# 湖南省第八届大学生计算机程序设计竞赛

The Eighth Hunan Collegiate Programming Contest

主办：湖南省教育厅

协办：湖南省高等教育学会计算机教育专业委员会

承办：湖南工业大学

2012 年 10 月 13 日

本次比赛 12 道题目，共 14 页。如有缺页，请立即通知赛场工作人员。

所有题目均采用标准输入输出，请不要读写任何文件。
所有题目的正确输出均是惟一的。你的输出只有和正确输出完全一致时才能通过。

# 题目 A. 三家人

有三户人家共拥有一作花园，每户人家的太太均需帮忙整理花园。A 太太工作了 5 天，B 太太则工作了 4 天，才将花园整理完毕。C 太太因为正身怀六甲无法加入她们的行列，便出了 90 元。请问这笔钱如何分给 A、B 二位太太较为恰当？A 应得多少元？

90/(5+4)*5=$50 元？如果这么想你就上当了！正确答案是 60 元。如果没想通的话再想想吧。下面回答一个一般性的问题：假定 A 太太工作了 x 天，B 太太工作了 y 天，C 太太除了 90 元，则 A 太太应得多少元？输入保证二位太太均应得到非负整数元钱。

**友情提示：**本题有个小小的陷阱哦。如果答案错的话，认真检查一下代码吧。

## 输入格式

输入第一行为数据组数 $T$($T$<=20)。每组数据仅一行，包含三个整数 $x, y, z$ (1<=$x, y$<=10, 1<=$z$<=1000)。

## 输出格式

对于每组数据，输出一个整数，即 A 太太应得的金额（单位：元）。

| 样例输入 | 样例输出 |
| --- | --- |
| 2<br>5 4 90<br>8 4 123 | 60<br>123 |

# 题目 B. 机器人的指令

数轴原点有一个机器人。该机器人将执行一系列指令，你的任务是预测所有指令执行完毕之后它的位置。

- LEFT：往左移动一个单位
- RIGHT: 往右移动一个单位
- SAME AS *i*: 和第 i 条执行相同的动作。输入保证 *i* 是一个正整数，且不超过之前执行指令数

## 输入格式

输入第一行为数据组数 *T*(*T*<=100)。每组数据第一行为整数 *n* (1<=*n*<=100)，即指令条数。以下每行一条指令。指令按照输入顺序编号为 1~*n*。

## 输出格式

对于每组数据，输出机器人的最终位置。每处理完一组数据，机器人应复位到数轴原点。

| 样例输入 | 样例输出 |
| --- | --- |
| 2<br>3<br>LEFT<br>RIGHT<br>SAME AS 2<br>5<br>LEFT<br>SAME AS 1<br>SAME AS 2<br>SAME AS 1<br>SAME AS 4 | 1<br>-5 |

# Problem C. Updating a Dictionary

In this problem, a dictionary is collection of key-value pairs, where keys are lower-case letters, and values are non-negative integers. Given an old dictionary and a new dictionary, find out what were changed.

Each dictionary is formatting as follows:

```
{key:value,key:value,...,key:value}
```

Each key is a string of lower-case letters, and each value is a non-negative integer without leading zeros or prefix '+'. (i.e. -4, 03 and +77 are illegal). Each key will appear at most once, but keys can appear in any order.

## Input

The first line contains the number of test cases $T$ ($T$<=1000). Each test case contains two lines. The first line contains the old dictionary, and the second line contains the new dictionary. Each line will contain at most 100 characters and will not contain any whitespace characters. Both dictionaries could be empty.

**WARNING:** there are no restrictions on the lengths of each key and value in the dictionary. That means keys could be really long and values could be really large.

## Output

For each test case, print the changes, formatted as follows:

- First, if there are any new keys, print '+' and then the new keys in increasing order (lexicographically), separated by commas.
- Second, if there are any removed keys, print '-' and then the removed keys in increasing order (lexicographically), separated by commas.
- Last, if there are any keys with changed value, print '*' and then these keys in increasing order (lexicographically), separated by commas.

If the two dictionaries are identical, print '`No changes`' (without quotes) instead.

Print a blank line after each test case.

## Sample Input

```
3
{a:3,b:4,c:10,f:6}
{a:3,c:5,d:10,ee:4}
{x:1,xyz:123456789123456789123456789}
{xyz:123456789123456789123456789,x:1}
{first:1,second:2,third:3}
{third:3,second:2}
```

## Output for Sample Input

```
+d,ee
-b,f
*c

No changes

-first
```

# 题目 D. 平方根大搜索

在二进制中，2 的算术平方根，即 sqrt(2)，是一个无限小数 1.0110101000001001111...

