COMP2209

Coursework report

01.01.2023 gdi1u21@soton.ac.uk- Georgi Iliev

Using different strategies to develop each exercise tested our knowledge of the material we were learning during this course. Tackling the more difficult problems involving Lambda calculus was time consuming, but well worth it.

I started by reviewing all the lectures that were needed to complete the exercises. I also borrowed "Programming in Haskell by Hutton", which took a key role in introducing me to the level of knowledge required to complete the assignments.

For the first exercise, I looked at various web solutions to the well-known "Black Box" game. I also played a couple of online Black Box games which helped to understand the basic concepts. I tested my implementation, which reassured me that the calcBBInteractions function could handle a variety of test cases and return the correct results for them at the same time.

The second exercise was the only one for which I had to carefully consider the algorithm so that the function would execute in the expected - less than 15 seconds time period. I repeatedly tried to apply the brute force algorithm, but unfortunately I was unable to complete the solution to this exercise.

In the third exercise, the combination of several functions contributes to obtain the desired function, which generates a more understandable version of the lambda expression that is in alpha normal form. By adding more test cases, I made sure that the "prettyPrint" function could handle different values satisfying the assumptions stated in the exercise description.

In the fourth exercise, I implemented a monadic Parser that parses a symbolic representation of an arithmetic expression and returns the corresponding abstract syntax tree. I found the recursive evaluation useful. Further testing ensured that the error handling in the code met the requirements of the exercise.

For the fifth exercise, which was based on recursion, the solution was limited by the maximum stack size. However, for too large arithmetic expressions, the recursion did not complete due to a stack overflow error. This could be avoided by using the combinator Y " Y = f -> ( x -> f (x x)) (x -> f (x x)) ". Basic arithmetic operations and larger numbers were additionally tried in the tests.

In the sixth and final exercise, innerRedn1 applied an inner-leftmost lambda expression reduction strategy. It takes a lambda expression and performs a single reduction of that expression, if possible, returning the reduced expression.

I took advantage of several tools, such as VSCode and its ability to suggest developing a more readable format when coding, and also suggestions for making the code clearer when installing appropriate extensions and plugins. On the other hand, the terminal also played a crucial role in developing my solutions. Finally, GitHub and the OpenAI chat helped to better understand the goals and objectives of the exercises.

Haskell's lazy and functional approach to programming expanded my programming skills and developed an interesting functional way of looking at certain tasks. The more I get into Haskell, the more I realize how everything around us can be built in a functional way.  
  
  
  
Sourcing List:

1.In the first exercise I helped myself with the following past solution:

Kırgöz, U.C. (2022) *UCKIRGOZ/comp2209-cwk*, *GitHub*. Available at: https://github.com/uckirgoz/COMP2209-cwk (Accessed: January 1, 2023).

2.For the second exercise, which I was unable to complete, I used a feature similar to one of last year's. I also tried to apply the Alpha-Beta Pruning algorithm, but unfortunately this was a failed attempt. This is the quote for the second exercise:

Hristov, E. (2022) *Haskell-coursework/challenges.hs at master · emiliyanhristov/haskell-coursework*, *GitHub*. Available at: https://github.com/emiliyanhristov/Haskell-Coursework/blob/master/Challenges.hs (Accessed: January 1, 2023).

3.For the fourth exercise I took some ideas from:

Hu, A. (2019) *Parsing simple arithmetic expressions in Haskell. based on https://two-wrongs.com/parser-combinators-parsing-for-haskell-beginners.html*, *Gist*. 262588213843476. Available at: https://gist.github.com/andrewhu-uw/10b44dbe4bc5eb817ab3f0329cbb902a (Accessed: January 1, 2023).

4.I also used the OpenAI chatGPT software, which helped me understand and interpret the requirements stated in the coursework correctly. The software also helped with the various test cases and was generally helpful in developing the exercises.