Open Feedback Dialog

codility

Training center

Check out Codility training tasks

Demo ticket

Session

ID: demoNTZZPT-YXC
Time limit: 120 min.

Status: closed

Created on: 2014-03-20 16:08 UTC
Started on: 2014-03-20 16:08 UTC
Finished on: 2014-03-20 16:13 UTC

Tasks in test

1 | {} ~~CountNonDivisible~~

Task score

100%

Test score

100%

100 out of 100 points

MEDIUM

1. CountNonDivisible

Calculate the number of elements of an array that are not divisors of each element.

score: 100 of 100

Task description

You are given a non-empty zero-indexed array A consisting of N integers.
For each number A[i] such that 0 ≤ i < N, we want to count the number of elements of the array that are not the divisors of A[i]. We say that these elements are non-divisors.
For example, consider integer N = 5 and array A such that:

```
A[0] = 3
A[1] = 1
A[2] = 2
A[3] = 3
A[4] = 6
```

For the following elements:

- A[0] = 3, the non-divisors are: 2, 6,
- A[1] = 1, the non-divisors are: 3, 2, 3, 6,
- A[2] = 2, the non-divisors are: 3, 3, 6,
- A[3] = 3, the non-divisors are: 2, 6,
- A[6] = 6, there aren't any non-divisors.

Write a function:

```
def solution(A)
```

that, given a non-empty zero-indexed array A consisting of N integers, returns a sequence of integers representing the numbers of non-divisors.
The sequence should be returned as:

- a structure Results (in C), or
- a vector of integers (in C++), or
- a record Results (in Pascal), or
- an array of integers (in any other programming language).

For example, given:

```
A[0] = 3
A[1] = 1
```

Solution

Programming language used: Python

Total time used: 6 minutes

Effective time used: 6 minutes

Notes: correct functionality and scalability

Task timeline

16:08:35 16:13:52

Code: 16:13:52 UTC, py, final, score: **100.00**

```
01. def solution(A):
02.     nums = {}
03.     for a in A:
04.         if a in nums:
05.             nums[a] += 1
06.         else:
07.             nums[a] = 1
08.
09.     N = len(A)
10.     D = [0] * (2 * N + 1)
11.     for n in nums.keys():
12.         i = n
13.         while i <= 2 * N:
14.             D[i] += nums[n]
15.             i += n
16.
17.     ret = [0] * N
18.     for i, a in enumerate(A):
19.         ret[i] = N - D[a]
20.
21.     return ret
22.
```

```
A[2] = 2
A[3] = 3
A[4] = 6
```

the function should return [2, 4, 3, 2, 0], as explained above. Assume that:

- N is an integer within the range [1..50,000];
- each element of array A is an integer within the range [1..2 * N].

Complexity:

- expected worst-case time complexity is O(N*log(N));
- expected worst-case space complexity is O(N), beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

Analysis

Detected time complexity:
O(N * log(N))

test	time	result
example example test	0.050 s.	OK
extreme_simple extreme simple	0.050 s.	OK
double two elements	0.050 s.	OK
simple simple tests	0.050 s.	OK
primes prime numbers	0.050 s.	OK
small_random small, random numbers, length = 100	0.050 s.	OK
medium_random medium, random numbers length = 5,000	0.080 s.	OK
large_range 1, 2, ..., N, length = ~20,000	0.140 s.	OK
large_random large, random numbers, length = ~30,000	0.180 s.	OK
large_extreme large, all the same values, length = 50,000	0.230 s.	OK

Training center