给定一个整数 $n$ 和一个 01 串 S，你的任务是在 sqrt($n$)的小数部分（即小数点之后的部分）中找到 S 第一次出现的位置。如果 sqrt($n$)是整数，小数部分看作是无限多个 0 组成的序列。

## 输入格式

输入第一行为数据组数 $T$ ($T$<=20)。以下每行为一组数据，仅包含一个整数 $n$ (2<=$n$<=1,000,000)和一个长度不超过 20 的非空 01 串 S。

## 输出格式

对于每组数据，输出 S 的第一次出现中，第一个字符的位置。小数点后的第一个数字的位置为 0。输入保证答案不超过 100。

| 样例输入 | 样例输出 |
| --- | --- |
| 2<br>2 101<br>1202 110011 | 2<br>58 |

# 题目 E. 最短的名字

在一个奇怪的村子中，很多人的名字都很长，比如 *aaaaa*, *bbb* and *abababab*。

名字这么长，叫全名显然起来很不方便。所以村民之间一般只叫名字的前缀。比如叫'*aaaaa*'的时候可以只叫'*aaa*'，因为没有第二个人名字的前三个字母是'*aaa*'。不过你不能叫'*a*'，因为有两个人的名字都以'*a*'开头。村里的人都很聪明，他们总是用最短的称呼叫人。输入保证村里不会有一个人的名字是另外一个人名字的前缀（作为推论，任意两个人的名字都不会相同）。

如果村里的某个人要叫所有人的名字（包括他自己），他一共会说多少个字母？

## 输入格式

输入第一行为数据组数 $T$($T$<=10)。每组数据第一行为一个整数 $n$(1<=$n$<=1000)，即村里的人数。以下 $n$ 行每行为一个人的名字（仅有小写字母组成）。输入保证一个村里所有人名字的长度之和不超过 1,000,000。

## 输出格式

对于每组数据，输出所有人名字的字母总数。

| 样例输入 | 样例输出 |
| --- | --- |
| 1<br>3<br>aaaaa<br>bbb<br>abababab | 5 |

# Problem F. Kingdoms

A kingdom has $n$ cities numbered 1 to $n$, and some bidirectional roads connecting cities. The capital is always city 1.

After a war, all the roads of the kingdom are destroyed. The king wants to rebuild some of the roads to connect the cities, but unfortunately, the kingdom is running out of money. The total cost of rebuilding roads should not exceed $K$.

Given the list of $m$ roads that can be rebuilt (other roads are severely damaged and cannot be rebuilt), the king decided to maximize the total population in the capital and all other cities that are connected (directly or indirectly) with the capital (we call it "accessible population"), can you help him?

## Input

The first line of input contains a single integer $T$ ($T$<=20), the number of test cases. Each test case begins with three integers $n$(4<=$n$<=16), $m$(1<=$m$<=100) and $K$(1<=$K$<=100,000). The second line contains $n$ positive integers $p_i$ (1<=$p_i$<=10,000), the population of each city. Each of the following $m$ lines contains three positive integers $u$, $v$, $c$ (1<=$u$,$v$<=$n$, 1<=$c$<=1000), representing a destroyed road connecting city $u$ and $v$, whose rebuilding cost is $c$. Note that two cities can be directly connected by more than one road, but a road cannot directly connect a city and itself.

## Output

For each test case, print the maximal accessible population.

| Sample Input | Output for Sample Input |
|---|---|
| 2<br>4 6 6<br>500 400 300 200<br>1 2 4<br>1 3 3<br>1 4 2<br>4 3 5<br>2 4 6<br>3 2 7<br>4 6 5<br>500 400 300 200<br>1 2 4<br>1 3 3<br>1 4 2<br>4 3 5<br>2 4 6<br>3 2 7 | 1100<br>1000 |

# 题目 G. 网格中的三角形

有一个 $n$ 行 $m$ 列单位正方形组成的网格。不难发现一共有 $n$+1 条横线，$m$+1 条竖线和它们形成的($n$+1)($m$+1)个交叉点。你可以选择三个**不共线**的交叉点，形成一个三角形。比如当 $n$=$m$=1 时，一共有 4 个交叉点，可以形成 4 个三角形。

问：有多少个三角形的面积在 A 和 B 之间（包含 A 和 B）。

## 输入格式

输入第一行为数据组数 $T$($T$<=25)。每组数据为四个整数 $n$, $m$, $A$, $B$ (1<=$n$, $m$<=200, 0<=$A$<$B$<=$nm$)。

