第三周实验报告

沈家成

2017 年 10 月 11 日

1　1205 Ackerman

1.1　主要难点

Ackerman 函数用递归是很容易实现的，但是要非递归化，就需要借助栈的思想。

1.2　解决方法

在手动计算的过程中，发现每次都是要提取出最后两个参数，根据情况，返回三个、两个或者一个参数，直到最后只剩下一个参数，就是最后的答案。因此，可以借用栈先进后出的特性。

刚开始先把要计算的 m,n 压栈，表示当前任务是计算 Ackerman(m,n)，然后弹出两个元素，表示要执行任务。根据 Ackerman 函数的定义，分解为新任务，再把新任务的参数压栈。这样循环往复，直到只有一个参数，就是最终的答案了。

2 1206 Pascal

2.1 主要难点

else 是可选的，这意味着即使 if-then 后面没有 else，仍然要算作正确的。在利用栈的结构检查匹配 begin end 匹配的时候，需要忽略掉还在栈顶的 if-then。

2.2 解决方法

2.2.1 if-then-else 配对

因为 if-then 是绑定在一起的，所以先把 if 压栈，如果遇到 then，就弹出栈顶检查，如果是 if，就把 then 压栈，表示这个 if 已经配对过 then 了。之后如果碰到 else，就弹栈检查是否为 then。

2.2.2 begin-end 配对

因为 else 是可选的，就会出现多余的 then 存在栈中，因此需要忽略 then 来进行 begin-end 配对。通过不断的弹栈，直到弹出 begin 或者栈空才结束。如果弹出 begin 就配对成功，如果栈空，就说明没有 begin 可供配对，报错。

2.3 小结

因为 else 和 if-then 不是严格的配对，所以在检查其他配对的时候，需要忽略这些。

3 1570 100number

3.1 主要难点

数据量很大,最多会有 100,000 个数字,每个数字最大会达到 2,000,000,000。这就需要降低算法的时间复杂度，否则很容易超时。

3.2 解决方法

题目中提到了数字已被排序，可以使用时间复杂度为 $O(logN)$ 的牛顿二分法，来查找二哥喜欢的数字在数列中的位置，进而得出比它大的数字个数。

3.3 小结

有序的数据能帮助算法降低时间复杂度。

4 4009 步步为赢

4.1 主要难点

在分割，重组数组的时候，容易混淆变量的含义、数字的位置。

4.2 解决方法

手动列出一步一步的过程，再照此过程编程实现。