Djesse Jackson
2712207
Dejackso

Project 2 Report

The process of writing, testing, and debugging this code was considerably long. My method was basically to look at the C++ version of the code and copy over each function in python syntax as best as I could. I found as I was writing the code that there were many differences between C++ and Python that made writing the code challenging.

One of the biggest challenges that I faced was navigating the differences between switch-case and match-case. Historically, whenever I needed the functionality of switch-case in python code, I would use if-elif-else statements in its place. However, as I studied the syntax of Python to write this program, I learned that newer versions of Python had capabilities similar to switch-case known as match-case. When using it in the lookup function, I was able to get the program to work with no issues. However, I found that I had an extremely difficult time using it in the lex function. I found that in the lex function, I expected the program to check whether the value of the character class matched the value stored in the variable for said character class. I found that it instead would take the value in the character class and store it in the name of the variable that I was using. This caused an error that made it such that my first case would always match regardless of the condition, and the other cases would never be evaluated. I also attempted to encase each variable in the int() function, and that made it so that the program would check whether or not the character class was an integer, which all of them were, and proceed with evaluating the first case. This resulted in all of my characters being treated like letters when that was not the case. I eventually abandoned the match-case structure entirely in lex in favor of the if-elif-else structure and this allowed my program to function as expected.

I did a lot of my testing as I was building out the code, expecting failure along the way in hopes of achieving success when I finally finished the code. One of the more interesting things that happened was the bug where I would get trapped in an infinite loop of the program saying that the lexeme was too long. Similar to the C++ code, I gave the lexeme a 100 character limit. Figuring out what was causing that error required me to put a lot of print statements throughout the code so that I could attempt to figure out what was causing the error in question. Ultimately, I realized that this was a result of me assuming that the end of file was indicated by a -1 like it is in C++, when it is actually represented by empty quotation marks “”. Upon realizing this, I was able to get out of the infinite loop of that particular error.

I also struggled with the fact that Python does not have any pointers for you to use. So I had to get creative when it came to figuring out how to return the value in the getVarValue function. This caused a bug where the values were not being stored correctly, so I would end up printing the incorrect numbers because they were working off of different values. While I did not give myself enough time to test every single scenario that the program was able to handle, I did create

an empyt .tiny file to ensure that the “empty .tiny file” error popped up as expected. I also ensured that the results in the provided sample.tiny file printed the expected results. I made another tiny file that only had one of the statements in the file so that I could test the code on a shorter file.

Overall, this was a very interesting project. I feel like it gave me a better understanding about the differences between C++ and Python, and that it gave me some useful experience in translating code from one language to another and debugging the program along the way.