LUCC Presents

Inter University Programming Contest 2015
Episode - 2

Thursday, 29th October, 2015

You get 12 pages

8 problems

&

240 minutes

Organized by:

Leading University Computer Club (LUCC)

Rules for the contest:

a) Every team must create an account in https://www.hackerrank.com/. The name of the team should be used as the ‘username’.

b) Teams are not allowed to access any website except the judge website.

c) Mobile phones of the contestants must be turned off.

d) Solutions to problems submitted for judging are called runs. Each run is judged as accepted or rejected by the judge, and the team is notified of the results.

e) A contestant may submit a clarification request to judges. If the judges agree that an ambiguity or error exists, a clarification will be issued to all contestants.

f) Contestants are not to converse with anyone except members of their team and personnel designated by the organizing committee while seated at the team desk.

g) While the contest is scheduled for a particular time length (four hours), the contest director has the authority to alter the duration of the contest in the event of unforeseen difficulties. Should the contest duration be altered, every attempt will be made to notify contestants in a timely and uniform manner.

h) A team may be disqualified by the Contest Director for any activity that jeopardizes the contest such as dislodging extension cords, unauthorized modification of contest materials, distracting behavior of communicating with other teams.

i) Eight problems will be posed. So far as possible, problems will avoid dependence on detailed knowledge of a particular applications area or particular contest language. Of these problems at least three will be solvable by a first year computer science student, another two will be solvable by a second year computer science student and rest will determine the winner.

j) Contestants will have foods available in their contest room during the contest. So they cannot leave the contest room during the contest without permission. The contestants are not allowed to communicate with any contestant (Even contestants of his own team) while are outside the contest floor.

k) Teams can bring up to 200 pages of printed materials with them but they can also bring three additional books. But they are not allowed to bring calculators, mobile phones or any machine-readable devices like CD, DVD, Pen-drive, IPOD, MP3/MP4 players, floppy disks etc.

l) The decision of the judges is final.

m) If you want to assume that judge data is weaker than what is stated, then do it at your own risk ☺

Contest link:

https://www.hackerrank.com/lu-iupc-2015

A	Choose Your Own Adventure
	Time limit: 3 Seconds

Problem Statement:

After reading the book *Tim and Marc Kill Kenny* about fifty zillion times, James decided he'd had it with choose-your-own-adventure stories. No matter what choices he made, it seemed like Kenny always fell down an abandoned mine shaft, got run over by a bus load of nuns, or was messily devoured by stray cats. James eventually found the page with the happy ending (where Kenny saves himself by trapping Tim and Marc between the pizza and the hungry programmers) by flipping through the book, but he can't figure out how to get there by following the rules. Luckily, he owns a C++ compiler...

Input:

Input to this problem will consist of a (non-empty) series of up to 100 data sets, each representing a choose-your-own-adventure story. Each data set will be formatted according to the following description, and there will be **no blank lines** separating data sets.

The first line contains a single integer *n* indicating the number of data sets.

A single data set has 2 components:

1. *Page Count* - A line containing a single integer X, where $1 < X < 100$, indicating the number of pages in the story.
2. *Page List* - A sequence of X lines, each of which represents a page from the book. Each line has the following components separated from one another by single spaces:
   - *Line type* - A single character indicating what type of line this is. It will represent either a "C" choice page, or an "E" end page. Page 1 is always a choice page.
   - *Text* - A string of text surrounded by double quotes. Including the quotes, this component will not exceed 256 characters. The quotes are given for input purposes only and should not be considered part of the text. The text will not contain embedded double quotes.
   - *Choices* - Two positive integers from 1 to X indicating the pages where the reader can go from this page. Only choice pages have this component.
   - *Ending Type* - Either the text "HAPPY" or "GRISLY". There will only be one happy ending per story, and only end pages have this component.

Output:

For each story in the input:

1. Output a single line, "STORY #" where # is 1 for the first story, 2 for the second story, etc.
2. Determine the story that begins on page 1 and ends on the happy ending page. Output the text of this story, printing one "page" of text per line. Note that there is only one such story for each data set.

Sample Input:

```
2
3
C "Arrived at LU for the contest" 2 3
E "Was devoured by sidewalk ants" GRISLY
E "Won the contest. Received glory and chocolates." HAPPY
5
C "Saw a peanut" 3 5
E "Made peanut butter sandwich" HAPPY
C "Found a hammer" 4 2
E "Hit self on head with hammer, ouch!" GRISLY
E "Ate the peanut, choked on it, and died" GRISLY
```

Sample Output:

```
STORY 1
Arrived at LU for the contest
Won the contest. Received glory and chocolates.
STORY 2
Saw a peanut
Found a hammer
Made peanut butter sandwich
```

B

Galactik Football

Time limit: 2 Seconds

It's Galactik Football time! The Galactik Football Association (GFA) has announced a football tournament between all the teams of all the planets in the galaxy (say **N**). Teams like Snow Kids, Shadows, Xenons, Red Tigers, Wambas, Pirates, etc. are in total enthusiasm and are practising hard to win the cup using their talent and flux.

