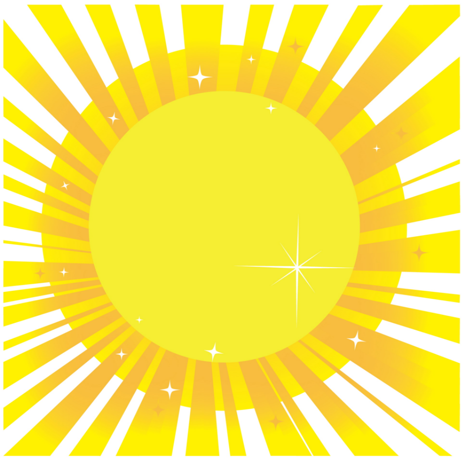
COMPUTER AND INFORMATION SCIENCE

UNIVERSITY OF OREGON

Eighteenth Annual University of Oregon Eugene Luks Programming Competition

2014 April 19, Saturday
am10:00 – pm2:00



Problem Contributors
David Atkins
Eugene Luks
Chris Wilson

Technical Assistance
David Atkins
Paul Bloch

PAY IT FORWARD

You have just graduated and started your first career job. It may seem like planning for retirement is the last thing you want to think about now, but starting retirement savings early is the key to a successful plan. Few companies now offer defined pension benefit plans and most workers in the current generation will work for many different employers during their life anyway. Government retirement programs will help, but an individual's own savings will be a very critical part of any retirement plan, so it is more important than ever to begin a savings plan early to insure that there will be a good nest egg to help support your retirement, or even allow you to retire early.

Although it is impossible to know the future, you can make some simplifying assumptions to help you estimate how much current income should be saved for retirement. For example, if you have a career working for 40 years (e.g., age 25-65) and save 10% of your salary every year, at the end of 40 years you would have 400% of your salary saved. If you then retired and paid yourself from your savings an amount equal to 50% of your working salary each year, your savings would last for just 8 years. That sounds pretty bleak, but over time the savings would grow not just from your yearly contributions, but also from investment returns. If you assume that your savings grows 3% each year, then through the power of compounding, your savings would instead be worth over 776% of your salary after 40 years. At first glance this would appear to last for 15 whole years (776/50 = 15.5), but since the investment growth would continue as you withdraw during retirement, your savings would actually last for 20 whole years of retirement.

For this problem you are to write a program which is given a retirement savings scenario and your program must compute how many whole years of retirement the savings will support.

The first line of input will be a number that indicates how many scenarios follow. Each scenario is on a line by itself consisting of the following four values: a percentage which is the amount of salary saved each year; an integer which is the number of years of savings; a percentage which is the annual investment growth rate of the savings; and a percentage which is the amount of salary to be withdrawn each year in retirement. The percentages may be floating point numbers. For each scenario, your program must output the number of whole years of retirement withdrawals possible in the form shown. If there is not enough for even the first year of withdrawal, output "Not enough saved to retire!", and if 50 or more years of withdrawals are possible, output "50+ years".

Sample Input

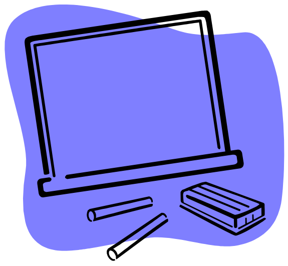
```
5
10.0 40 0.0 50.0
10.0 40 3.0 50.0
5.0 10 2.0 75.0
25 45 5 25
30.0 1 0.0 30.0
```

Sample Output

```
8 years
20 years
Not enough saved to retire!
50+ years
1 year
```

Comments: Assume the amount saved is credited at the beginning of each year, and the growth is added at the end of each year. In retirement, assume you withdraw at the beginning of each retirement year and the growth is added at the end of the year.

We Like UNIX, We Like C, We Want More 330



As an avid student at the UO, you have discovered the incredible experience of learning about UNIX system calls. To your joy, you have found out that the CIS Department offers two sections of CIS 330 next term covering this material. One is taught by Professor B and the other by Professor C. Each issues a syllabus at the start of the term, and the syllabus gives the expected number of hours you will need to spend each week on programming. Since they each have their own style, the number of hours per week may be different for the two professors.

This term is N weeks long – the UO is flexible now. Your goal is to maximize your workload. Fortunately, you are allowed to switch from one section to the other. However, whenever you switch sections you are not allowed to work for a week (due to bureaucratic constraints). It does not matter in which section you start, nor in which section you end.

For example, if Professor B's schedule in weeks 1 through N=5 is (12, 3, 20, 6, 19) and Professor C's schedule is (8, 26, 20, 2, 5), you maximize your workload by taking Professor C's section for weeks 1, 2, and 3, then switching to take Professor B's section in week 5. This results in a schedule of a total of 73 hours of programming work.

The input will start with an integer K, being the number of cases (scenarios for a term). This will be followed by K descriptions of a term. A term description starts with a line containing the number of weeks N. This is followed by N lines of the form B C, where on the ith line B is an integer giving the expected number of hours of work in Professor B's section in week i, and C is the same for Professor C.

Sample Input

```
2
5
12 8
3 26
20 20
6 2
19 5
7
5 14
6 7
19 21
4 9
21 13
16 5
19 9
```

Sample Output

```
Case 1: 73
Case 2: 98
```

Comments:
You can assume that K, N ≥ 1, and B, C ≥ 0 are all integers.

