Lab 01- 30 points

Overview

For today's lab you will create a well-commented java project that will contain a class called **`Lab_01_ArrayUtilities {…}`** with the following two static methods: **`buildIntArray(…)`** and **`swap(…)`**. Also include in your project a class called **`Lab_01_Tester {…}`** and include the **`main(…)`** method. I will give you till Tuesday (for Thursday Labs), and Wednesday (for Friday Labs) to complete the lab, but this will not always be the case. Check the Canvas discussion board for helpful hints, and answers to questions I get from the class.

Write the following code for the ArrayUtilities class methods:

1. **`buildIntArray(int length, int fromNum, int toNum)`**
    a. The buildIntArray() method should accept three parameters: 1] An array element length, 2] a "from number", 3] and a "to number" so that if a call to this method looked like: **buildIntArray(20,5,30)** an array of 20 random numbers between, and including, 5 and 30 would be returned.

2. **`swap(int[] nums, int i, int j)`**
    a. The swap() method will accept 3 parameters: 1] An int array; and 2,3] two variables that will represent the indices of the array whose values should be swapped. Due to the fact that java passes arrays by reference, anything you do to the array within the swap() method changes the original array.

Your Lab_01_Tester's main() method should do the following:

1. **Arrays and File IO**
    - Create a file (through code) called "Lab_01_nums.txt".
        - Hint:
          PrintWriter PW = new PrintWriter(new File("Lab_01_nums.txt"));
    - Use your method created in your ArrayUtilities class to create a 20 element array of random integers from 10 to 29. Note, in order to use this method you need to include the full class path: **Lab_01_ArrayUtilities.*buildIntArray*(…)**
    - Print the list of that array's values onto the screen AND save it to the text file. Output should look like my example on the next page. Consider using the toString method of the Arrays class: **Arrays.toString() - https://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html#toString(int[])**
    - Using a loop and the swap() method swap the value of even **indexed** elements with the next element in the array. That is, swap (the value in) element [0] with (the value in) element [1], element [2] with [3], and so on. Be careful when you reach the end of the array and try for Arrays of various sizes (even and odd sized arrays) .
    - Print the list of array values to screen again and append it to the text file.
    - Sort the array using Arrays.sort(*int array*) https://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html#sort(int[])
    - Print the list of array values again and append it to the text file
    - Close the file

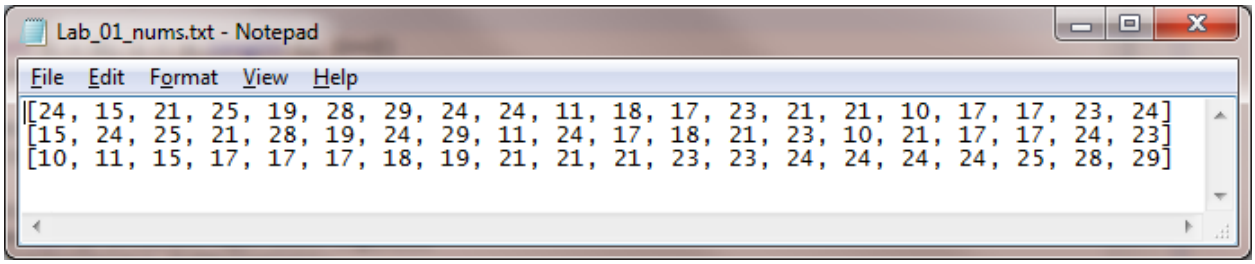
2. **Odd Numbers**
    - Iterate through the created array of numbers from above and count the number of odd values.
    - Print the output to the user with a understandable description

3. **Vowels, Consonants, and Other characters**
   - Manually create a text file in the root workspace folder called **words.txt** and include the following lines:

     apple
     banana
     grapes
   - Write the code that opens the file, reads each line, and counts the number of **vowels (A, E, I, O, U)**, **consonants** and **other characters**.
     - Hint: Scanner in = new Scanner(new File("words.txt"));
     - while(in.hasNext()) { …
     - Hint: To determine if a character is an "other" character, you can start by determining if the specific character is or is not a letter. The isLetter method of the Character class might be helpful: link here
     - Also helpful, especially if you're scanning character by character, is the Character.isWhitespace method.
   - Using the printf() function to format your output display a message to the user with the results.
     (you should get: **7 vowels**, **10 consonants**, and **0 other characters**)
   - I will be testing this using my own **words.txt** file so be sure you cover all scenarios of possible text data. DO NOT SUBMIT or create a words.txt file!

Sample Console and Text File Output:

```
[24, 15, 21, 25, 19, 28, 29, 24, 24, 11, 18, 17, 23, 21, 21, 10, 17, 17, 23, 24]
[15, 24, 25, 21, 28, 19, 24, 29, 11, 24, 17, 18, 21, 23, 10, 21, 17, 17, 24, 23]
[10, 11, 15, 17, 17, 17, 18, 19, 21, 21, 21, 23, 23, 24, 24, 24, 24, 25, 28, 29]
Odd values: 13
There were 7 vowels, 10 consonants and 0 other characters in the text file.
```



Grading Rubric:

```
==========================================================================================================
 5 pts - File names, method names, commented code, proper turn-in (FYI: 0/30 pts for late turn-in)
10 pts - buildIntArray() and swap() methods are free from coding errors and written well
15 pts - main() method code is free from coding errors and written well
==========================================================================================================
```

- Files were named correctly
- Turn-in process was followed
- Files turned in on time (no points for assignment for late work)
- Code is well documented and methods named correctly

- buildIntArray()
  - Method returns an int array of random values which also includes fromNum and toNum

- swap()
  - Method is void and correctly swaps two elements from the array

- main()
  - Uses PrintWriter() to write data to the file: Lab_01_nums.txt
  - Data is printed to the console like my example
  - A loop is used to swap every other adjacent elements of array

- **The number of odd values from the array is correct and printed to console**
- **Scanner object used to read words.txt file (do not include your words.txt in turn-in)**
- **Correctly counts the number of vowels (A,E,I,O,U), consonants, and other characters**
- **Uses printf() to print a well formatted string with these values**
- **Produces efficient coding (i.e., don't write 30 lines of code when it should take 5)**

Turn-In Process:

Turn in a .zip of your "src" folder containing your 2 .java files to Canvas by your respective due date and time shown above, and on Canvas. **ENSURE THAT YOU TURN IN .java files only in the zip and they compile.** No late labs will be accepted. Grading will be based on conforming to the standards we reviewed in class as well as following the requirements of this lab.