Each planet of the galaxy has a government. Some governments have a **mutual agreement** between them. If planet **A** has mutual agreement with planet **B**, then there is a **bidirectional spaceway** between **A** and **B** using which anybody can go from **A** to **B** and **vice-versa**. People can use these spaceways to travel from one planet to another, if there exists a **path** between them using some of the spaceways.

Each planet has its own football ground. The GFA has planned the matches in such a way that a team can have a match at any of these grounds. The GFA has come across some problems in the execution of their plan. They have found out that there are many pairs of planets between which there **does not exist any path**, so the football team of one of those planets can't reach the other planet. They requested the corresponding governments to make a spaceway between them, but because of absence of mutual agreement (duhhh.. Politics!), these government did not agree. So the GFA suggested that they will make teleports between some pairs of planets which will be used only by the football teams to travel.

But there are two types of governments in the galaxy:

1. Some of the governments are greedy (duhhh..). They want to make money (You don't say!) through the GFA. So each of these government has asked the GFA for a tax value which it has to pay if it wants to make a teleport ending at their planet.
2. Others want to sponsor the event, so they will give money to the GFA if they make a teleport ending at their planet (That's new..). The GFA would always avoid such governments no matter what the consequences are, because these kind of governments have always some dirty plans in their minds for the GFA.

Now, the GFA wants to make bi-directional teleports between planets such that the football teams of any planet can reach any other planet to play a football match, using spaceways between the planets and/or teleports made by the GFA.

The GFA also has financial problems and want to spend as little money as possible. They have come to you so that you can help them calculate the minimum amount of money needed to fulfill their plan.

Input

The input file contains several test cases separated by a blank line. The first line of each case consists of two integers - **N** and **M**. **N** is number of planets and **M** is number of pairs of planets which have a mutual agreement, i.e they have a spaceway between them. Then, **M** lines follow, each containing two space separated integers **A** and **B**, denoting a mutual agreement and hence a spaceway to travel, between planet **A** and planet **B**. Then, **N** lines follow. The **ith** line has an integer **C**. If **C ≥ 0**, then it represents the tax value which the GFA has to pay to the government of planet i (it's a **type 1** government). If **C < 0**, then it represents the money the **ith** government will pay to the GFA (it's a **type 2** government).

Output

Print the minimum amount needed for the GFA to fulfill their plan if it can be fulfilled, else print "-1" (without quotes).

Constraints

```
1 ≤ N ≤ 100,000
0 ≤ M ≤ 1,000,000
0 ≤ |C| ≤ 10,000
1 ≤ A,B ≤ N
A ≠ B
```

Sample Input

```
6 6
1 2
2 3
1 3
4 5
5 6
4 6
1
3
5
2
4
6

3 1
2 3
1
-1
-1
```

Sample Output

```
3
-1
```

C

Akbar and Race Team

Time limit: 2 Seconds

Coach Akbar is forming a team for the Annual Inter Galactic Relay Race. He has N students that train under him and he knows their strengths. The strength of a student is represented by a positive integer.

The coach has to form a team of K students. The strength of a team is defined by the strength of the weakest student in the team. Now he wants to know the sum of strengths of all the teams of size K that can be formed modulo 1000000007. Please help him.

Input

The first line contains the number of test cases T. Each case begins with a line containing integers N and K. The next line contains N space-separated numbers which describe the strengths of the students.

Output

For test case output a single integer, the answer as described in the problem statement.

Constraints

```
1<=T<=100
1<=N<=100000
1<=K<=N
0<=Strength of each student<=2000000000
Strength of all the students are different.
```

Sample Input

```
2
2 1
5 4
3 2
1 0 2
```

Sample Output

```
9
1
```

Explanation

For first test case: 5+4=9, as team can only consist of 1 student.
For second test case: `min(1,0) + min(1,2) + min(0,2) = 0+1+0 =1`

D

Survive

Time limit: 2 Seconds

The game is simple. You initially have ‘H’ amount of health and ‘A’ amount of armor. At any instant you can live in any of the three places - fire, water and air. After every unit time, you have to change your place of living. For example if you are currently living at fire, you can either step into water or air.

- If you step into air, your health increases by 3 and your armor increases by 2.
- If you step into water, your health decreases by 5 and your armor decreases by 10.
- If you step into fire, your health decreases by 20 and your armor increases by 5.

If your health or armor becomes <= 0, you will die instantly. Find the maximum time you can survive.

Input

The first line consists of an integer T, the number of test cases. For each test case there will be two positive integers representing the initial health H and initial armor A.

Output

For each test case find the maximum time you can survive. You can choose any of the 3 places as your first move.

Input Constraints

```
1 <= T <= 10
1 <=H,A<= 1000
```

Sample Input:

```
3
2 10
4 4
20 8
```

Sample Output:

```
1
1
5
```

E	Tanya and Combination
	Time limit: 5 Seconds

Tanya loves solving problems which are deemed impossible by her fellow classmates. The current problem which she is working on is to express a number **`N`** as sum of powers of number **`X`** (Not necessarily distinct) such that the number of powers of number **`X`** used should be minimum.

