# 数据结构和算法

作者：小甲鱼

让编程改变世界

Change the world by program

# 单链表的整表创建

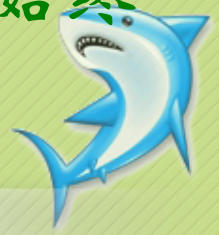
- 对于顺序存储结构的线性表的整表创建，我们可以用数组的初始化来直观理解。
- 而单链表和顺序存储结构就不一样了，它不像顺序存储结构数据这么集中，它的数据可以是分散在内存各个角落的，他的增长也是动态的。
- 对于每个链表来说，它所占用空间的大小和位置是不需要预先分配划定的，可以根据系统的情况和实际的需求即时生成。
- 人生就要追求向单链表一样，灵活应变！

# 单链表的整表创建

- 创建单链表的过程是一个动态生成链表的过程，从“空表”的初始状态起，依次建立各元素结点并逐个插入链表。
- 所以单链表整表创建的算法思路如下：
  - 声明一结点p和计数器变量i；
  - 初始化一空链表L；
  - 让L的头结点的指针指向NULL，即建立一个带头结点的单链表；
  - 循环实现后继结点的赋值和插入。

# 头插法建立单链表

- 头插法从一个空表开始，生成新结点，读取数据存放到新结点的数据域中，然后将新结点插入到当前链表的表头上，直到结束为止。
- 简单来说，就是把新加进的元素放在表头后的第一个位置：
  - 先让新节点的next指向头节点之后
  - 然后让表头的next指向新节点
- 嗯，用现实环境模拟的话就是插队的方法，始终让新结点插在第一的位置。



# 动画演示

# 头插法建立单链表

- 我们说好的代码呢？
  - CreateListHead.c

# 尾插法建立单链表

- 头插法建立链表虽然算法简单，但生成的链表中结点的次序和输入的顺序相反。
- 就像现实社会我们鄙视插队不遵守纪律的孩子，那编程中我们也可以不这么干，我们可以把思维逆过来：把新结点都插入到最后，这种算法称之为尾插法。（小甲鱼给这个算法想到一个容易记住的艺名，叫“菊花”）
- 好，那我们接下来就结合欢乐的动画一起来理解理解菊花的内涵吧~

# 动画演示

# 尾插法建立单链表

- 我们说好的代码呢？
  - CreateListTail.c