UF UNIVERSITY of FLORIDA
Herbert Wertheim
College of Engineering

COP 3502

Character Sorter Project

Overview

This character frequency count projects aims to improve your understanding of strings, loops, methods, type casting, arrays, and sorting. This project will expose students to a real world scenario they could encounter as a programmer and will develop problem solving skills.

Scenario

You now work for a company that is trying to determine character patterns in all english documents. Your goal is to create a program that will read in textual input from the user, analyze the frequency of each occurring character and then output to the user a sorted representation of their output based on their menu choice. You can expect a full page of characters to be sorted, so your program should be able to handle large input not just one line. (This just means that the string you will be reading in is very long; it will not have multiple new line characters.)

Structure

You must implement three methods to complete this project; but of course one can implement more as they see fit. The three methods that must be implemented are as follows:

```
        // The return type and parameters you need are up for you to decide.
        // As there are numerous ways to solve the same problem we will not
        // confine you certain data types being passed in and returned.


public static <return type> alphabeticalSort (Array/String)
{
        // The goal of this method is to sort all of the characters
        // in alphabetical order.
}


public static <return type> frequencySort (Array/String)
{
        // The goal of this method is to sort all of the characters
        // from highest frequency to lowest.
}


public static <return type> charTypes (Array/String)
{
        // The goal of this method is to sort the data into four categories:
        // Textual characters, Numeric characters, WhiteSpace characters,
        // and Symbolic characters
}
```

Key Points and Hints:

- Any **built-in sorting features are off limits** as the purpose is to become familiar with the logical process behind sorting.
- Make sure to utilize arrays, as this project will help you learn how to manipulate them.
- Feel free to use the built-in Character library functions that will help with some parts of the program.
- As should be standard for all of your programs, make sure to code for potential bad input from the user.
- Hint: Remember all characters are also represented by numbers (i.e. character 'A' is also represented as decimal number 65). Look up an ASCII table to see how all the characters relate. One could potentially convert characters to their decimal numeric value to aid them in sorting and counting.
- Alphabetically speaking, as per the ASCII table, symbols and numbers come before capital characters and lowercase characters come last, again reference the ASCII table for guidance.
- When sorting by frequency you may run into some ties (i.e two characters with frequency of 7), in this case which ever character comes first in ASCII order should come first.
- For the character types method, the sum of the all the types should equal the sum of the total characters input; if they are off, then some of the characters are not being categorized properly.
- You can make your main method very clean by using methods properly and abstraction most of the computation and analysis into the methods. Of course make sure to follow decent coding standards to make your code readable and use comments where necessary.

Sample Output

Here is what the running program should look like, make sure everything lines up exactly as shown:
In this particular example, only one line of input is sorted just to show an example; we will provide more sample text to be sorted later.

```
Welcome to Character Sorter Program
Please input a string to be sorted
AAAAbbbbccdd  42424242  &&%%$#@  (COMMENT: All user input is bolded and red)

Please select the option you would like to see

1. Display character frequencies alphabetically
2. Display sorted frequencies
3. Show types of character frequencies
4. Exit
1
  freq: 2           (COMMENT: This looks like no character has been displayed, but in fact its the Space)
# freq: 1
$ freq: 1
% freq: 2
& freq: 2
2 freq: 4
4 freq: 4
@ freq: 1
A freq: 4
b freq: 4
c freq: 2
d freq: 2


Please select the option you would like to see

1. Display character frequencies alphabetically
2. Display sorted frequencies
3. Show types of character frequencies
4. Exit
2
The sorted by frequency characters are:

2 freq: 4
4 freq: 4
A freq: 4
b freq: 4
  freq: 2
% freq: 2
& freq: 2
```

```
c freq: 2
d freq: 2
# freq: 1
$ freq: 1
@ freq: 1


Please select the option you would like to see

1. Display character frequencies alphabetically
2. Display sorted frequencies
3. Show types of character frequencies
4. Exit
3
Textual Character count: 12
Numerical Character count: 8
WhiteSpace Character count: 2
Symbol Character count: 7


Please select the option you would like to see

1. Display character frequencies alphabetically
2. Display sorted frequencies
3. Show types of character frequencies
4. Exit
4

Character Sorter Exited Successfully
```

FOR BAD INPUT:
If bad input is entered, whether it be an integer not 1-4 or an invalid data type just print:

```
        Error, bad input, please enter a number 1-4
```

Submission Guidelines

Make sure your output matches precisely with the sample out provided above.
Name your file CharSorter.java, make sure to ZIP your SRC with the .java file inside.
Submit on Canvas under Project 2.