Java方向编程题答案

day38

[编程题]24979-最近公共祖先

链接：https://www.nowcoder.com/questionTerminal/70e00e490b454006976c1fdf47f155d9

【题目解析】：

这是一道很经典的考察二叉树的问题，题目本身不难，很常规的考察公共祖先问题，使用递归可以很简单的解决。但是请做完的同学再想想有没有别的解法?

【解题思路】：

满二叉树的子节点与父节点之间的关系为root = child / 2 **注意这里为什么不是(child-1)/2 因为题目中根节点编号有意设置为1了**

利用这个关系，如果a ！ = b，就让其中的较大数除以2， 如此循环知道a == b， 即是原来两个数的最近公共祖先

这种思路简洁高效，希望每个童鞋都能掌握

【示例代码】：

```
import java.util.*;
public class LCA {
    public int getLCA(int a, int b) {
        while (a != b) {
            if (a > b) {
                a /= 2;
            }else {
                b /= 2;
            }
        }
        return a;
    }
}
```

[编程题]36935-空格替换

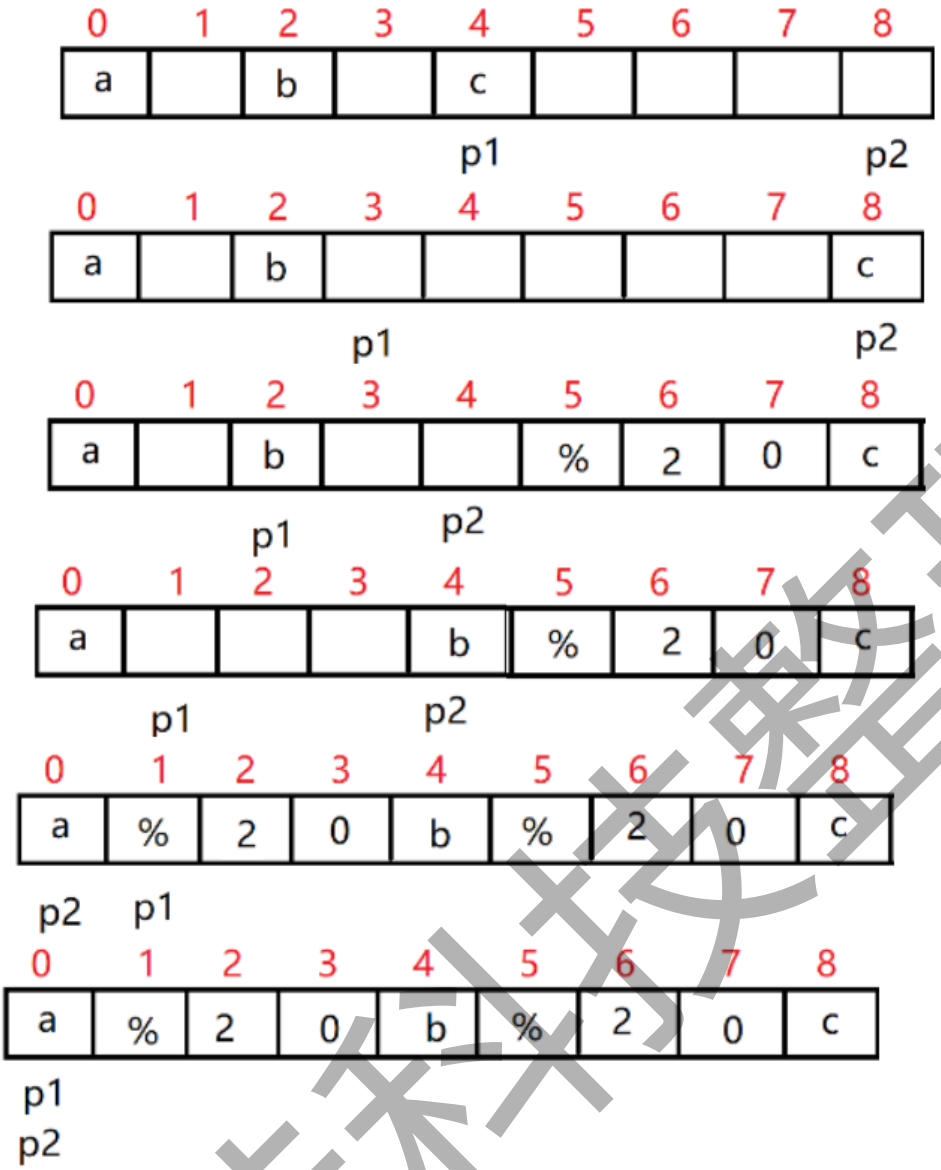
https://www.nowcoder.com/questionTerminal/b0850698cb41449188344cdb647f3e99

【题目解析】： 空格替换是一道字符串考题中经常会出现的题型，这道题比如：a b c d。加%20之后变为：a%20b%20c%20d; 我们会发现，原来的空格1个字符，变成%20了，3个字符。如果原来长度为:字符个数+空格数=4+3 那么处理后大小变为:字符个数+空格数+空格数乘以2 即： 4+3+3*2=13。

【解题思路】：

思路一：我们可以使用StringBuilder，当碰到空格，我们就直接追加一个%20。

思路二：遍历原有的字符串，计算出新的添加了%20的数组的长度。计算方式：见上面【题目解析】。把原来的字符全部拷贝到新的数组ch2当中，然后我们从后往前去遍历ch2,不是空格，向后拷贝。遇到空格手动添加%20.具体的步骤，童鞋们看着示例2的代码，**进行画图理解。因为一般在现场面试的时候，面试官肯定会让你写这种方法来考验你。**



【示例代码一】：

```
import java.util.*;
public class Replacement {
    public String replaceSpace(String iniString, int length) {
        // 如果允许分配额外空间
        if (iniString == null || iniString.length() <= 0) {
            return iniString;
        }
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < length; i++) {
            char c = iniString.charAt(i);
            if (c == ' ')
                sb.append("%20");
            else
                sb.append(c);
        }
        return sb.toString();
    }
```

```
}
```

【示例代码二】：

```
import java.util.*;
public class Replacement {
    public String replaceSpace(String iniString, int length) {
        //字符串生成字符数组
        char[] ch = iniString.toCharArray();

        if(iniString == null || length <= 0) {
            return null;
        }

        int mLen = 0;//计算字符的个数
        int numBlank = 0;//计算空格的数组
        int i = 0;//从0号下标开始遍历

        while(i < ch.length) {
            ++mLen;
            if(ch[i] == ' ') {
                ++numBlank;
            }
            ++i;
        }
        //替换为%20之后，新数组的长度。
        int newlen = mLen + numBlank * 2;//a%20b%20c%20d a b c d
        //新的数组
        char []ch2 = new char[newlen];
        //先把ch里面的内容，全部拷贝到ch2,我们要在ch2里面进行操作。
        System.arraycopy(ch,0,ch2,0,length);

        int indexofMLen = mLen-1;//也必须减一
        int indexofnew = newlen-1;//不减一就越界了
        while(indexofnew > indexofMLen && indexofMLen >= 0)
        {
            if(ch2[indexofMLen] == ' ')
            {
                ch2[indexofnew--] = '0';
                ch2[indexofnew--] = '2';
                ch2[indexofnew--] = '%';
            }
            else
            {
                ch2[indexofnew] = ch2[indexofMLen];
                indexofnew--;
            }
            --indexofMLen;
        }
        return  String.copyValueOf(ch2);
    }
}
```