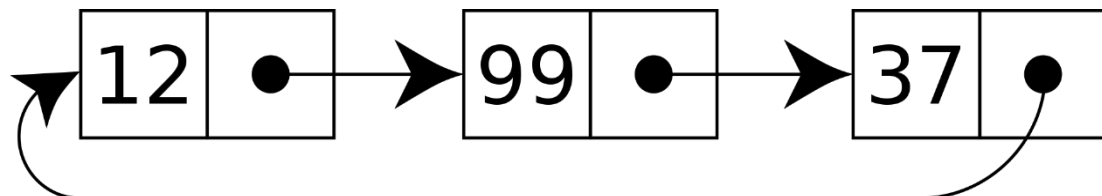
在计算机科学中，链表作为一种基础的数据结构可以用来生成其它类型的数据结构。链表通常由一连串节点组成，每个节点包含任意的实例数据（data fields）和一或两个用来指向上一个/或下一个节点的位置的链接（"links"）。链表最明显的好处就是，常规数组排列关联项目的方式可能不同于这些数据项目在记忆体或磁盘上顺序，数据的访问往往要在不同的排列顺序中转换。而链表是一种自我指示数据类型，因为它包含指向另一个相同类型的数据的指针（链接）。链表允许插入和移除表上任意位置上的节点，但是不允许随机存取。链表有很多种不同的类型：单向链表，双向链表以及循环链表。



一个单向链表包含两个值：当前节点的值和一个指向下一个节点的链接



一个双向链表包含三个值：数值，向后的节点链接，向前的节点链接



用单向链表构建的循环链表

链表的优缺点：

1. 优点：

链表的主要优势有两点：一是插入及删除操作的时间复杂度为 $O(1)$ ；二是可以动态改变大小。

2. 缺点：

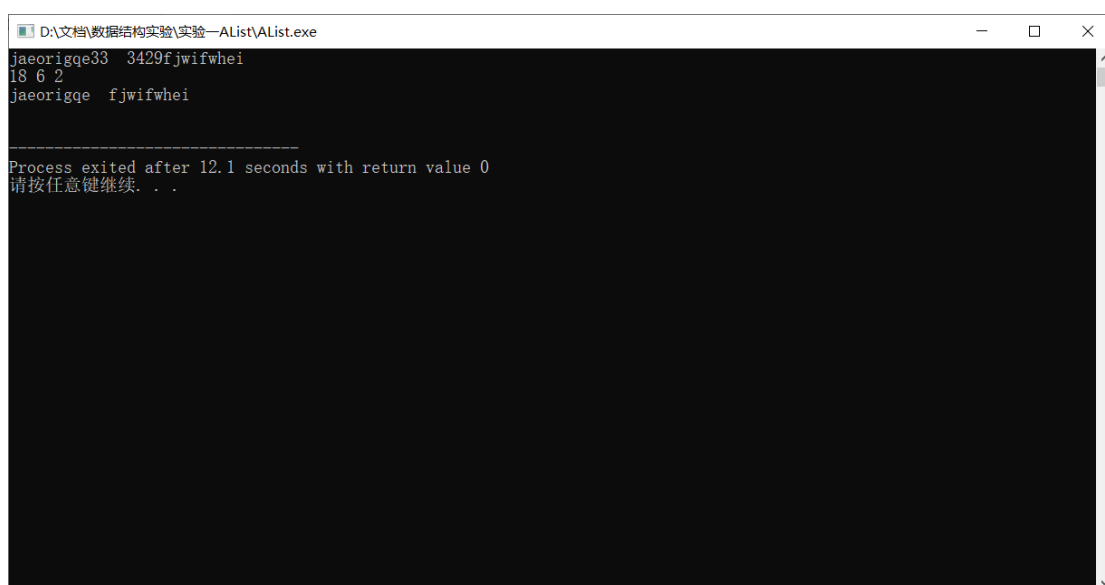
由于其链式存储的特性，链表不具备良好的空间局部性，也就是说，链表是一种缓存不友好的数据结构。

四、遇到的问题及解决方法

1. 在编写“List.h”头文件时未加入条件编译`#ifndef/#define/#endif`，发生报错。

在 c 语言中，对同一个变量或者函数进行多次声明是不会报错的。所以如果 h 文件里只是进行了声明工作，即使不使用`#ifndef`宏定义，一个 c 文件多次包含同一个 h 文件也不会报错。使用`#ifndef`可以避免下面这种错误：如果在 h 文件中定义了全局变量，一个 c 文件包含同一个 h 文件多次，如果不加`#ifndef`宏定义，会出现变量重复定义的错误；如果加了`#ifndef`，则不会出现这种错。

