Practice Arena

Practice problems aimed to improve your coding skills.

- PRACTICE-02_SCAN-PRINT
- PRACTICE-03_TYPES
- LAB-PRAC-02_SCAN-PRINT
- LAB-PRAC-01
- PRACTICE-04_COND
- BONUS-PRAC-02
- LAB-PRAC-03_TYPES
- PRACTICE-05_COND-LOOPS
- LAB-PRAC-04_COND
- LAB-PRAC-05_CONDLOOPS
- PRACTICE-07_LOOPS-ARR
- LAB-PRAC-06_LOOPS
- LAB-PRAC-07_LOOPS-ARR
- LABEXAM-PRAC-01_MIDSEM
- PRACTICE-09_PTR-MAT
- LAB-PRAC-08_ARR-STR
- PRACTICE-10_MAT-FUN
- LAB-PRAC-09_PTR-MAT
- LAB-PRAC-10_MAT-FUN
- PRACTICE-11_FUN-PTR
- LAB-PRAC-11_FUN-PTR
- LAB-PRAC-12_FUN-STRUC
- LABEXAM-PRAC-02_ENDSEM
- LAB-PRAC-13_STRUC-NUM
  - Too tired to create a story - part I
  - Too tired to create a story - part II
  - Too tired to create a story - part III
  - Point Proximity
  - The Bisection Method
  - The pace is too fast
  - A Question on Quadrilaterals
  - The Trapezoidal Technique
  - Constrained Candy Crush
  - Major Mobile Madness
  - The Newton Raphson Method
  - The Palindrome Decomposition
- LAB-PRAC-14_SORT-MISC

The Newton Raphson Method

LAB-PRAC-13_STRUC-NUM

The Newton Raphson Method [20 marks]

Problem Statement
The Newton-Raphson method is a popular method for finding the roots of functions. It is a precursor to the Newton method for optimizing non-linear functions. Given a real-valued function $f : \mathbb{R} \to \mathbb{R}$, and an initial guess of the root x_0, the NR method iteratively improves this guess using the following update rule

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Then, using x_1, it obtains a (hopefully) better estimate of the root x_2 as

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

Thus, you can get x_{t+1} using x_t. You have to stop updating when $abs(x_t - x_{t+1}) < eps$ i.e. the absolute difference between two successive estimates is strictly smaller than eps where eps will be given to you. When the above happens, simply output x_{t+1} as your output.

The first line will contain n, a strictly positive number indicating the degree of the polynomial, followed by a space, followed by eps a floating point number (store it as a double). The second line will contain n+1 integers (may be zero or negative or positive), containing the coefficients of the polynomial from zero degree to max degree i.e. if the polynomial is a cubic (i.e. n = 3)

$$f(x) = a \cdot x^3 + b \cdot x^2 + c \cdot x + d$$

then we will give you the coefficients as
d c b a
The last line will contain x_0 the initial guess you should use. x_0 will not be an integer and you should store it as a double. Run the NR algorithm as shown above and give your output correct to 2 decimal places, using the %0.2lf flag in printf.

Caution

1. All coefficients of the polynomial will be integers but they may be zero or negative too.
2. You may use the fabs() function from math.h to compute the absolute value of a non-integral number.
3. The roots and intermediate values in your computations may be non-integral. Use double variables for all your computations.
4. Be careful while computing the derivative polynomial.
5. We assure you that if you follow the above rules correctly, you will never run into a divide-by-zero situation.
6. Be careful about extra/missing lines and extra/missing spaces in your output.

EXAMPLE:
INPUT
2 0.01
1 -2 1
3.00

OUTPUT:
1.01

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/6265)