Romulan Secret Agents

Romulan spies Ael and Bok'ra communicate via a symmetric cipher system which depends upon a daily secret key

$$(p, a, b)$$

where p is a 25-digit (decimal) prime and $1 \leq a,b < p$. Messages are assumed to be nonnegative integers $< p$. Then the encryption $\mathbf{E}(M)$ of the plaintext message M satisfies

$$\mathbf{E}(M) \equiv aM + b \pmod{p},$$

with $0 \leq \mathbf{E}(M) < p$.

Your Cardassian Intelligence Agency assignment involves decrypting the messages exchanged between Ael and Bok'ra. You can count on your counterspies to provide enough instances of plaintext/ciphertext pairs for you to decrypt all messages of the day.

In this problem, you are given some pairs $(M, \mathbf{E}(M))$ from a given day together with an additional encrypted message $\mathbf{E}(M_0)$ and you are to determine M_0.

The first line of input to the problem will consist of an integer n indicating the number of instances (days) being covered. The first line for each instance will consist of an integer r indicating the number of known plaintext/cipher pairs, which will then follow, one pair of integers to a line, separated by a single space. Finally, there will be a line with an encrypted message.

Your output should consist of n lines, each with a corresponding decryption.

For better readability, the sample **Input/Output** assumes only 4-digit primes.

Sample Input

```
2
4
4674 7348
3121 6600
3740 6435
5324 4383
2572
5
1182 1296
1576 1253
473 1920
1351 1668
2579 1392
785
```

Sample Output

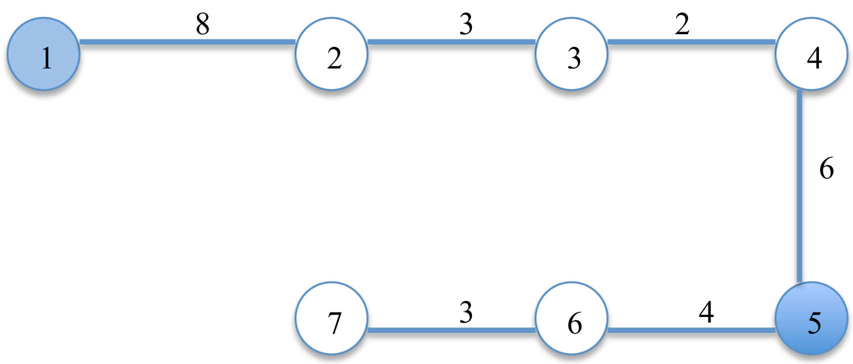
```
3
10
```

Remark. In the first instance, it turns out that $p = 9341$, $a = 211$, $b = 1939$. In the second, $p = 2797$, $a = 2733$, $b = 1425$.

Average Distance to Closest Hospital

You are to help city planners determine the best placement of hospitals. The city consists of a collection of roads and intersections of roads. The intersections are numbered 1,2,…,N, and roads run between some of these intersections. These roads have different lengths, which the planners of course know. Hospitals are located on some of these intersections. What the planners need to know is the average distance of each non-hospital intersection from its closest hospital. Distances are given in kilometers.

For example, in the simple routing pictured below, there are hospitals placed at intersections 1 and 5. Looking at all the other intersections, intersection 2 is 8 kilometers from the nearest hospital, intersection 3 is also 8 kilometers away (from the other one), 4 is distance 6, 6 is distance 4, and intersection 7 is 7 kilometers away. The sum of these distances is 33, and there are 5 intersections without hospitals. The average is 33/5, or 6.6 kilometers.



The input will start with an integer K, the number of cases, each being a separate hospital placement scenario. This will be followed by K descriptions of hospital and city street layout. One line will contain a number H, the number of hospitals. The next line contains two integers N M, where N is the number of intersections and M the number of streets between intersections. The next M lines are of the form I J L, where I and J are two intersections connected by a road, and L is the length of that road in kilometers.

Comments:

- All roads are two way.
- All road lengths are positive integers.
- Not all intersections have a road between them.
- The graph will of course have cycles.
- $0 < H < N$
- The graph is connected.

Sample Input

```
2
3
6 10
1
5
3
1 5 3
1 2 9
1 3 6
5 2 10
2 6 11
4 2 7
3 5 8
3 6 14
4 3 10
6 4 3
2
7 6
1
5
1 2 8
2 3 3
3 4 2
4 5 6
5 6 4
6 7 3
```

Sample Output

```
10.666667
6.6
```

ALPHABETIC BASE CHANGE

On Romulus, numbers are written alphabetically – well maybe not, but suppose they are. They may look like HBBAD or TSCE. How do we read them? We cannot! Well, we can if we know the "base" and how to interpret the characters. Here we need to solve a related problem to help the Romulans. They often write numbers in different bases, and need a program to translate from one base to another.

Suppose we look at the number BAAC in base D. This means that D is the largest possible character – the characters that can be used are A, B, C, and D. We would call that base 4 here on Earth. An A corresponds to 0, B to 1, C to 2, and D to 3 as "digits" in a number. So, BAAC in base D translates to 66 in base 10. If we want to translate BAAC (=66) to base B (or base 2), we get BAAAABA. Similarly, the number WHAT in base X is written in base Y as TSCE.

The first line of the input is the number of cases K. Each case is described by two lines. One line contains two characters A B, where A is the input base and B is the target base. The next line is a string of characters, valid for the input base A. The output should be a number in the target base B.

Sample Input

```
3
D B
BAAC
B D
BAAAABA
X Y
WHAT
```

Sample Output

```
BAAAABA
BAAC
TSCE
```

Comments:

- All characters are upper case, and only characters are used.
- Every input and target base will be at least B (that is, every base will be ≥ 2).
- K > 0
- All numbers will fit into a java long (64 bits).