**数据结构实验报告**

姓名： 粟锦 学号：U201817053 班级：软工1803班

**实验四**

**实验描述**

1、Radix Sort。实现桶式排序和基于桶式排序的基数排序。在基数B，数组长度n和最大元素值m中，对排序时间影响最大的是哪一个？元素在未排序数组中的顺序是否对时间复杂度有影响？设计试验证明你的想法

2、Stack。用C语言设计堆栈，并实现中缀表达式到后缀表达式的转换

**Radix Sort：**

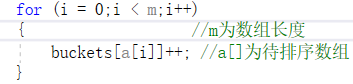
**桶式排序：**

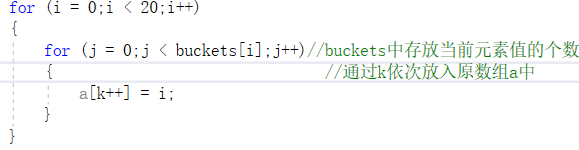
**分析：桶式排序是一种特殊的排序方式，首先假设需要排列的所有元素的值都在一个范围内，假设为0~m，然后建立编号为0~m的桶（数组），再对需要排列的数组进行一次遍历，将元素的值放入对应编号的桶中。在对数组进行完一次遍历后，再对桶进行一次遍历，按顺序取出桶里的元素放入原数组中即可。**

**实现过程：**

首先假设桶的个数为20，即排序的元素大小在0~19之间。 IMG_256

然后对需要排序的数组的元素进行遍历，将每个元素放入与其值对应的桶中。



最后再对桶进行遍历，根据每个桶中元素的个数，直接依次放入原数组即可。

**时间复杂度分析：**

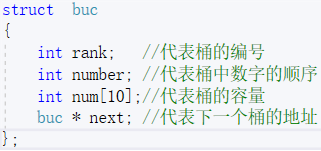
由于桶式排序只需要分别对数组以及桶各进行一次遍历，所以时间复杂度为O(m+n)，A其中m为桶的个数，n为数组元素的个数。然而桶式排序在元素的数值非常大时就会占用很高的空间复杂度，因此要根据情况选择使用。

**基数排序：**

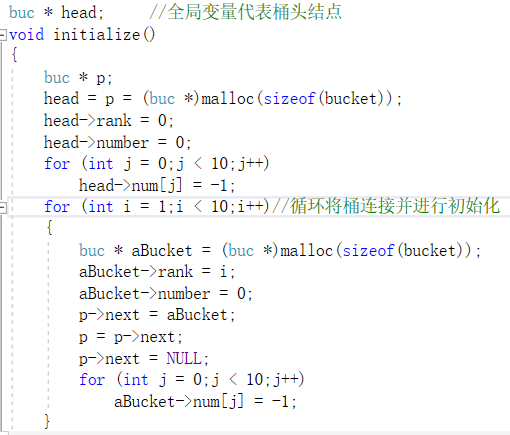
**分析：首先根据选择的进制设置相应个数的桶（假设使用十进制，则设置0~9编号的十个桶），将10个桶以链表的形式连接起来。然后对数组进行遍历。我们采取低位优先的方式，首先对元素的个位进行分析，将元素分别放入与其个位相对应的桶中。进行完一次遍历后，再根据先入先出原则， 将0~9号桶中的元素依次放回原数组。第二次遍历时，比较的是元素的十位并进行相同操作。直到对所有元素中的最高位进行完一次遍历操作后，将桶中的元素依次放回原数组便得到了排序后的数组。**

