Project Report

Checkpoint Two

Kyle Monteiro- kmonte04@uoguelph.ca -1005077

Frank Maciel- fmaciel@uoguelph.ca -0999850

What has been done for this checkpoint?

In regards to checkpoint two numerous tasks have been completed.

We have worked on the symbol tree parsing which successfully prints out the tree structure for the C - language. This consisted of visit functions which were implemented to correctly handle the tree at various stages of parsing.

There was also the addition of the -s and -a flag in the program. The -s symbol is meant to represent the word “symbol” and which when enabled displays the symbol tree. The -a symbol stands for abstract symbol tree and when activated displays precisely this.

The type checking error handling was also performed on the symbol tree parser. All of the listed type checking errors in the marking scheme were performed: checking the array’s range/index, checking if types of both sides of the operand match, checking that two operands themselves also match, ensuring that the function return type matches the function return value, testing the conditions within if and while statements and ensuring that they are of boolean/int value.

We also tested the code throughout implementation to ensure everything worked smoothly. The 1-5.cm files were a great testing source in determining implementation of testing conditions as well as bringing to light various other issues in the program.

Related techniques and the design process

During the process of building the program, we once again decided to build incrementally; building one piece of the checkpoint at a time.

At each step/each addition to the codebase, we tested to ensure that all was working as per usual, using the provided test files as well as our created 1-5.cm files

We started the assignment by adding flag capabilities, and followed with working on our symbol tree parser; the goal being to print out a correct symbol tree according to specification.

Once this was working we worked on the type-check error handling aforementioned, using the 1-5.cm created files to test these errors.

Lessons gained in the implementation process

The idea of building a project incrementally is truly coming to fruition with this assignment both with the previous checkpoint and this one. It has made the coding process all the more efficient.

When going through the project actually sitting down (in-person and virtually) really helped solidify the game plan and where we were going with this checkpoint.

Consistently checking at each step has most certainly been the biggest lesson this time around as it helped catch so many small errors early on that by the time we were finished we noted that there weren't any major debugging issues we had to tackle.

Assumptions, limitations, and improvements

The assumptions imposed overall on checkpoint two came from the project specifications/grading scheme.

In terms of error handling there were no real assumptions other than those which were specified.

There were nol prominent limitations with the code, most of the building process was smooth in terms of access to data and functionality.

For further improvements going forward we would like to start by treating array variables that have been subscripted as int instead of arrays. Then we would like to tackle checking function call parameter types against the types specified in the function declaration.

Team member contributions

Overall the assignment was performed either in person or via discord peer-programming. The work performed was done together and so it’s safe to say that the contributions came quite even for the second checkpoint.