Tasks

Recursion

Package `FilesInDirs` contains `FindFilesRecursive.java`. The purpose of the class is to recursively iterate through a given directory, and its subdirectories.

- Implement the method `findFiles(File file)` in a way, so that it increments the variable `numFiles`, for every "atomic" file (atomic meaning it has no sub-files), and simultaneously writes the full file path in the console, for each file incremented. If the method encounters a directory, it should instead increment the variable `numDirs` and make a recursive call (meaning `FilesInDirs` should be called in the method).

Manipulation af arrays

Within the package `ArrayManipulation` contains the class `ArrayManipulation.java`. Within the `ArrayManipulation.java`-class you will find the methods `evenOdd()` and `sort()`. The method `evenOdd()` should take an array of random numbers (between 0 - 100) as input.

Task A

The numbers must be arranged so that all odd numbers precede all even numbers. Write a method using the method signature `public int[] evenOdd(int [] array)`, which rearranges all the numbers in the array. For example:

`Input: [71, 1, 2, 68, 36, 59, 70, 22, 81, 89]`

`Output: [71, 1, 59, 81, 89, 22, 70, 36, 68, 2]`

```
Hint: You might want somewhere to temperarily store your numbers, while you are sorting them
```

Task B

For this task you are going to be implementing the method signature `private void sort(int [] array, int splitIndex)`. The point of the method is to sort the input array, in a manner so that it arranges the numbers of the within the array in ascending order.

The `int splitIndex` is supposed to denote where in the array, the sorting must start over from lowest to highest, in case you might have more than one type of numbers to be sorted. We now want to call this method from within the `evenOdd()`-method, as we want the two parts of the array to be sorted like they are underneath:

`Input: [71, 1, 2, 68, 36, 59, 70, 22, 81, 89]`

`EvenOdd: [71, 1, 59, 81, 89, 22, 70, 36, 68, 2]`

`Output: [1, 59, 71, 81, 89, 70, 68, 36, 22, 2]`

In the example above the `int splitIndex` is set to 5, in the context of a 5 number array.

```
Hint: You might want to check out the Arrays.sort() function for this
```

```java
package FilesInDirs;

import java.io.File;
import java.util.Scanner;
public class FindFilesRecursive {
    private int numDirs;
    private int numFiles;
    public static void main(String[] args) {
        // Prompt the user to enter a directory or a file
        System.out.print("Enter a directory or a file : ");
        Scanner input = new Scanner(System.in);
        String directory = input.nextLine();
        File startDir = new File(directory);

        FindFilesRecursive ffr = new FindFilesRecursive();
        ffr.findFiles(startDir);
        System.out.println("\n*************\n" + ffr);
    }

    // Exercise: If a file is a directory: Call all files recursively,
    // else print full path to the file. Count both dirs and atomic files.
    private void findFiles(File file) {
        if(file.isDirectory()){
            File[] files = file.listFiles();
            numDirs++;
            for(File file1 : files){
                findFiles(file1);
            }
        }
        if(file.isFile()){
            numFiles++;
            System.out.println(file.getPath());
        }
    }

    @Override
    public String toString() {
        return "FindFilesRecursive{" + "noDirs=" +
 numDirs + ", noFiles=" + numFiles + '}';
    }
}

```

```java
package ArrayManipulation;

import java.util.Arrays;
import java.util.Random;
public class ArrayManipulation {
    public static void main(String[] arg) {
        Random generator = new Random(222);
        int[] array = new int[10];
        for (int i = 0; i < array.length; i++) {
            array[i] = generator.nextInt(100);
        }
        System.out.println("Input:  " + Arrays.
toString(array));

        ArrayManipulation arrMani = new
ArrayManipulation();
        int[] result = arrMani.evenOdd(array);
        System.out.println("Output: " + Arrays.
toString(result));
    }

    public int[] evenOdd(int[] array) {
        //Vi har en temp-array som egentligt
definerer længden af den.
        int[] temparray = new int[array.length];
        //Her kan det ses, at vi har et startpunkt
for selve ulige numre.
        int oddnumber = 0;
        //Her kan det ses, at vi har en lige nummer
som er temparray længden.
        int evennumber = temparray.length-1;
        //For hvert tal i array har vi en nummer som
er modulus af indekset.
        for(int number : array){
            if(number % 2 == 0){
                temparray[evennumber] = number;
                evennumber--;
            }else {
                temparray[oddnumber] = number;
                oddnumber++;
            }
        }
        System.out.println("eventnumber: " + Arrays.toString(temparray));
        sort(temparray,oddnumber);
        return temparray;
    }

    private void sort(int[] array, int splitIndex) {
        Arrays.sort(array,0,splitIndex);

        //Nu skal vi til at sortere array'et.
        for(int i = splitIndex; i<array.length; i++){
            array[i] = -array[i];
        }

        //Nu skal vi sortere de negative tal.
        Arrays.sort(array, splitIndex, array.length-1);

        for(int i = splitIndex; i< array.length;i++){
            array[i] = -array[i];
        }
    }
}



```