CS51 Final Project

Samuel Kim

May 2023

1 Extension: Lexical Scope

For my first extension, I modified eval_d to manifest lexical scope instead of dynamic scope. The result was my implementation of eval_l, which in the end, I found to have many similarities with the dynamic evaluator.

Due to the large overlap between the two evaluators, I created a general helper function that encapsulates their similarities while also accounting for the differences between dynamic and lexical evaluation. One of the primary differences that I had to consider when implementing the lexical environment evaluator was function evaluation. According to lexical semantics, a function is evaluated within the environment that existed at the point the function was defined. This environment in encapsulated within a closure along with the function expression in order to ensure that newly defined variables do not change the function evaluation. Within a dynamic environment, previously defined variables can be updated in this manner which can ultimately alter the function evaluation depending on when it is applied. The second difference between the two evaluators is in their treatment of recursive ("Letrec") expressions. In lexical environments, we have to manually update the environment each time we reach f when evaluating the function definition. Thus, following the approach outlined in the readme, I extended the environment with the value "Unassigned" as the value of the variable being recursively defined. Then, the function definition is evaluated within this extended environment. The resulting value is used to replace the value for the variable name and the function body is then evaluated within this new environment. Finally, the lexical and dynamic evaluators differ in their implementation of function application. This is what results in the different values of the final two test cases within the unit tests for these evaluators. Since in a lexical environment, variables within a function will always evaluate to the value they held at the time the function was defined, the function evaluation will return a closure of the function expression and the environment at that time. Once the function is applied, this lexical environment is used in evaluating the function body. The eval_helper function uses the is_dynamic boolean to check which evaluator was called and apply the appropriate lexical/dynamic semantic rules accordingly.