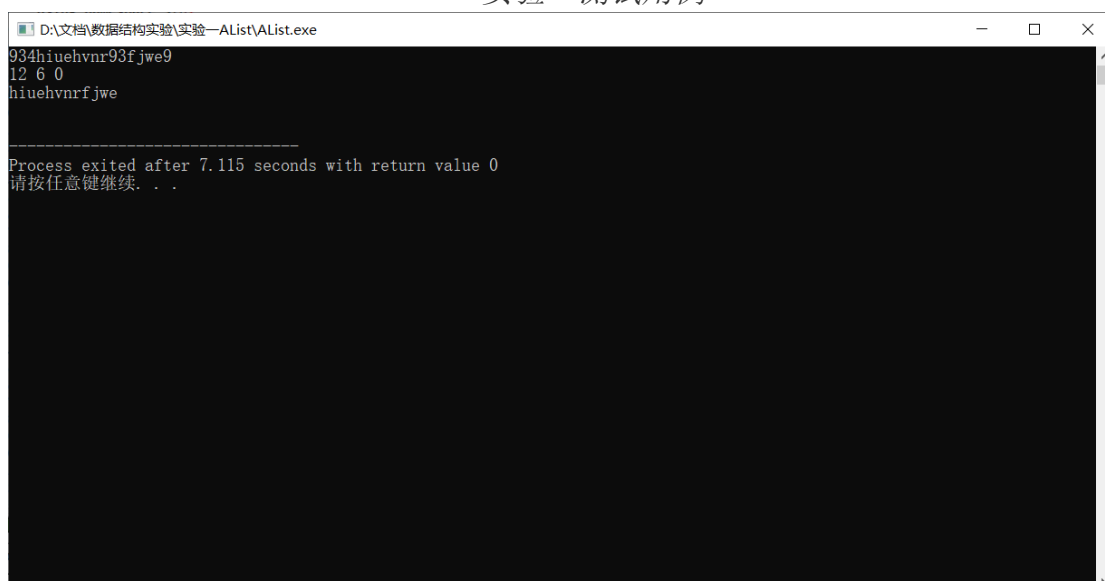
五、测试用例



```
D:\文档\数据结构实验\实验一\AList\AList.exe
jaeorigqe33 3429fjwifwhei
18 6 2
jaeorigqe fjwifwhei

-----
Process exited after 12.1 seconds with return value 0
请按任意键继续. . .
```

实验一测试用例



```
D:\文档\数据结构实验\实验一\AList\AList.exe
934hiuehvn93fjwe9
12 6 0
hiuehvnrfjwe

-----
Process exited after 7.115 seconds with return value 0
请按任意键继续. . .
```

实验一测试用例

```
D:\文档\数据结构实验\实验一\AList\AList.exe
kfowi49584&*~&()okewr3
10 6 6
kfowi&*~&()okewr

-----
Process exited after 22.37 seconds with return value 0
请按任意键继续. . .
```

实验一测试用例

```
D:\文档\数据结构实验\实验一\AList\AListTest.exe
*****
*顺序表中各个基本操作演示*
*****
向顺序表中依次添加元素12、15、9、48、55
添加元素后顺序表中内容为:
12 15 9 48 55

当前顺序表长度为:
5

查看当前curr的值为:
4

设置线性表中栅栏curr的位置为2
并在当前位置curr插入元素4
插入元素后顺序表中内容为:
12 15 4 9 48 55

设置线性表中栅栏curr的位置为4
并删除当前位置curr处的元素
删除元素后顺序表中的内容为:
12 15 4 9 55

-----
Process exited after 0.04066 seconds with return value 0
请按任意键继续. . .
```

顺序表基本操作实现

```
D:\文档\数据结构实验\实验一\List\ListTest.exe
*****
*单链表中各个基本操作演示*
*****

向单链表中依次添加元素12、15、9、48、55
添加元素后单链表中内容为:
12 15 9 48 55

当前链表长度为:
5

查看当前curr的值为:
4

设置线性表中栅栏curr的位置为2
并在当前位置curr插入元素4
插入元素后单链表中内容为:
12 15 4 9 48 55

设置线性表中栅栏curr的位置为4
并删除当前位置curr处的元素
删除元素后单链表中内容为:
12 15 4 9 55

-----
Process exited after 0.04769 seconds with return value 0
请按任意键继续. . .
```

单向链表基本操作实现

六、总结

对于线性链表和顺序表都属于线性表问题，但是线性链表的插入删除比顺序表要简单，方便。顺序表的查找比较方便，线性链表做元素寻找时，必须从头结点开始寻找。各有各的优缺点。

经过这次实验，我对于线性表的顺序结构的相关代码已基本熟悉，算法知识得到了复习与巩固。在写代码的过程与调试中，在解决问题过程中，丰富了个人编程的经历和经验，提高了个人解决问题的能力。

数据结构中的线性表的实验很多知识都是以前 C++ 语言学过的，只是将这些知识进行了一些结构化，使程序的结构看起来更加合理，方便代码管理。另外，在使用链表时指针的使用是最值得我们注意的地方，它需要我们在逻辑顺序上不能出错，否则就会造成错误，很多时候编译器识别不到，只有在运行时才会出现内存错误的提示，调试起来也不是那么明显。所以指针是我应该重点掌握的地方，只有学好了指针，才可能学好 C++。总之今后在实验过程中要更加注重这些基本的操作。由于这段时间事情比较多，故在实验上未花很多的精力，用了两天把线性表两种实现方法做了，有些不尽人意的地方也没有加以修正，但是实验的过程之中我又对 C++ 的知识重新温习了一遍，获益匪浅。我希望能够做更多这样的实验，这样我才能发现自己在实际操作中的不足并加以改正。

2020.10.24

杨杰