COMP 2209 - Programming 3
Coursework Challenges

James Palmer - jp14g23

December 17, 2025

1 Introduction

In this assignment I completed 6 programming challenges, and this report will outline the process, and my development strategy. Throughout the process of solving these problems, I used decomposition to break down them into smaller parts, which aligns neatly with the style of functional programming. For writing and testing, I used VSCode, printing in the main function to test my functions.

I performed unit testing as I went, ensuring that each function worked correctly before I incorporated it into the overall flow of the program. This fits into my overall philosophy of breaking the main function into individual smaller functions, with the main function simply calling them to put them together.

My original plan was to approach these challenges in order, from 1 through to 6. However, due to their difficulty I adapted this plan. Challenges 1,3 and 5 went smoothly, but when I got stuck on 2, 4 and 6, I would move on to another challenge, then come back for a new perspective.

2 Computing ray interactions

In this challenge I focused heavily on abstracting the problem, and focusing on easier parts of it first. I decided to get representation of the grid first, before thinking about the logic behind the travel of the ray.

3 Solving Black Box

This challenge came with significant roadblocks regarding the efficiency of finding a solution. Reading the paper "Black Box The Ultimate Game of Hide and Seek" [1] was extremely useful, and I adapted the method of using likelihood scores to find atom configurations with more insight.

4 Unparsing ASTs

I found initially understanding the lambda calculus tricky, but this video [2] was very helpful in understanding the concept behind it. I then read this paper [3] from the University of Dallas until I was confident enough to work on the challenge. Again I used decomposition to break the challenge down into just unparsing each type of expression.

5 Parsing lambda expressions

I removed as much complexity as I could with this, with removing whitespace, and handling the definitions part seperately, however I could not find a solution to bypassing the infinite recursion.

6 Translating to CPS

In order to gain an understanding for Continuation Passing Style, I used this [4] lecture from the Haskell Love conference given by Jeremy Gibbons.

7 Reductions

In this section I relied heavily on lecture 26 [5], as it provided a lot of guidance on evaluation functions. I chose to adapt the functions shown in the lecture and altered them to meet the requirements of the task.

References

[1] Joey Shepard, Archie Monji, and Helio Tejeda. *Black Box: The Ultimate Game of Hide and Seek*. UCI iCAMP Summer Research Program, supported by NSF Grant #DMS-0928427. Aug. 2013.

[2] Computerphile. *Understanding Lambda Calculus*. Accessed: 2025-01-09. 2025. URL: `https://www.youtube.com/watch?v=eis11j_iGMs`.

[3] Vijay Gupta. *An Introduction to Lambda Calculus*. `https://personal.utdallas.edu/~gupta/courses/apl/lambda.pdf`. Available at `https://personal.utdallas.edu/~gupta/courses/apl/lambda.pdf`. n.d.

[4] Konfy. *Jeremy Gibbons - Continuation-passing style, defunctionalization, and associativity*. Accessed: 2025-01-09. 2025. URL: `https://www.youtube.com/watch?v=8gnhaE2nmQ0`.

[5] Julian Rathke. *Interpreters: Writing an Interpreter in Haskell*. Lecture notes for COMP2209 - Programming III. 2025.