Quiz 1 (Practice # 2)

This quiz is **open-book and closed-internet**. Feel free to use any physical materials you have brought, but you may not access resources online (except funprog). Proctors will be available to answer administrative questions and clarify specifications of coding problems, but they should not be relied on for coding help.

Each problem is worth 25 points, for a total of 100 points.

You *must* submit your quiz via funprog before the deadline in order to receive credit. This quiz assumes you have Python 2.7 installed on your machine.

The `resources` directory contains **Python documentation** for commonly-used data structures.

Problem 1: `median`

Given a list of numbers, return the median. If the list has an *odd* number of elements, this is the *middle* element when the list is written in increasing order. If the list has an *even* number of elements, this is the mean (average) of the two *middle* elements.

Examples:

`median([1, 2, 3])` should return `2`.

Problem 2: `is_quasidrome`

A palindrome is a sequence of elements that is identical when read forwards or backwards. A *quasidrome* is a sequence that is either a palindrome or can become a palindrome by removing one element.

Given a string, return a Boolean indicating whether it is a quasidrome. Spaces are treated as any other character.

Examples:

`is_quasidrome("aa")` should return `True`
`is_quasidrome("aab")` should return `True`
`is_quasidrome("aa b")` should return `False`

Problem 3: `is_permutation`

Given two strings, return a Boolean indicating whether one is a *permutation* of the other.

Examples:

`isPermutation("aabc", "abca")` should return `True`
`isPermutation("aabc", "abc")` should return `False`

Problem 4: `count_triangles`

Three vertices `A`, `B`, `C` form a triangle if the edges `AB`, `BC`, and `AC` all exist. Note that `(A, B, C)` and `(B, C, A)` are the same triangle, and `AB` and `BA` are the same edge.

Implement `count_triangles(edges)`, which takes a *list of edges*, and returns the number of triangles that can be made with `edges`. Each edge is a 2-element lists `[string1, string2]`, where `string1` and `string2` are are names of two vertices connected by the edge.

You may assume that at most one edge exists between any two vertices.

Examples:

`count_triangles([["1","2"], ["2","3"], ["3","1"]])` should return `1`
`count_triangles([["1","2"], ["2","3"]])` should return `0`
`count_triangles([["1","2"], ["2","3"], ["3","4"], ["1","3"], ["2","4"]])` should return `2`