CPSC 233 – Coding Challenge 5 – Practice 1

This is a practice coding challenge. You may ask other students and your TA any questions you wish. Note that you are expected to complete the actual coding challenge independently. Having a solution for this coding challenge will not help you come up with your own solution for the actual coding challenge. Only if you can solve this practice independently can you be confident that you can complete the actual coding challenge successfully. See previous coding challenges for rules for coding challenges and best practices for success.

Remember to always create a 'skeleton' first and compile and test this using the provided JUnit test. For this coding challenge, you will be asked to create one class. Make sure you create a skeleton of all classes first. You can upload your solution to WebCAT directly. You do not need to place it in a zip file.

Before a method ends, make sure to close any file that you have opened. Not doing so may cause tests to fail. If using Scanner, make sure that a method only has a single instance of Scanner open at a time. (Not doing so may also cause problems with the tests.)

Create a class called FileExercises that contains the following methods.

- Method parseNonNegativeInt that takes a string as an argument. It should convert the argument to an int and return the result. If the string does not contain an integer or contains a negative integer, the method should throw an IOException.

- Boolean method is3ByteRGB that takes a filename as an argument. The method should read the file as a binary file and assume it has the following format:
    - The file contains two integers. The first integer is the number of rows and the second integer is the number of columns. The row and column represents the size of the grid of pixels represented in the file.
    - The rest of the file contains pixels in order. First all the pixels for the first row, then the pixels in the second row, etc.
    - Each pixel contains three integers. The first integer represent the red intensity of the pixel colour, the second the green intensity of the pixel colour and the third the blue intensity of the pixel colour.
    - The method returns true if all colour intensity is a number that can be represented by one byte (and each pixel can be represented by 3 bytes). A byte can represent any number between 0 and 255.
    - If any exceptions occur while opening or reading the file, the method should return false.

- Method append that does not return anything and takes an array of int's and a filename as a String as arguments. The method should add the int (as integers, not strings) to the end of the file. This method should catch any exceptions but not do anything if an exception occurs.

- Method encrypt that does not return anything and takes a shift, input filename and output filename as arguments. It should read the input file as a text file and encrypt the content using a Caesar cypher with

the shift provided. The resulting text should be placed in the output file specified. (If the output file already exists, replace the existing file with a new file that contains the required content.) You can find a description of the Caesar cypher on-line. If a character in the file is not alphabetic, leave the character unchanged. Convert upper case character to upper case characters and lower case characters to lower case characters.