Rack 'em Up

Computer Science Society
Programming Contest
Spring 2008

Bowling is a game requiring many talents—the ability to lift and throw a sixteen-pound ball, a keen fashion eye for gaudy clothes, and the advanced mathematical skills necessary to keep score. Computers aren't much help in lifting or dressing, but they are fine tools for keeping score. In this problem, we define a new way of scoring known as "Fibonacci" scoring and write a program to keep score.

In bowling, the basic idea is to roll the ball toward standing wooden pins, trying to knock down as many as possible. A game consists of ten *frames*. Each frame begins with ten pins standing, and the bowler is given a first roll to knock down as many pins as possible, and a second roll (if any pins are left standing after the first roll) to knock down as many of the remaining pins as possible. If all ten pins are knocked down after the first roll, it is called a *strike*. If all ten pins are knocked down after the second roll, it is called a *spare*. If some pins remain standing after the second roll, it is called an *open frame*.

Let p_i be the number of pins knocked down and r_i be the number of rolls taken in frame i, for $i = 1, 2, 3, \ldots, 10$. Note that a strike has $p_i = 10$ and $r_i = 1$, a spare has $p_i = 10$ and $r_i = 2$, and an open frame has $0 \leqslant p_i < 10$. A score s_i is defined for each frame $i = 1, 2, 3, \ldots, 10$ using "Fibonacci" scoring, in which a strike usually earns $p_i + s_{i-1} + s_{i-2}$ points, a spare usually earns $p_i + s_{i-1}$ points, and an open frame earns p_i points. Formally, the score s_i for frame i is defined recursively by

$$s_i = \begin{cases} p_i + s_{i-1} + s_{i-2} & \text{if } p_i = 10 \text{ and } r_i = 1 \text{ and } i \geqslant 3, \\ p_i + s_{i-1} & \text{if } p_i = 10 \text{ and } ((r_i = 2 \text{ and } i \geqslant 2) \text{ or } (r_i = 1 \text{ and } i = 2)), \\ p_i & \text{otherwise,} \end{cases}$$

and the total score is $\sum_{i=1}^{10} s_i$.

Input Format

Each line of the input represents a game and contains a list of numbers recording how many pins were knocked down by each roll of the ball.

Output Format

Each line shows the scores s_i for frames $i = 1, 2, 3, \ldots, 10$ of the corresponding game in the input. The last number in each line is the total score. Right-justify each number output in a 5-column field, with `' = '` preceding the total.

Input Sample

```
6 4    10    10   6 3   9 1   8 1    10    10   8 2   8 2
 10   5 5    10   9 1   8 0   6 3    10    10    10    10
```

Output Sample

```
   10   20   40    9   19    9   38   57   67   77 =   346
   10   20   40   50    8    9   27   46   83  139 =   432
```