COMP2209: Programming III

Coursework Report

Xiaoke Li / xl5u20@soton.ac.uk / 31951473

Overview

6 questions completed.
The second question took a little longer to run and I am not sure if I could finish it within the time limit. All the examples given in the PDF have been passed and I have tried to write some tests and passed them too.

Statement:

I find samples of code on the web providing similar behaviour to these challenges. Within reason, I incorporate, adapt and extend such code in my own implementation. And I have indicated the source in both the report and the code

Challenge 1

The “calcuBBInteracions” function starts by checking for errors in the input such as an empty edge position or a non-positive value for n. If there are no errors, it calculates the interaction between each edge and atom by calling the “calcSingleInteraction” function.
The “calcSingleInteraction” function takes an edge position and calculates the interaction by calling the initial function. The initial function generates the initial position and direction based on the side of the grid and the entry point.

Challenge 2

I found this problem to be very similar to the Eight Queens problem, the restriction became just a matter of satisfying the “calcuBBInteracions” function, so accordingly I used the backtracking method. So c2 problem is equated to the n-queen problem, starting with an atom and gradually increasing the atom, outputting when a possible solution was found However, because Challenge 1 is not well optimized, Challenge 2 uses the "calcuBBInteracions" function for each recursion, resulting in a not good enough runtime.

https://literateprograms.org/eight_queens_puzzle__haskell_.html
https://gist.github.com/tronje/cf1ec8f05b854ff488760dce396a1f5f
https://wiki.haskell.org/99_questions/Solutions/26

Challenge 3

The stack overflow has sample of code to convert lambda to alpha normal form. I incorporate, adapt and extend such code in my own implementation within reason.

https://stackoverflow.com/questions/40316605/implementing-alpha-equivalence-in-haskell

Challenge 4

It is similar to the reference given by the cw PDF, just follow them. reference below

COMP2209 lecture on Parsing,
The monadic parser tutorial by HuAon in Chapter 13 of his Haskell textbook,
on-line tutorial: http://www.cs.nott.ac.uk/~pszgmh/pearl.pdf

Challenge 5

There is only one solution, Just follow the cw pdf explanation

Challenge 6

Lessons37, 38, 39 have inspired me

References:

GhatGPT
https://stackoverflow.com/questions/40316605/implementing-alpha-equivalence-in-haskell
https://literateprograms.org/eight_queens_puzzle__haskell_.html
https://gist.github.com/tronje/cf1ec8f05b854ff488760dce396a1f5f
https://wiki.haskell.org/99_questions/Solutions/26
COMP2209 lecture
Programming in Haskell by Graham Hutton
on-line tutorial: http://www.cs.nott.ac.uk/~pszgmh/pearl.pdf