Note: The powers of a number can be **0, 1, 2, 3, 4 ...**

Input Format

The first line will contain T $(1 <= T <= 10^5)$, the number of test cases. Then, **`T`** lines follow, each containing 2 space separated integers **`N`** and M $(1 <= N, M <= 10^{14})$.

Output Format

For each test case, output the minimum number of such numbers (powers of **`M`**) which can be summed up to produce **`N`**.

Sample Input

```
3
4 4
5 3
6 1
```

Sample Output

```
1
3
6
```

Explanation

Case 1. 4 can be expressed as 4^1. Case 2. 5 can be expressed as sum of $3^0 + 3^0 + 3^1$. Case 3. 6 can be expressed as sum of $1^1 + 1^1 + 1^1 + 1^1 + 1^1 + 1^1$.

F	Ujjal and Jumps
	Time limit: 2 Seconds

Ujjal being a very talkative child, was scolded by his teacher multiple times. One day, the teacher became very angry and decided to give him a very rigorous punishment. He made him stand on the school field which has **`X`** axes.

Ujjal initially stood at **`X = 0`**. The teacher asked him to run to **`X = N`**. But, to make the process quick, Ujjal decided that he will make jumps of **2 or 3 steps** only, i.e., from **`X = S`** he can jump to **`X = S+2`** or **`X = S+3`.**

Ujjal decided that he will jump 2 steps with probability **`P/100`** and jump 3 steps with probability **`1-P/100`**.

You need to find the probability that he will reach **exactly on `X = N`**.

Constraints

$0 < N <= 10^6$

$0 <= P <= 100$

Input Constraints

The first line contains two integer **`N`** and **`P`**.

Output Constraints

Your answer must contain **exactly 6 digits after the decimal point**.

Sample Input

```
5 20
```

Sample Output

```
0.320000
```

Explanation

There are two ways to reach 5.

1. 2+3 with probability = 0.2 * 0.8 = 0.16
2. 3+2 with probability = 0.8 * 0.2 = 0.16

So, total probability = 0.32.

G

Happy Number

Time limit: 2 Seconds

Write a program to determine if a number is "happy".

A happy number is a number defined by the following process: Starting with any positive integer, replace the number by the sum of the squares of its digits, and repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1. Those numbers for which this process ends in 1 are happy numbers.

Example: 19 is a happy number

- $1^2 + 9^2 = 82$
- $8^2 + 2^2 = 68$
- $6^2 + 8^2 = 100$
- $1^2 + 0^2 + 0^2 = 1$

Input

The first line of the input is a positive integer `T (T <= 100)`, denoting the number of test cases. Each of the next `T` lines contains a single integer `N (1 <= N <= 1000000)`.

Output

For each test case, print “`Happy`” (without the quotation) if N is a happy number. Print “`Not Happy`” otherwise.

Sample Input

```
5
19
7
21
13
999
```

Sample Output

```
Happy
Happy
Not Happy
Happy
Not Happy
```

H

Chandu Mama And Fibonacci

Time limit: 2 Seconds

This year, Leading University is hosting an inter university programming contest. Chandu Mama is one of the legendary problem setter of Bangladesh. The contest director of Leading University Mr. A, also best friend of Chandu Mama thought that it'd be great if Chandu Mama sets a problem for their upcoming programming contest. So he requested his childhood friend Chandu Mama to set a problem for **LU Inter University Programming Contest**. To keep the request of his best friend, Chandu Mama agreed to set a problem.

Given N and K, you have to represent N as a summation of some Fibonacci numbers **not necessarily distinct**. For example if we write $N = A_1 + A_2 + A_3 + ... + A_P$, then Ai, $(1 <= i <= P)$ must be a Fibonacci number. Also you need to **minimize** P.

Example:
$10 = 2 + 8$ and $6 = 3 + 3$ are valid where $12 = 4 + 8$ is not.
$6 = 1 + 2 + 3$ is also not valid because you can represent 6 as a summation of 2 Fibonacci numbers.

Chandu Mama doesn't like Fibonacci numbers greater than K. So you can't use a Fibonacci number which is greater than K. That means $A_i <= K$, $(1 <= i <= P)$.
For $N = 10$ and $= 8$, a valid solution is $10 = 2 + 8$,
For $N = 15$ and $K = 8$, a valid solution is $15 = 5 + 5 + 5$.

If you don't know what is a Fibonacci number, then see the recurrence relation below:
$F_0 = 0$
$F_1 = 1$
$F_i = F_{i-1} + F_{i-2}$, $i >= 2$

Input

The first line of the input contains an integer T $(T <= 10000)$ denoting the number of test cases. Each of the following T lines has two space separated integers N and K, $1<=N, K<=10^{18}$.

Output

For each input, print the output in the format, 'Case C: P' (quote for clarity). Here C is the case number starting from 1, P is the minimum number of Fibonacci numbers not greater than K(**not necessarily distinct**) whose summation is exactly equal to N.

Sample Input:	**Sample Output:**

3
5 3
60 49
1000000000000000000 999999999999999999
	Case 1: 2
	Case 2: 3
	Case 3: 27