**Instructions**

1. Please solve the following problems using the language your interviewer asked you to use. Use Python to implement them.
2. We are looking for clean, well factored code.
3. Treat this problem as a library. No UI or file/console input is expected or required.
4. Provide a full suite of unit tests.
5. Add a README.md file that at least includes:
   * runtime versions
   * how to install any dependencies
   * how to run your tests from the command line
6. Please upload your code to github or other public code repo and send in a link.
7. It is expected that the candidate to complete the tasks in the allotted time requested by the interviewers.

**1) Pricing Problem**

We have a network of vendors who re-sell our products. We wish to provide them an application to calculate the total cost of an order.

The app needs to give volume discounts and include sales tax.

Another system will accept input from the user, and will call this component with 3 inputs:

* number of items
* price per item
* 2-letter province/state code

The application should output the total price.

The total price is calculated by:

* calculate the total cost for the items
* deduct discount based on the quantity
* add sales tax based on the province/state code

The following tables give the discount rate and tax rates:

|  |  |
| --- | --- |
| **Order Value** | **Discount Rate** |
| $1,000 | 3% |
| $5,000 | 5% |
| $7,000 | 7% |
| $10,000 | 10% |

|  |  |
| --- | --- |
| **Province** | **Tax Rate** |
| AB | 5% |
| ON | 13% |
| QC | 14.975% |
| MI | 6% |
| DE | 0% |

**Example 1:**

Command: run 500 1.00 ON

Input:  500 items, $1 per item, ON

Output: $565.00

**Example 2:**

Command: run 3600 2.25 ON

Input:  3600 items, $2.25 per item, MI

Output: $7984.98

**2) Odd Occurrences in Array Problem**

Find value that occurs in odd number of elements.

A non-empty array A consisting of N integers is given. The array contains an odd number of elements, and each element of the array can be paired with another element that has the same value, except for one element that is left unpaired.

For example, in array A such that:

A[0] = 9 A[1] = 3 A[2] = 9

A[3] = 3 A[4] = 9 A[5] = 7

A[6] = 9

* the elements at indexes 0 and 2 have value 9,
* the elements at indexes 1 and 3 have value 3,
* the elements at indexes 4 and 6 have value 9,
* the element at index 5 has value 7 and is unpaired.

Write a function:

**def solution(A)**

that, given an array A consisting of N integers fulfilling the above conditions, returns the value of the unpaired element.

For example, given array A such that:

A[0] = 9 A[1] = 3 A[2] = 9

A[3] = 3 A[4] = 9 A[5] = 7

A[6] = 9

the function should return 7, as explained in the example above.

Write an **efficient** algorithm for the following assumptions:

* N is an odd integer within the range [1..1,000,000];
* each element of array A is an integer within the range [1..1,000,000,000];
* all but one of the values in A occur an even number of times.

**3) Missing Integer Problem**

Find the smallest positive integer that does not occur in a given sequence.

Write a function:

**def solution(A)**

that, given an array A of N integers, returns the smallest positive integer (greater than 0) that does not occur in A.

For example, given A = [1, 3, 6, 4, 1, 2], the function should return 5.

Given A = [1, 2, 3], the function should return 4.

Given A = [−1, −3], the function should return 1.

Write an **efficient** algorithm for the following assumptions:

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

**4) Minimum Average Two Slice Problem**

Find the minimal of any slice containing at least two elements.

A non-empty array A consisting of N integers is given. A pair of integers (P, Q), such that 0 ≤ P < Q < N, is called a slice of array A (notice that the slice contains at least two elements). The average of a slice (P, Q) is the sum of A[P] + A[P + 1] + ... + A[Q] divided by the length of the slice. To be precise, the average equals (A[P] + A[P + 1] + ... + A[Q]) / (Q − P + 1).

For example, array A such that:

A[0] = 4

A[1] = 2

A[2] = 2

A[3] = 5

A[4] = 1

A[5] = 5

A[6] = 8

contains the following example slices:

* slice (1, 2), whose average is (2 + 2) / 2 = 2;
* slice (3, 4), whose average is (5 + 1) / 2 = 3;
* slice (1, 4), whose average is (2 + 2 + 5 + 1) / 4 = 2.5.

The goal is to find the starting position of a slice whose average is minimal.

Write a function:

**def solution(A)**

that, given a non-empty array A consisting of N integers, returns the starting position of the slice with the minimal average. If there is more than one slice with a minimal average, you should return the smallest starting position of such a slice.

For example, given array A such that:

A[0] = 4

A[1] = 2

A[2] = 2

A[3] = 5

A[4] = 1

A[5] = 5

A[6] = 8

the function should return 1, as explained above.

Write an **efficient** algorithm for the following assumptions:

* N is an integer within the range [2..100,000];
* each element of array A is an integer within the range [−10,000..10,000].

**5) Brackets Problem**

Determine whether a given string of parenthesis (multiple types) is properly nested.

A string S consisting of N characters is considered to be properly nested if any of the following conditions is true:

* S is empty;
* S has the form "(U)" or "[U]" or "{U}" where U is a properly nested string;
* S has the form "VW" where V and W are properly nested strings.

For example, the string "{[()()]}" is properly nested but "([)()]" is not.

Write a function:

**def solution(S)**

that, given a string S consisting of N characters, returns 1 if S is properly nested and 0 otherwise.

For example, given S = "{[()()]}", the function should return 1 and given S = "([)()]", the function should return 0, as explained above.

Write an **efficient** algorithm for the following assumptions:

* N is an integer within the range [0..200,000];
* string S consists only of the following characters: "(", "{", "[", "]", "}" and/or ")".