[O] Number Sequences and Laws

```
Time limit:   2 second
Memory limit: 262140 kBytes
```

Description

Mathematics has many laws that relate to different number sequences. One of them is Benford's law. Benford's law describes the relative frequency distribution for **leading digits** of numbers in many naturally occurring datasets.

If leading digits 1-9 had an equal probability, they would each occur about 11.1% of the time. However, Benford's law states that in many real-world sequences, approximately 30% of numbers start with 1, while less than 5% start with 9. According to this law, leading 1s appear roughly 6.5 times as often as leading 9!

This law has practical applications. For instance, it helped computer scientists uncover a bot network on social media. This was possible because, for most genuine users, the number of followers that their followers have adheres to Benford's law, whereas artificial accounts significantly veer from this pattern. Furthermore, Benford's law can be applied to detect if a country manipulates macroeconomic data, among other uses. However, it is important to note that many datasets do not conform to Benford's law.

Your task is to determine if the Fibonacci sequence complies with Benford's law. To do this, write a program that determines the first digit of each number in an n-element Fibonacci sequence and calculates the distribution of these first digits.

Input

The input file contains several lines (at most 8). Each line contains a positive integer n ($n \leq 10\,000\,000$), which defines the length of the Fibonacci sequence to analyze ($F_1 = 1, F_2 = 1, F_3 = 2, \ldots, F_n$).

Output

For each n, output:

- A line containing the value of n.
- A line for each digit from 1 to 9 consisting of:
  - The digit (right-aligned in 4 spaces), followed by a space and '|',
  - The count of numbers starting with that digit in the sequence (right-aligned in 10 spaces), followed by a space and '|',
  - The relative frequency of that digit, formatted to 2 decimal places (right-aligned in 9 spaces, including the decimal point and two digits).
- A blank line after each set of results.

Example

Input

```
5
10
```

Output

```
5
   1 |         2 |    40.00
   2 |         1 |    20.00
   3 |         1 |    20.00
   4 |         0 |     0.00
   5 |         1 |    20.00
   6 |         0 |     0.00
   7 |         0 |     0.00
   8 |         0 |     0.00
   9 |         0 |     0.00

10
   1 |         3 |    30.00
   2 |         2 |    20.00
   3 |         2 |    20.00
   4 |         0 |     0.00
   5 |         2 |    20.00
   6 |         0 |     0.00
   7 |         0 |     0.00
   8 |         1 |    10.00
   9 |         0 |     0.00
```