Python Programming Assignment # 3

Instructions

For each of the following problems, create an error free efficient Python program. Each program should be submitted in a separate Python file respectively that follows a particular naming convention. (E.g. The Python program for Question 1 should be in .py file with name Assign3Answer1.py. The Python program for question 2 should be in .py file with name Assign3Answer2.py. The program should execute properly in PyCharm)

Please, make sure that you have PythonLabs properly installed and it is properly configured in PyCharm. Submit your assignment by Wednesday 31 October 2018 EoD.

Problems

Problem 1: Maximal Product of any Quadruplet (10 points)

A non-empty array A consisting of N integers is given. The product of quadruplet (P, Q, R, S) equates to A[P] * A[Q] * A[R] * A[S] ($0 \leq P < Q < R < S < N$).

For example, array A such that:

```
A[0] = -3 A[1] = 1 A[2] = 2 A[3] = -2 A[4] = 5 A[5] = 6
```

contains the following example triplets:

- (0, 1, 2,3), product is −3 * 1 * 2 * -2 = 12
- (1, 2, 4,5), product is 1 * 2 * 5 * 6 = 60
- (2,3, 4, 5), product is 2 *-2 * 5 * 6 = -120
- (0,3, 4, 5), product is -3 *-2 * 5 * 6 = 180

Your goal is to find the maximal product of any quadruplet.

This problem has two parts:

1. Create a list of **15 numbers** randomly using PythonLabs RandomList function. Modify the list such that around 50% of numbers are negative numbers.
2. Write a function:

   ```
   def solution(A)
   ```

   that, given a non-empty array A, returns the value of the maximal product of any quadruplet.

Note: Use sorting in your solution.

For example, given array A such that:

A[0] = -3 A[1] = 1 A[2] = 2 A[3] = -2 A[4] = 5 A[5] = 6

the function should return 180, as the product of triplet (0, 3, 4, 5) is maximal.

Write an efficient algorithm for the following assumptions:

- N is an integer within the range [4..100,000];
- each element of array A is an integer within the range [−1,000..1,000].

Problem 2: Average number of comparisons (10 points)

Write a Python program that contains function named `test_search` that will compute the average number of comparisons made by a successful linear search. Your function should take two arguments, size and ntests, which determine how long the test list should be and how many tests to run. For example, to search a list of 100 numbers 25 times the call to test_search would be

```
test_search(100,25)
```

Your function should create an empty list to hold the results of the tests. Use a loop to call isearch the specified number of times, making sure it searches for a random value known to be in the list. Figure out how many comparisons were made by each test and append the count to the list of results. Finally, use your mean function to compute the average number of comparisons.

Here is an example for 250 tests on a list of 1,000 numbers:

```
>>> test_search(1000,250)
```

512.384

The results show that on average it took 512 comparisons to search a list of 1,000 items.

Problem 3: Scrabble sort (10 points)

Write a Python program containing function named `scrabble_sort` that will sort a list of strings according to the length of the string, so that shortest strings appear at the front of the list (but words that all have the same number of letters will be arranged in alphabetical order). To test your function make a list of random words by passing 'words' to RandomList in PythonLabs, e.g.

```
>>> a = RandomList(20, 'words')
>>> scrabble_sort(a)
>>> a
['mum', 'gawk', 'wist', 'forgo', 'caring', ... 'unquestioned']
```