## 输出格式

对于每组数据，输出面积在 A 和 B 之间的三角形个数。

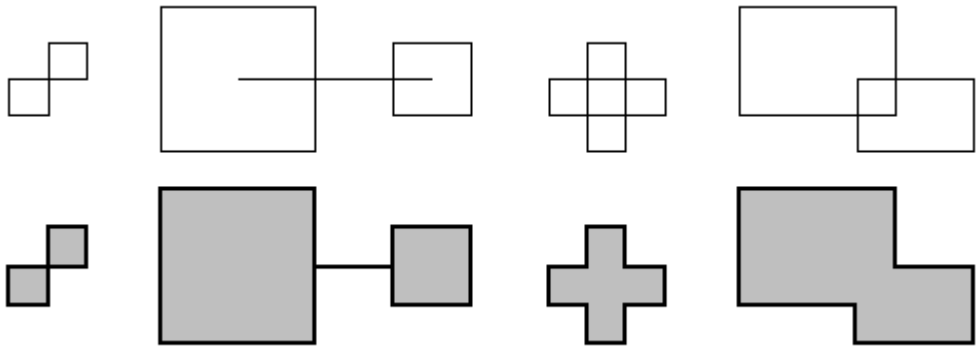
| 样例输入 | 样例输出 |
| --- | --- |
| 4<br>1 1 0 1<br>1 2 1 2<br>10 10 20 30<br>12 34 56 78 | 4<br>6<br>27492<br>1737488 |

# Problem H. Tin Cutter II

In a Tin Cutting factory there is a machine for cutting parts from tin plates. It has an extraordinarily sharp knife able to make horizontal or vertical segment cuts in the tin plates. Each cutting process consists of a sequence of such cuts. Each segment cut is given by its endpoints that are always located inside the tin plate. During the cutting process some parts of tin plate can fall out and so some holes in the plate can emerge.

Factory management needs to predict the length of visible border lines at the end of the given sequence of cuts. Write a program that answers this question.

Here are four examples:



The first row in the picture are four cuttings and the second row are their corresponding resulting plates. Each gray area is a separate hole, and thick lines are visible border lines after cutting. There are 2, 2, 1, 1 holes respectively (from left to right), and the length of visible border lines are 8, 26, 12, 20 respectively.

## Input

The first line of input contains a single integer $T$ ($T$<=100), the number of test cases. The first line of each test case contains an integer $n$ (1<=$n$<=100), the number of segment cuts. Each of the following $n$ lines describe a segment cut with four integers $x1$, $y1$, $x2$, $y2$ that means a segment cut from ($x1$,$y1$) to ($x2$,$y2$) (0<=$x1$,$y1$,$x2$,$y2$<=10000). The segment is always horizontal or vertical.

## Output

For each test case, print the total length of the border lines.

## Sample Input

```
4
6
0 0 1 0
1 0 1 2
1 2 2 2
2 2 2 1
2 1 0 1
0 1 0 0
9
0 0 4 0
4 0 4 4
4 4 0 4
0 4 0 0
6 1 8 1
8 1 8 3
8 3 6 3
6 3 6 1
2 2 7 2
8
0 1 3 1
3 1 3 2
3 2 0 2
0 2 0 1
1 0 2 0
2 0 2 3
2 3 1 3
1 3 1 0
8
0 1 4 1
4 1 4 4
4 4 0 4
0 4 0 1
3 0 6 0
6 0 6 2
6 2 3 2
3 2 3 0
```

## Output for Sample Input

```
8
26
12
20
```

# Problem I. Collecting Coins

In a maze of *r* rows and *c* columns, your task is to collect as many coins as possible.

Each square is either your start point "`S`"(which will become empty after you leave), an empty square "`.`", a coin square "`C`" (which will become empty after you step on this square and thus collecting the coin), a rock square "`O`" or an obstacle square "`X`".

At each step, you can move one square to the up, down, left or right. You cannot leave the maze or enter an obstacle square, but you can push each rock at most once (i.e. You can treat a rock as an obstacle square after you push it).

To push a rock, you must stand next to it. You can only push the rock along the direction you're facing, into an neighboring empty square (you can't push it outside the maze). For example, if the rock is to your immediate right, you can only push it to its right neighboring square.

Find the maximal number of coins you can collect.

## Input

The first line of input contains a single integer $T$ ($T$<=25), the number of test cases. Each test case begins with two integers $r$ and $c$ (2<=$r$,$c$<=10), then followed by $r$ lines, each with $c$ columns. There will be at most 5 rocks and at most 10 coins in each maze.

