CSCI 2100: Functional Programming
Topic: File Operations
Worksheet04_07_22

Full Points: 25 **Due by: April 14, 2022**

Answer the questions (that are not source codes or sample outputs) in a file called `worksheet4_07_22_answers.txt/pdf`.

Tasks:

1. Create two directories (`dataset` and `fileOperations`), at some place in your local machine.

   For example, I keep my Haskell code here:

   - `/Users/fhamid/Desktop/NCF/FunctionalProgramming/Code/`

   So, I created the directories in this location.

   - `/Users/fhamid/Desktop/NCF/FunctionalProgramming/Code/fileOperations`
   - `/Users/fhamid/Desktop/NCF/FunctionalProgramming/Code/dataset`

   We will create source codes for today's lab in the `fileOperations` directory and datafiles in the `dataset` directory.

2. Now, put the `numbers.dat` file in the `dataset` directory and the `part0.hs` in the `fileOperations` directory.

3. Run the program. Possible commands:
   - `runhaskell part0.hs` OR
   - `runghc part0.hs` OR
   - `ghc --make part0.hs` and then `./part0`

4. Observe the output. Check the dataset directory.

5. Run the source code (`part0.hs`) several times. Observe the output.

6. **[3 points]** Briefly state what happens when you run `part0.hs`. Record your answer on `worksheet04_07_22_answers.txt/pdf` file.

7. **[7 points]** Copy `part0.hs` as `part1.hs` and modify the instructions so that your program adds all the numbers after reading them from the `numbers.dat` file and prints the summation on the terminal.
    a. You may find the following information useful:

```
Prelude> lines "1\n2\n3\n"
["1","2","3"]
Prelude> read "1" :: Float
1.0
```

    b. You may need to use the `let` and `lambda` expressions.
    c. You may comment out the `writeFile` instruction.

8. **[10 points]** Copy `part1.hs` as `part2.hs` and modify the instructions so that your program does the followings:

`inputList:` Reads the numbers from the `numbers.dat` file (code is already available).
`outputList:` Creates a new list where every value is the addition of two consecutive numbers from the inputList. Assume, it is a left associative operation. You may define a new function that solves the task.

$$f(x_1, \dots x_n) = x_1,\ x_1 + x_2,\ x_2 + x_3, \dots, x_{n-1} + x_n$$

Sample:

```
inputList: []
outputList: []

inputList: [1]
outputList: [1]

inputList: [1, 2, 3]
outputList:[1, 3, 5]

inputList: [1, (-10)]
outputList: [1, -9]
```

Write the `outputList` in a new file named as `part2_output.dat`. Make sure that the new file in stored in the `dataset` directory.

Hint: You may find the following functions (`show` and `unlines`) useful:

```
Prelude> show 1
"1"
Prelude> show 1.5
"1.5"
Prelude> map show [1.5, 3, 6]
["1.5","3.0","6.0"]
Prelude> unlines $ map show [1.5, 3, 6]
"1.5\n3.0\n6.0\n"
```

Sample input	Sample output
1 112 -12 13 11111 -5 12 17 191 -111 2 17	1.0 113.0 100.0 1.0 11124.0 11106.0 7.0 29.0 208.0 80.0 -109.0 19.0

9. [5 points]

One of the very useful file modes is **AppendMode.** It lets us

- open (if the file already exists) or create a file and
- insert new data to the end of the file.

We may use `appendFile` function to access a file in append mode.

`appendFile` has a type signature that's just like `writeFile`, only `appendFile` doesn't truncate the file to zero length if it already exists but it appends stuff to it.

Now create a new Haskell program (one with `main`) called `part3.hs` that does the followings:

- Takes a string as input.
- If the string is null, then the program stops.
- Otherwise, it counts the total number of vowels in the string and saves the information in a file named as `part3_output.dat`
- The program repeats the same process until the user enters a null string (meaning, until the user presses the ENTER ← key).

The expected format of input and output is shown in the following table:

bash-3.2$ runghc part3.hs Enter a string: Hi there! Enter a string: Are you busy? Enter a string: NNNNN Enter a string: sure thing :) Enter a string:	-- part3_output.dat Hi there! >> 3 Are you busy? >> 5 NNNNN >> 0 sure thing :) >> 3 I need to TEST it soon >> 8

bash-3.2$ runghc part3.hs Enter a string: I need to TEST it soon Enter a string: bash-3.2$	

Submit:

Place all the following files in a folder called worksheet04_07_22_Lastname, zip it, and upload. Lastname should be replaced with your lastname.

Worksheet04_07_22_answers.txt/pdf
part1.hs
part2.hs
part3.hs
part0_output.dat
part2_output.dat
part3_output.dat

---------- That is all for today ☺ ----------