CSW2 (Section 29)
Moodle Assignment
Date: 30-03-2024

General Instructions

- Solve the questions in separate files. Name of the public class and file should be question number[dot]java, for example, Q1.java, Q2.java, etc.
- Use exception handling and try-catch blocks whenever possible. Avoid declaring throws.
- Copied, shared and matching submissions will not be checked and graded.

Q1. (a) Take two existing text file names as input from user. If a file does not exist (use FileNotFoundException), prompt for correct file name again after showing a message.

(b) Also, prompt the user for an output file name. Check if it already exists (Use File exists()). In that case, display a message and ask for a new file name. Otherwise, create the output file.

(c) Line number *i* of the output file is concatenation of *i*-th line of the first and second input files separated by a space. Use empty string if a file contains less than *i* lines.

Line#	Inp1.txt	Inp2.txt	Out.txt
1	Hi,	Red.	Hi, Red.
2	I am from	Istanbul.	I am from Istanbul.
3		Turkey	Turkey
4		Asia	Asia

Q2. Take a filename (say, source.txt) and a sequence of unordered integers (say, 11, 0, 4, 6, -1) as input. Copy chosen line numbers only (i.e., 0, 4, 6, and 11) to an output file. Use a negative number (-1 in this example) to denote end of input. If line number exceeds maximum number of lines then throw IOException and exit.

Q3. You are given a text file as input (use String filename="input.txt" or from user; your choice.) It contains N file names (N lines). These files contain english stories.

(a) Use a map to count and store the aggregate frequency of words in the stories.

(b) Take another file called meaningless.txt as input which contains very common words like articles, prepositions and others, e. g., am, is, are, to, etc. Remove these words from the map. (Note: It is enough to test with say three words like a, an , the.)

Section 29

CSW2 (Section 29)
Moodle Assignment
Date: 30-03-2024

(c) Write the final map entries (w1:count1) in file high-freq.txt in order of descending frequencies, i. e., most common word in first line, second common and so on so forth.

Q4. Create your own checked SqrtNegativeException class. In another class write a method with the signature-

double square_root(double)

to find square root of doubles. This method should throw SqrtNegativeException if the passed number is negative. From main handle this exception using try catch and ask user for another input.

------e.o.f------

Section 29