Assignment A4 (40 marks)

***Focus:** Binary I/O*

Q1. **[10 marks]** Write a Java program to track the total sum of integers across all program executions. You can store an `int` representing the sum in a file, e.g., `count.dat`. Each time your program is executed, ask the user for an integer, read the count from the file, sum the two values together, and then store only the sum back into the file. If `count.dat` doesn't exist (e.g., first time to run the program), store only the user input in the file. Use binary IO for your code. For this question, just write your code in the `main` method and declare the checked exceptions in the method's header.

Q2. **[10 marks]** Write a Hex Viewer method with the following header:

```
private static void viewHex(String filename)
```

The method reads bytes from `filename` and displays them in hex representation. The output should be formatted as in the example below, i.e., each line consists of 8 pairs of hex numbers, then '|', then another 8 pairs. Use `Integer.toHexString()` to convert a byte into a string representing the equivalent hex. Use a `try` statement to handle `IOException` and display a simple error message if an I/O exception occurs.

Sample run (for the attached data.dat file)

```
89 50 4E 47 0D 0A 1A 0A | 00 00 00 0D 49 48 44 52
00 00 00 02 00 00 00 02 | 08 06 00 00 00 72 B6 0D
24 00 00 00 01 73 52 47 | 42 00 AE CE 1C E9 00 00
00 04 67 41 4D 41 00 00 | B1 8F 0B FC 61 05 00 00
00 09 70 48 59 73 00 00 | 0E C4 00 00 0E C4 01 95
2B 0E 1B 00 00 00 1B 49 | 44 41 54 18 57 63 BC 7A
FA D8 FF 1F DF BF 33 30 | BD FA F0 9E E1 F3 AF 9F
0C 00 73 86 0B FE AA 7D | 68 4E 00 00 00 00 49 45
4E 44 AE 42 60 82
```

Q3. **[20 marks: 10 for each method]**

Develop a class `BackupRestore` that has two static methods:

```
public static int backup(String filename, double partSize)

public static int restore(String filename, int numberOfPieces)
```

The `backup` method backs up huge files by splitting them into smaller pieces that may fit into storage devices with different capacities (e.g., 700 mb for CD-Rs). The size of each piece is equal to `partSize` megabytes. The method creates the output files `filename.1`, `filename.2`, etc. and then returns an integer representing the number of files created.

The `restore` methods does the opposite. It combines `filename.1`, `filename.2`, etc. into `filename`. The method returns an integer representing the total size in MB of the file `filename`.

In both methods, if an `IOException` occurs, display an error message and return -1. Use buffer classes to improve performance.

Try your code on this pdf (A4_2024W.pdf). Split it into pieces of 0.05 MB (50 KB) size, delete the original file, and then combine your pieces back again and open the file to see if your code works.

Bonus (5 marks): develop your code for Q3 once with a buffered I/O stream and once without a buffer, and compare the results in terms of the time taken to backup and restore the course syllabus. The output should be similar to the sample below:

Without a buffer:

```
Backing up...
Restoring...
Done in 48495 milliseconds.
```

With a buffer:

```
Backing up...
Restoring...
Done in 188 milliseconds.
```

Notes on the bonus:

a) Use System.currentTimeMillis() to measure the time taken by your methods

b) To get a significant time difference between buffered I/O stream and unbuffered, *read one byte at a time.* Do not read into a byte array.

c) Don’t use input.getAvailable() method to check if you reached the end of the input file. Instead, use another method for checking the end of file.

Grading

- 15 % for logic explanation
- 70 % for proper code structure and logic
- 15 % for correct syntax and formatting

Submission Instructions

For programming questions, explain in few, simple sentences **the algorithm you used to tackle the problem**. Add these sentences as a **block comment at the beginning of your program**. For coding questions, make sure to use appropriate code formatting and structure (e.g., indentation, brackets, etc.).

For this assignment, you need to do the following:

1- Create a Java project of which name consists of your student number followed by the assignment number, e.g., “1234567_A2”.
2- Create one class for each question and write your answer inside that class. Your classes should have the same name as the question number (e.g., Q1)
3- After solving all questions, open Windows Explorer (or any other file explorer).
4- Navigate to your Java project folder (can be found inside your Eclipse workspace folder).
5- Locate the “src” folder for this project (the folder that includes the source code for all questions).
6- Zip the “src” folder and rename the zipped file to match your project name (e.g., 1234567_A2.zip).
7- Submit the zipped file to Canvas.

Note that you can resubmit an assignment, but the new submission overwrites the old submission and receives a new timestamp.