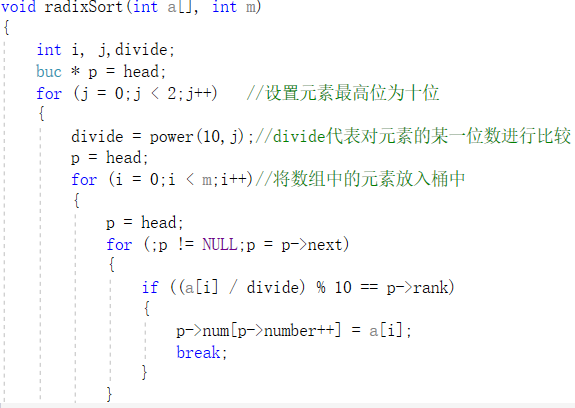
**实现过程：**

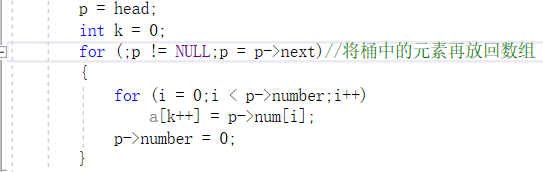
首先先定义好一个桶。



然后将桶进行初始化，设置桶的各种参数以及将桶连接起来。



最后便进行基于桶式排序的基数排序 



**时间复杂度分析：**

我们假设选取的基数是B，数组的总长度为n，其中最大元素的位数是m。可以知道，在对元素某一位数进行入桶出桶的操作时，需要对数组和桶各进行一次循环遍历。在寻找与元素位数匹配的桶时，最坏情况是始终匹配到最后一个桶，因此入桶的时间复杂度是O(Bn)。出桶时循环的次数等于数组元素的个数，所以出桶的时间复杂度是O(n)。

由此可知，对元素某一位数进行入桶出桶的操作需要的时间复杂度是O((B+1)n)。由于最大元素的位数是m，因此我们一共需要进行m次入桶出桶操作，所以时间复杂度是O((B+1)mn)。

因为在实际排序过程中，选择的基数以及最大元素的位数都很小，可近似看为常数，因此时间复杂度可简化为O(n)，即数组元素的个数对排序时间的影响最大。

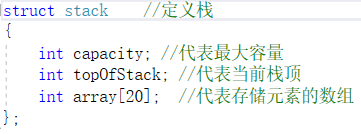
**Stack：**

**实现堆栈：**

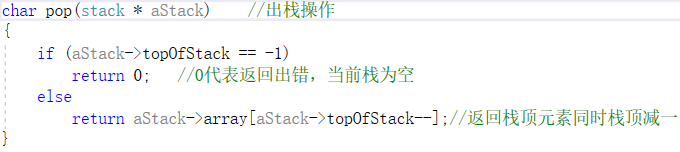
**分析:我们用数组来实现堆栈，定义一个结构体，其中包含三个变量（topOfStack,capacity,array）分别代表当前栈的顶部、栈的最大容量以及存储栈元素的数组。并根据结构体来定义一些关于栈的操作（pop、push）**

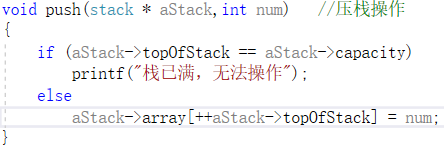
**实现过程：**

首先定义一个栈的结构体：



然后定义关于栈的压栈、出栈操作：





如此，便已实现用数组进行栈操作。

**中缀表达式转后缀表达式：**

**分析:**

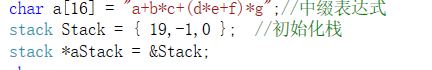
**创建一个上述的结构体，通过对中缀表达式的遍历，对遍历到的字符进行判断并进行不同的处理：**

1. **数字，直接输出**
2. **“(”，压栈**
3. **“)”,将栈中的所有元素依次弹出直到弹出“(”。**
4. **“+”、“\*”,依次弹出栈内元素直到找到一个比该符号优先级低的符号后再将该符号压栈。但不弹出“(”。 优先级：“+”<“\*”<“)”**

**当表达式遍历完后，输出栈中的全部元素，得到的便是后缀表达式。**

**实现过程：**

首先创建一个结构体并进行初始化：



然后对中缀表达式进行遍历判断即可：