Due Date: Friday, Feb. 21st @ 11:59pm

Points: 100

This is an individual assignment.

Restrictions: you cannot use any methods from the Java Array(s) class to copy an array, check for equality, or otherwise manipulate an array.

You also may not call the Java System.arraycopy method

You must write the Java code to perform these functions.

In completing the list method, you will need to use (import):

- **the Java File class. You may use any of the methods that are available in the Java File class in developing your algorithm. In particular, you should need to use the:**
  - **java.io.File**
  - **java.io.FileNotFoundException**
- **the java Scanner class**
- **the javax.swing.JOptionPane class**
- **You may also use any of the class that you created in any previous assignment in this class. Specifically, if needed, you can use your:**
  - **ArrayBag,**
  - **ListBag, or**
  - **SinglyLinkedList (from textbook Code Fragments)**
- **classes if needed.**

Create a NetBeans project named Lab104 and ensure it is saved to a location like desktop or your flash drive. In the project you will do the following:

Create a Recursion Class that will implement the following recursive algorithms:

- Implement a recursive algorithm to compute the nth Harmonic number defined as:

$$\mathrm{H}_n = \sum_{k=1}^{n} \frac{1}{k}$$

- Implement Isabel's technique for summing the values in an array of n integers as described in problem C-5.24 on page 223 of the textbook.
  - The values to be summed by Isabel's technique should be stored in an ASCII text file. You can create this file using a text editor like Notepad.

  - The values should be white space delimited in this file.
  - To call to Isabel's technique your program needs to:
    - Request the path to the data file
    - Load the contents of the data file into an array
    - Call Isabel's technique
  - As described in the text, Isabel's technique only works for arrays that contain a power of 2 number of integers. If the number of integers in the data file are not a power of 2 your method must throw an appropriate exception.
- Implement a recursive method with the signature

  void list(String path)

  that prints all entries of the file system rooted at the given path. Note that a path would may look something like: C:\Windows\DigitalLocker. The list of entries should be displayed in the console and not in a JOptionPane.
- Because the list of entries can be very long this method will print the list of entries to the console rather than return a list of entries.

Create a Client Class that will fully test each of the recursive methods that you created above in an interactive menu fashion. This class should:

- Use a JOpitonPane to ask the user which algorithm they wish to test.
  - The user should be provided with a menu/list to select from
  - If the user selects an invalid option, the program should
    - Inform the user of their mistake and allow them to try again
  - The last option in the menu/list should allow the user to quit the program.
- For each recursive technique, use a JOptionPane to ask the user for any parameters/values needed for the test.
  - For Isabel's technique the parameter should be the path that contains a blank (white space) separated list of integers values that will make up the array.
    - When you read in the values from the file you should skip any values that are not integers.
    - The presence of one or more on-integer values in the input file does not automatically make the file invalid. If the number of valid integer values in the file is a power of 2 then it is a valid input file.
    - If the input file does not contain a power of 2 number of valid integers your program should inform the user of the problem and

ask them to enter a new filename or quit back to the algorithm selection menu.
        - If the user enters a file that does not exist your program should inform the user that the file does not exist and ask the user to enter a correct filename or quit back to algorithm selection menu.
        - The user should have the option of giving up on trying to run Isabel's technique and if the user selects that option they should be returned to the main menu.
- Output for each test should be shown in the system console and must include:
    - The name of the algorithm being tested.
    - The parameters entered by the user
    - Values used if different from parameters (e.g. the contents of the array read in for Isabel's technique).
    - Error messages give to the user (e.g. file not found)
    - The results of the test.
    - Note that the JOptionPane interactions will not be captured in the system console.
- At the end of a test the user should be given the options of running another test or exiting the program, i.e. after running one of the recursive algorithms the user should be returned to the main menu.

You may want to review Chapter 11 from Java Illuminated. A scanned copy of this chapter is available on Blackboard in Course Content -> Textbook Resources.

Things to turn in:

- Open a Microsoft Word document
- Copy and Paste the source code of the **Recursion Class** (make sure to use *Ctrl + A* to select all the source code of the program, *Ctrl + C* to copy, and Ctrl + V to paste.).
- Copy and Paste the source code of the **Recursion_Client** Class
- Copy and paste the output of the client program
    - You do not need to take screen shots of the JOptionPanes in action, you just need to show the contents of the system console.
- Next, zip the Project folder.
- Finally on blackboard, submit both your Word document and project zipped file.