Code Jam 2014 - World Finals

ARAM

Problem

In the game League of Legends™, you can play a type of game called "ARAM", which is short for "All Random, All Mid". This problem uses a similar idea, but doesn't require you to have played League of Legends to understand it.

Every time you start playing an ARAM game, you're assigned one of **N** "champions", uniformly at random. You're more likely to win with some champions than with others, so if you get unlucky then you might wish you'd been given a different champion. Luckily for you, the game includes a "Reroll" function.

Rerolling randomly reassigns you a champion in a way that will be described below; but you can't reroll whenever you want to. The ability to reroll works like a kind of money. Before you play your first ARAM game, you begin with **R** RD ("reroll dollars"). You can only reroll if you have at least 1 RD, and you must spend 1 RD to reroll. After every game, you gain 1/**G** RD (where **G** is an integer), but you can never have more than **R** RD: if you have **R** RD and then play a game, you'll still have **R** RD after that game.

If you have at least 1RD, and you choose to reroll, you will spend 1RD and be re-assigned one of the **N** champions, uniformly at random. There's some chance you might get the same champion you had at first. If you don't like the champion you rerolled, and you still have at least 1RD left, you can reroll again. As long as you have at least 1RD left, you can keep rerolling.

For example, if **R**=2 and **G**=2, and you use a reroll in your first game, then after your first game you will have 1.5 RD. If you play another game, this time without using a reroll, you will have 2.0 RD. If you play another game without using a reroll, you will still have 2.0 RD (because you can never have more than **R**=2). If you use two rerolls in your next game, then after that game you will have 0.5 RD.

You will be given the list of champions, and how likely you are to win a game if you play each of them. If you play 10^{100} games and choose your strategy optimally, what fraction of the games do you expect to win?

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each starts with a line containing three space-separated integers: **N**, **R** and **G**. The next line contains **N** space-separated, real-valued numbers $\mathbf{P}_i$, indicating the probability that you will win if you play champion `i`.

Output

For each test case, output one line containing "Case #x: y", where x is the test case number (starting from 1) and y is the proportion of games you will win if you play 10^{100} games.

y will be considered correct if it is within an absolute or relative error of 10^{-10} of the correct answer. See the FAQ for an explanation of what that means, and what formats of real numbers we accept.

Limits

Memory limit: 1 GB.
1 ≤ **T** ≤ 100.
0.0 ≤ $\mathbf{P}_i$ ≤ 1.0.
$\mathbf{P}_i$ will be expressed as a single digit, followed by a decimal point, followed by 4 digits.

Small dataset

Time limit: 60 seconds.
1 ≤ **N** ≤ 1000.
1 ≤ **R** ≤ 2.
1 ≤ **G** ≤ 3.

Large dataset

Time limit: 120 seconds.
1 ≤ **N** ≤ 1000.
1 ≤ **R** ≤ 20.
1 ≤ **G** ≤ 20.

Sample

Sample Input

```
3
2 1 1
1.0000 0.0000
3 1 1
1.0000 0.0000 0.5000
6 2 3
0.9000 0.6000 0.5000 0.1000
0.2000 0.8000
```

Sample Output

```
Case #1: 0.750000000000
Case #2: 0.666666666667
Case #3: 0.618728522337
```

Note

League of Legends is a trademark of Riot Games. Riot Games does not endorse and has no involvement with Google Code Jam.