Practice Arena

Practice problems aimed to improve your coding skills.

Too tired to create a story - part II

LAB-PRAC-13_STRUC-NUM

Too tired to create a story - part II [20 marks]

Problem Statement
The input to the problem will be a single strictly positive integer n, followed by a floating point number eps denoting the *precision* we want in your output. You have to output the square root of that integer. However, you have to calculate the square root using the Babylonian method (often called the Hero's method as well). The method is described below. Use float variables for all your calculations.

Recall that you are given a strictly positive integer n, and a precision parameter eps which will be a floating point number. Store n in an int variable and eps in a float variable. The algorithm starts off by taking an initial guess of the square root, say x0 (the method to find x0 is given below). At each step, the algorithm updates the guess according to the following rule.

x1 = (x0 + n/x0)/2

i.e. the next guess is the average of the previous guess and the ratio of the number n and the previous guess. This is repeated till we have abs(xt - n/xt) < eps i.e. the absolute difference between the guess and the ratio of the number n and the guess is strictly less than eps, the precision parameter we gave you in the input.

You can show that once this happens, xt is very close to the square root of n. More precisely, one can prove a theorem that shows that actually if xt satisfies the above condition, then we must have abs(xt - sqrt(n)) < eps. You have to choose the initial guess x0 from the set {2.000, 20.000, 200.000, 2000.000, ….} i.e. two times powers of 10. The guess x0 is chosen such that its square is closest to n. In case squares of two numbers from this set are equally close to n, choose the smaller one as the initial guess x0.

In the first line of the output, print the value of the initial guess as a floating point number to three digits of precision (use the %0.3f flag in printf to achieve this). In the next line, print the square root you have computed, correct to 3 decimal places (use the %0.3f flag in printf to achieve this).

Caution

1. The initial guess must also be printed as a floating point number with 3 digits of precision.
2. Even though your inputs are integers, your output is not an integer. Use float variables for all calculations.
3. Do not try to cheat by using the math.h sqrt() function to calculate the square root. We are looking for the answer given by the Babylonian which will have errors depending on the precision parameter given to you. The math.h sqrt() function will give you a highly accurate output and hence will not match the expected output.
4. There are two lines in your output with no spaces.

EXAMPLE:
INPUT
441 1.0

```
OUTPUT:
20.000
21.025
```

Explanation: 400 is closest to 441 among (4, 400, 40000, etc). Note that the square root of 441 is exactly 21.000 but due to the precision constant being high, Babylonian method is returning an answer with error. This error will go down if we decrease the epsilon parameter.

Grading Scheme:
Total marks: **[20 Points]**

There will be no partial grading in this question. An exact match will receive full marks whereas an incomplete match will receive 0 points. Please be careful of missing/extra spaces and missing/lines (take help of visible test cases). Each visible test case is worth 1 point and each hidden test case is worth 2 points. There are 2 visible and 4 hidden test cases.

Start Solving! (/editor/practice/6256)