



## 第十单元：链表

教学内容	链表,用 typedef 定义类型
教学目标	
应知	<ul style="list-style-type: none"><li>了解链表的结构,链表基本操作的含义</li><li>能够写出 C 语言对链表节点的结构描述</li><li>能够写出有关链表操作的关键实现语句</li></ul>
应会	<ul style="list-style-type: none"><li>有关链表操作的关键实现语句:创建,插入,删除,输出</li></ul>
难点	<ul style="list-style-type: none"><li>链表的结构特点和操作处理</li></ul>

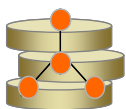
### 1. 专业英语词汇

英文词汇	中文名
create	创建
insert	插入
delete	删除
modify	修改

### 教学方法

- 形象比喻,注重思路引导

### 教学过程



#### 1. 回答下列问题:

- 什么是链表?
- 为什么使用链表?
- 链表的数据结构有何特点
- 链表结构同数组相比有什么优点?
- 链表将涉及到我们之前学过的什么内容?又有什么新内容?
- 静态链表与动态链表有什么区别?

#### 2. 动态链表的操作型

- 假设一个链表由 5 个节点构成,节点数据是学生的学号和姓名(链表按学生学号的升序排列),则此链表在内存中的存储方式示意图如下



该节点地址	学生学号	学生姓名	节点中的 next 指针
2000H	9911104	Alice	202BH
2010H	9911129	Tom	NULL
2020H	9911118	Jordan	2010H
202BH	9911105	Allen	201AH
201AH	9911108	Rodman	2020H

9911104→9911105→9911108→9911118→9911129

### 3. 创建动态链表

**问题：**怎样输入节点

怎样使节点相链

**思路：**

- ① p1 指向新建结点
- ② p2 = p1; 使 p2 指向新节点, 即 p2 指向链表中最后一个节点
- ③ p1 再指向更新的节点
- ④ p1 = p2 -> next; 使旧节点链上新节点
- ⑤ p2 = p1; 使 p2 指向新节点, 即 p2 指向链表中最后一个节点, 再到第③步, 如此循环输入

### 4. 删除结点

**问题：**怎样找到将要删除的节点

如何实现节点删除

**思路：**

- ① 假设有 a, b, c, d, e 五个节点, 要删的节点是 s
- ② 在链表中寻找要删的节点, 先使 p1, p2 同指头节点, 如果不是目标, p1 后移
- ③ 如仍然不是目标, p2=p1, p2 也已过来
- ④ p1 再指向下一个节点, p1 = p1 -> next
- ⑤ 如果发现是目标节点, 则删除, p2 -> next = p1 -> next, 表示删除节点 p1. (p2 是 p1 的前一个节点)

### 5. 插入结点

**问题：**怎样找到插入位置

怎样实现插入

**思路：**



- ① 假设  $p_0$  为待插入节点, 先使  $p_1, p_2$  都指向头节点, 比较  $p_1 \rightarrow \text{num}$  和  $p_0 \rightarrow \text{num}$ , 寻找待插入位置, 如果不是目标, 使  $p_1$  指向下一个节点, 再比较  $p_1 \rightarrow \text{num}$  和  $p_0 \rightarrow \text{num}$
- ② 如果不是目标位置, 使  $p_2$  指向刚比过的  $p_1$ ,  $p_2 = p_1$
- ③  $p_1$  再指向下一节点,  $p_1 = p_1 \rightarrow \text{next}$ , 再比较
- ④ 找到目标位置, 插入  $p_0$ , 两条语句  $p_2 \rightarrow \text{next} = p_0$ ;  $p_0 \rightarrow \text{next} = p_1$ ; 表示  $p_0$  插入到  $p_2$  和  $p_1$  之间 ( $p_2$  是  $p_1$  的前一个节点)
- ⑤ 如插入到表头:  $\text{head} = p_0$ ;  $p_0 \rightarrow \text{next} = p_1$ ;
- ⑥ 如插入到表尾:  $p_1 \rightarrow \text{next} = p_0$ ;  $p_0 \rightarrow \text{next} = \text{NULL}$ ;
- ⑦ 如插入到空表:  $\text{head} = p_0$ ;  $p_0 \rightarrow \text{next} = \text{NULL}$ ;

## 6. 链表输出

**思路:** 从头节点开始 ( $p = \text{head}$ )

打印完指向下一个节点 ( $p = p \rightarrow \text{next}$ )

直到节点为  $\text{NULL}$



### 学生容易出错的地方

- ❏ 创建链表时每个新节点都应该申请空间
- ❏ 插入、删除操作时的指针移位理解不透
- ❏ 最后一个节点的  $\text{next}$  指针赋值为  $\text{NULL}$

## 问题与讨论

- ❏ 指向结构变量的指针如何定义和使用?
- ❏ 链表的定义和常用操作?
- ❏ 链表节点在 C 语言中如何描述?
- ❏ 如何申请和释放节点空间?

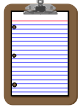
## 小结 (可由问题与讨论方式给出)



- ❏ 链表是一种动态存储结构, 可以按需随时分配存储空间



## 课后任务



整理课堂笔记



尝试调通链表程序