Problem Instructions

1. This is a timed exercise; however, it is on the honor system. We expect you to spend no more than two hours total over the next 48 hours to complete it. No need to rush. We don’t consider how long it takes you to complete the problem so please take the time to check your work thoroughly before submission.
2. If you do not finish within the time allotted, please submit what you have completed and provide an explanation in your email of what work remains to be done. If you have problems with your development environment, please notify us immediately via email by responding to this message.
3. We will accept solutions in **Java** or **Python**. Please use only standard libraries and write your code to be portable. You are free to use your favorite IDE, editor, and reference documentation.
4. Your solution will be judged primarily for its correctness. However, we also place significant emphasis on code clarity, design and efficiency.
5. If you have questions about the problem, please use your best judgment and carefully document your assumptions in the code.
6. When you have your final solution, please reply to this email with your solution as a .zip, .tar, or .tar.gz attachment. Do not include any binaries, only source code.

Product Spec

You have to test a program that generates 1000 lines. Each line consists of 2 numbers: line number and a random number, separated by a tab. Each random number (n) is ranging in value such that 100 <= n <= 500. For example:

random_numbers.txt:

```
1	315
2	289
3	109
4	500
…
```

Task

Write a test program in Java that takes a single command line argument, filename of the file to be parsed (*random_numbers.txt)* and verifies that the output satisfies the conditions stated above. Give each testcase a descriptive name and display it along with a Pass or a Fail result next to it, based on your validation. For failures, display every failure encountered on the next line(s) (if more than 1). For example:

```
ValidateFoo: Pass
ValidateBar: Fail
        Line <message>: Value <message>
        Line <message>: Value <message>
        Line <message>: Value <message>
        ...
ValidateMoo: Pass
```

Additionally, only if all of the above mentioned tests pass, the same test program should generate a text file named *difference.txt*, containing the same list of numbers and corresponding percentage difference (2-digit accuracy) between two neighboring numbers. If any of the tests failed, do not generate this file. For example:

1	315	*(Can say 0.00% or blank)*
2	289	-8.25% *(% difference between 315-> 289)*
3	109	-62.28% *(% difference between 289 -> 109)*
4	500	+358.72% *(% difference between 109 -> 500)*

…