CSCE 314 [Section 100] Programming Languages – Summer 2017

Anandi Dutta

Assignment 3

Assigned on Tuesday, 20th June, 2017

Electronic submission to eCampus due at 23:59, Tuesday, June 27th, 2017 By electronically submitting this assignment to eCampus by logging in to your account, you are signing electronically on the following Aggie Honor Code: "On my honor, as an Aggie, I have neither given nor received any unauthorized aid on any portion of the academic work included in this assignment."

Note 1: This homework set is individual homework, not a team-based effort. Discussion of the concept is encouraged, but actual write-up of the solutions must be done individually.

Note 2: Submit electronically exactly one zipped file, namely, yourLastName-yourFirstNamea3.zip, that zip file will contain two files: a .hs file and a pdf file on eCampus.tamu.edu. We are not providing any skeleton code for this assignment.

Note 3: Please make sure that the Haskell script (the .hs file) you submit compiles without any error when compiled using the Glasgow Haskell Compiler (ghc), that is installed in the departmental servers (linux.cse.tamu.edu and compute.cse.tamu.edu). If your program does not compile, there is a chance that you receive zero points for this assignment.

Note 4: Remember to put the head comment in your file, including your name, UIN, and acknowledgements of any help received in doing this assignment. You will get points deducted if you do not put the head comment. Again, remember the honor code.

[1] A3×3 grid containing all the integers 1 to 9 is called a magic square if every row, every column, and both diagonals all add up to 15, as in:

2	7	6
9	5	1
4	3	8

Suppose that 3×3 grids are represented using the following types:

```
type Grid = Matrix Int

type Matrix a = [Row a]

type Row a = [a]
```

and that you are given functions

```
rows :: Matrix a -> [Row a]

cols :: Matrix a -> [Row a]

diags :: Matrix a -> [Row a]
```

that extract the rows, columns and diagonals from a matrix.

a) Define a function sort :: [Int] -> [Int] that sorts a list of integers into numeric order, using a sorting method of your choice. [6 points]

b) Using sort, define a function isValid :: Grid -> Bool that decides if a grid is valid, in the sense that it contains all the integers 1 to 9. [6 points]

c) Define a function isMagic :: Grid -> Bool that decides if a grid is magic, in the sense that all rows, columns and diagonals sum to 15. [6 points]

d) Using the library function replicate :: Int -> a -> [a], define the matrix choices :: Matrix [Int] that contains [1..9] in every cell. [6 points]

e) [6 points] Define a function cp :: [[a]] -> [[a]] that returns the Cartesian product of a list of lists, e.g. cp [[1,2,3],[4,5,6]] should give:

[[1,4],[1,5],[1,6],[2,4],[2,5],[2,6],[3,4],[3,5],[3,6]]

f) Using cp, define a function collapse :: Matrix [a] -> [Matrix a] that collapses a matrix of choices into a choice of matrices. [6 points]

g) Using your answers to the previous parts of this question, define the list magics :: [Grid] of all possible magic squares of size 3×3. [6 points]

h) Submit a pdf file named hw3comment.pdf. In that file, describe how you can make your Sudoku Puzzle solver more interactive and more efficient. Describe your plan for implementation (you can mention the Haskell packages for doing that). [8 points]

NB: This is an open-ended assignment. Feel free to use anything from Haskell to solve this problem. **In your comment, describe how you test your functions.** Provide some examples. Also, you can think of it as a Sudoku Puzzle Solver. You can also consider the option from taking the puzzle from the user, and showing the result to user. Monads!!! This part is optional. Think about how you can solve a large Sudoku puzzle more efficiently? Parallel Programming? If we wish you can try to write your code following that approach too. But, again this is optional. Have fun!