# 数据结构回文串实验报告

**班级：2015211306 姓名：李航 学号：2015211284 分工：**

**班级：2015211306 姓名：魏晓 学号：2015211301 分工：**

1. **需求分析**

**用栈和队列的不同属性来实现判断回文串，这段程序的主要目的是用栈存储用队列存储，根据不同的出栈规则，实现回文串的比较，输入的是一串字符，并且用#做结尾，输出，当输入是回文串时输出true，不是的输出false，程序可以判断输入是否为回文串，正确的输入不能包含#，只能在结尾判断是不是到终点。**

1. **概要设计**

**要实现对回文串的判断，首先让输入用相同的方式存进不同的存储区，然后用不同的操作对存储在栈和队列中的字符串进行比对，如果有一次没有比对成功则自动退出并输出false，如果所有比对都通过那么输出true**

**用到两个主要的结构：**

**struct stak{**

**char \*top;**

**char \*base;**

**};**

**struct que{**

**char \*base;**

**char \*top;**

**};**

**这是两个结构分别是栈存储和队列存储，首先输入字符串，然后把他们分别存储在不同区域，在进行不同的比对，释放申请的空间**

1. **详细设计**

**因为定义了不同的存储方式，设计到栈操作，队列操作，**

**cin>>bbs;**

**while(bbs!='#')**

**{**

**\*(s.top)=bbs;**

**++s.top;**

**\*(q.top)=bbs;**

**++q.top;**

**cin>>bbs;**

**}**

**s.top--;**

**q.top--;**

**最重要的就是最后的s.top--，只有这样，最后新的s.top才能指向正确的值，否则程序的运行结果一直是false，是由循环结构决定的，也可以采取另外一中方式就是先申请再赋值就不会出现这种状况**

**while(q.top!=q.base)**

**{**

**if((\*s.top)!=(\*q.base))**

**{**

**delete(s.top);**

**delete(q.top);**

**cout << "False!" << endl;**

**return 0;**

**q.base++;**

**}**

**else**

**{**

**q.base++;**

**s.top--;**

**}**

**}**

**这一段程序实现了字符串的比对，这段程序利用不同的出战方式，挨个对比但是这个程序的时间复杂度有点高，主要就是因为当比对的达到一半时，实际上比对已经完成，后面比对毫无意义比对完一个字符就释放对应的空间，**

**如果检测到有字符不符合就直接退出，**

**4．调试分析报告**

**调试的过程依然是输入部分，由于输入的结构，所以在退出循环必须把加的一个是s.top释放，否则整个比对就会完全错误，注意new空间以后delete否则内存不久就会崩溃，**

**5.用户使用说明**

**打开exe文件直接输入想要判断的字符，并以#结尾程序就能自动根据输入判断是不是回文串**

**6.测试结果**

**输入：qwreerwq# 输出：True**

**输入：we# 输出：false**

**输入：wqerqwerqwer# 输出：false**