## Output

For each test case, print the maximal number of coins you can collect.

| Sample Input | Output for Sample Input |
|---|---|
| 3<br>3 4<br>S.OC<br>..O.<br>.XCX<br>4 6<br>S.X.CC<br>..XOCC<br>...O.C<br>....XC<br>4 4<br>.SXC<br>OO.C<br>..XX<br>.CCC | 1<br>6<br>3 |

# 题目 J. 病毒

你有一个日志文件，里面记录着各种系统事件的详细信息。自然的，事件的时间戳按照严格递增顺序排列（不会有两个事件在完全相同的时刻发生）。

遗憾的是，你的系统被病毒感染了，日志文件中混入了病毒生成的随机伪事件（但真实事件的相对顺序保持不变）。备份的日志文件也被感染了，但由于病毒采用的随机感染方法，主日志文件和备份日志文件在感染后可能会变得不一样。

给出被感染的主日志和备份日志，求真实事件序列的最长可能长度。

## 输入格式

输入第一行为数据组数 $T$($T$<=100)。每组数据包含两行，分别描述感染后的主日志和备份日志。每个日志文件的格式相同，均为一个整数 $n$ (1<=$n$<=1000)（代表感染后的事件总数）和 $n$ 个不超过 100,000 的正整数（表示感染后各事件的时间戳）。注意，感染后可能会出现时间戳完全相同的事件。

## 输出格式

对于每组数据，输出真实事件序列的最长可能长度。

| 样例输入 | 样例输出 |
| --- | --- |
| 1<br>9 1 4 2 6 3 8 5 9 1<br>6 2 7 6 3 5 1 | 3 |

# Problem K. Cross-Shaped Tests

There is an $n*n$ matrix. Each number can be increased or decreased. If we increase a number by $x$ (which may be non-integer), the cost is $c*x$, If we decrease a number by $x$, the cost is $d*x$, where $c$ and $d$ are two non-negative integers.

The theoretical goal is to make every number equal to $F$, but in practice we only do $Q$ tests, each test is to specify a square, then adding all the numbers in the same row or column, if the difference between the sum and $(2n-1)F$ is at most $e$, the test is successful.

Your task is to survive all the tests with minimal cost. It's not hard to see that the actual new matrix might be very different from the theoretical goal.

## Input

The first line contains $T$ ($T$<=100), the number of test cases. Each of the following lines begins with six integers $n$, $c$, $d$, $F$, $e$, $Q$(1<=$n$<=12, 0<=$c$,$d$<=100, -10<=$F$<=10, 0<=$e$<=5, 1<=$Q$<=$n^2$). Then an $n*n$ matrix of integers followed. Each integer will have an absolute value of no greater than 10. Then $Q$ lines followed. Each line contains two integers $x$, $y$ (1<=$x$,$y$<=$n$), that means we make a test on the square at row $x$, column $y$. Rows are numbered 1 to $n$ from top to bottom, columns are numbered 1 to $n$ from left to right. Each test case is terminated by a blank line.

## Output

For each test case, print the minimal cost, to five digits after the decimal point.

| Sample Input | Output for Sample Input |
|---|---|
| 2<br>5 12 23 2 1 2<br>0 0 1 0 0<br>1 2 3 1 1<br>3 1 5 3 3<br>0 0 1 0 0<br>0 0 1 0 0<br>2 3<br>3 3<br>2 0 1 0 0 3<br>1 -1<br>-1 -2<br>1 1<br>2 1<br>2 2 | 58.00000<br>0.50000 |

# 题目 L 安全区域

太空中有 $n$ 类卫星（即类型 1、类型 2…）共 $m$ 个。对于满足 $1<=i<=n$ 的任意整数 $i$，类型 $i$ 的所有卫星共同保卫着包含它们的最小凸多面体（即凸包。可能会退化，即体积为 0）。如果空间中一个点被至少 $k$ 类卫星保护，我们说这个点属于安全区域。

你的任务是计算所有安全区域的总体积。

## 输入格式

输入第一行为数据组数 $T$($T$<=25)。每组数据第一行为三个整数 $n$, $k$ 和 $m$($1<=k<=n<=5$, $4<=m<=50$)。以下 $m$ 行每行包含三个**实数** $x$, $y$, $z$，表示一个类型 $t$ 的卫星，坐标为 ($x$,$y$,$z$)($1<=t<=n$, $0<=x,y,z<=10$)。每组输入数据后有一个空行。

注意：评测数据（而非样例数据）中卫星坐标都是随机生成的。

## 输出格式

对于每组数据，输出安全区域的总体积，四舍五入保留小数点 5 位。

## 样例输入

```
2
2 1 16
1 0 0 0
1 0 0 2
1 0 2 0
1 0 2 2
1 2 0 0
1 2 0 2
1 2 2 0
1 2 2 2
2 1 1 1
2 1 1 3
2 1 3 1
2 1 3 3
2 3 1 1
2 3 1 3
2 3 3 1
2 3 3 3

1 1 4
1 0 0 0
1 0 1 0
1 0 0 1
1 1 0 0
```

## 样例输出

```
15.00000
0.16667
```