29 January 2017

Minimum Mutations!

A gene string can be represented by an 8-character long string, with choices from "A", "C", "G", "T".

Suppose we need to investigate about a mutation (mutation from "start" to "end"), where ONE mutation is defined as ONE single character changed in the gene string.

For example, "AACCGGTT" -> "AACCGGTA" is 1 mutation.

Also, there is a given gene "bank", which records all the valid gene mutations. A gene must be in the bank to make it a valid gene string.

Now, given 3 things - start, end, bank, your task is to determine what is the minimum number of mutations needed to mutate from "start" to "end".

If there is no such a mutation, return -1.

If the start and end string are the same, return 0.

Example:

start: "AACCGGTT"

end:   "AAACGGTA"

bank: ["AACCGGTA", "AACCGCTA", "AAACGGTA"]

return: 2

Note:

Starting point is assumed to be valid, so it might not be included in the bank.

If multiple mutations are needed, all mutations during in the sequence must be valid.

Organize your solution into a driver class (with a main method) and utility class containing the logical implementation.

Points will be docked for code that's too tightly coupled.

The starting and ending sequences and mutation bank may be (in descending order of points awarded) read from a file, input through the console, or hardcoded in the driver class.

Bonus: Implement JUnit tests on your utility methods. (this will require your solution to be a Maven project, not a simple Java project)