Rules:

共有两道程序题，**第一题为基础题100分，第二题为附加题20分**。两题都不需要考虑异常输入。

1. 同分情况下，尽早提交答案者排名靠前。允许多次提交答案，以最后一次答案为准。
2. 提交说明：在提交答案的邮件正文注明编程语言（Python需注明版本）和运行环境（包括操作系统信息）
3. C/C++ 程序的编译运行方式：将会用 g++ -O2 进行编译，(version g++ 4.x)
4. Java 程序的运行方式：javac  HelloJava.java ; java HelloJava
5. 程序执行限时：如果题目没有特殊要求，程序运行时间限时1秒
6. 如果题目没有特殊要求，程序的所有输入通过标准输入获得。
7. 如果题目没有特殊要求，运行结果通过标准输出打印。
8. 不得输出任何题目没有要求的输出信息（比如log, 提示信息 "input:" "Succeed!" 等等）。输出多余信息会影响case自动执行通过率，从而影响成绩。
9. 每个test case全对算对，没有半对的说法。Case超时算错。

## Question 1版本号比大小

时限 1s

**【问题描述】**

   判断任意两个版本号的大小。数字越大代表版本越新。

**【输入格式】**

输入为两个版本号，以空格隔开。版本号由数字和“.”组成 (“.”不是必须出现的)。每个子版本号的取值0到999，版本层级数1到20。例如，可能存在版本号的形式有：“2”或者“2.0.5”或者“0.0.1”等等。

**【输出格式】**

标准输出一个整数。如果版本1>版本2输出1，版本1<版本2,输出-1，其他0。

【样例1输入】

1.2.5 2.7

【输出】

-1

【样例2输入】

1.0.4 1.0.0.1

【输出】

1

【样例3输入】

02.10.5 002.2.2

【输出】

1

【解题思路】

首先使两个字符长度一样，长度少的那个版本号用0补齐。

然后开始对比同级的数字大小，大返回1，小返回-1，如果相同，继续比较下一级，如果全部相同返回0。

**Question 2 消消乐**

时限 1s

**【问题描述】**

有一排各种颜色的宝石，数目在1-100个，可以通过碰触来消除。

当同一种颜色的k(k>=1)个宝石紧挨着的时候，可以一次消除并且或得k\*k分奖励。

求可能得到的最高得分。

**【输入格式】**

标准输入一个字符串，内容是多个以逗号分隔的大写字母，代表不同颜色的宝石

**【输出格式】**

标准输出一个整数代表消除完获得的最高得分

【样例输入】

A,C,B,B,B,C,D,C,A

【样例输出】

23

【样例解释】

B,B,B 🡪 A,C,C,D,C,A 得到3\*3=9分

D 🡪 A,C,C,C,A 得到1\*1=1分

C,C,C 🡪 A,A 得到3\*3=9分

A,A 🡪 结束 得到2\*2=4分

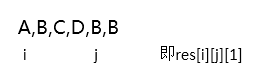


【解题思路】

动态规划的算法思想。

1. **建模**。设函数dp(i,j,k)用来求一串宝石的最高得分，其中i是左边界，j是右边界，k是和右边界相等的宝石数目，用k\*k来计算消除相同宝石所得分。为减少重复计算，用数组res[i][j][k]来记录这个最高得分。

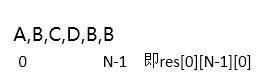
举例来说，A, B, C, D, B, B就可以写成:



或者：



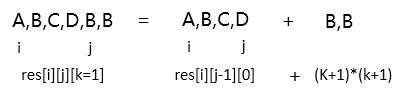
所以，对于长度为N的字符串来说，最终就要求dp(0,N-1,0):



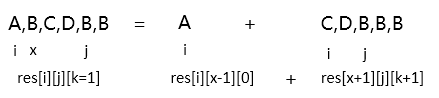
1. **寻找子问题状态转移规律: 进行消除面临两种选择：**

一是直接对右边界k个相同的宝石进行消除：

res[i][j][k] = dp(i,j-1,0) + (1+k)\*\*2,



或者从左往右找到和右边界相同的宝石x，拆分成两个数组再消除：



最终递推公式即选择这两种方法得分最高的那个：

res[i][j][k] = max(res[i][j][k], dp(i,x-1,0) + dp(x+1,j,k+1))

1. **递归终止条件**是: i>j return 0