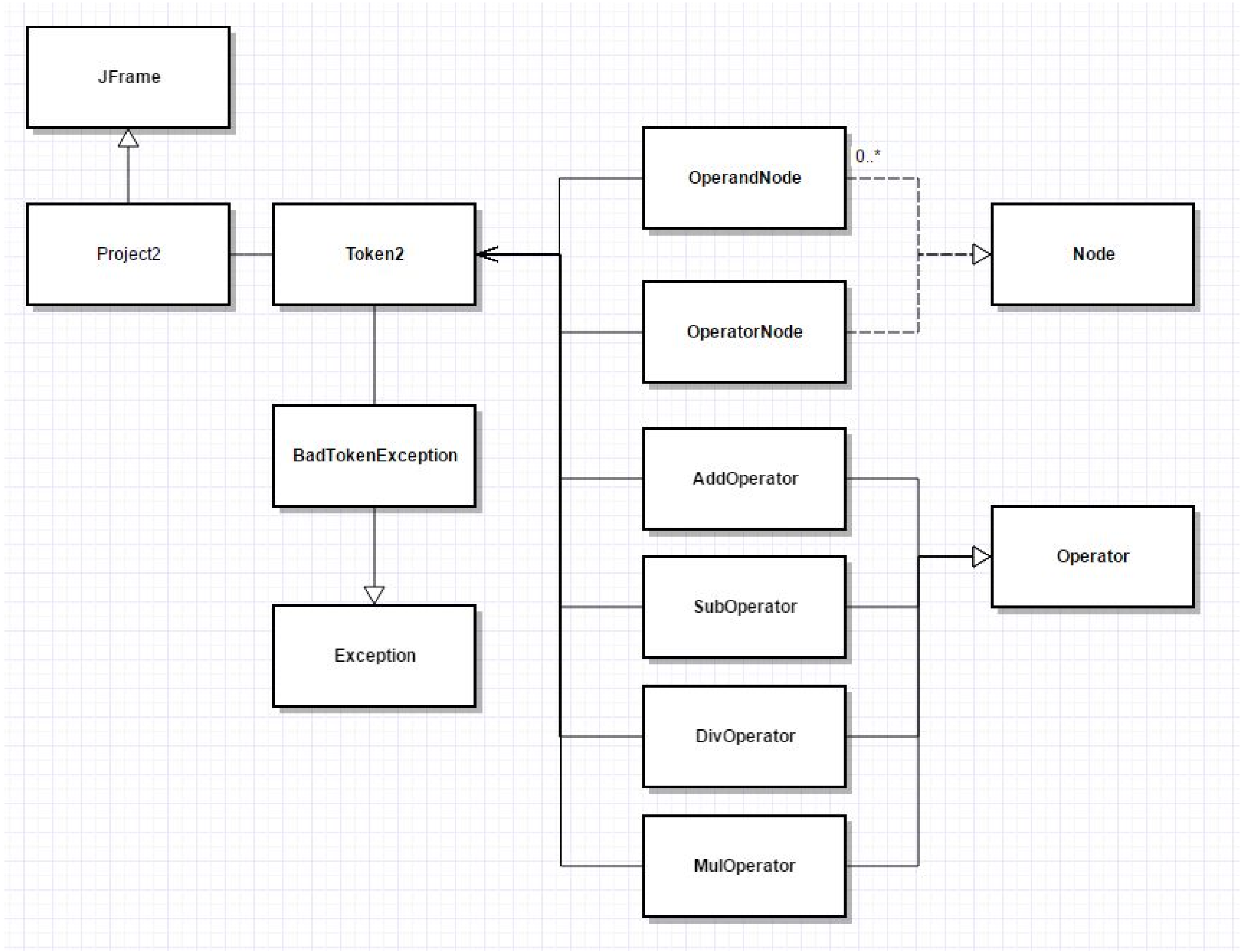
David Robbins
CMSC 350

April 8, 2017
Project 2

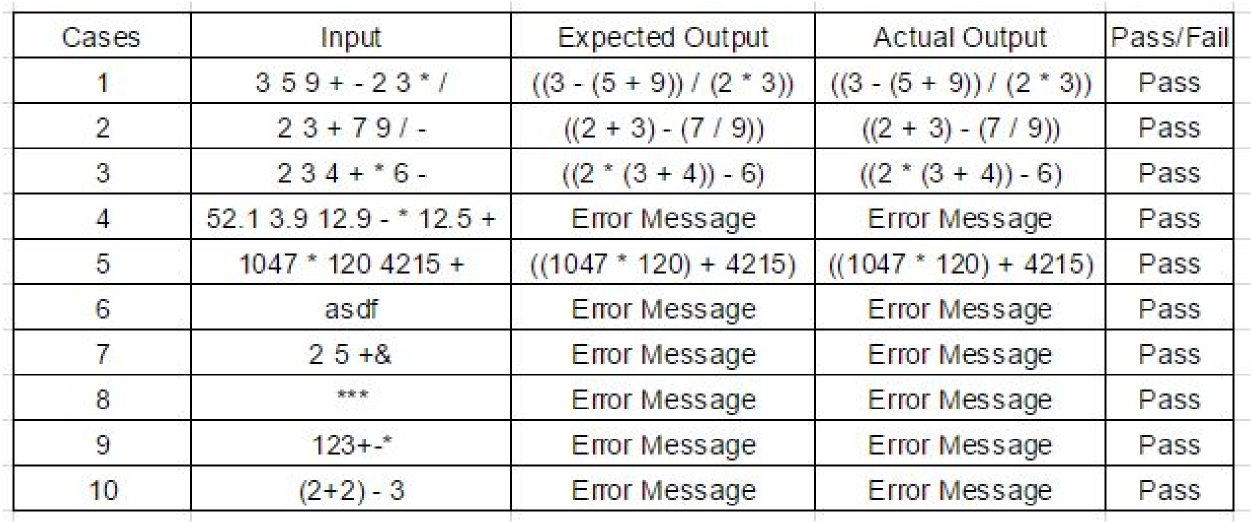
Project 2

A.



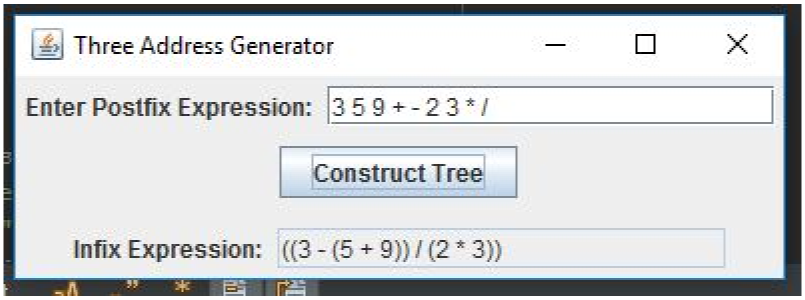
B.

Test Table



Cases	Input	Expected Output	Actual Output	Pass/Fail
1	3 5 9 + - 2 3 * /	((3 - (5 + 9)) / (2 * 3))	((3 - (5 + 9)) / (2 * 3))	Pass
2	2 3 + 7 9 / -	((2 + 3) - (7 / 9))	((2 + 3) - (7 / 9))	Pass
3	2 3 4 + * 6 -	((2 * (3 + 4)) - 6)	((2 * (3 + 4)) - 6)	Pass
4	52.1 3.9 12.9 - * 12.5 +	Error Message	Error Message	Pass
5	1047 * 120 4215 +	((1047 * 120) + 4215)	((1047 * 120) + 4215)	Pass
6	asdf	Error Message	Error Message	Pass
7	2 5 +&	Error Message	Error Message	Pass
8	***	Error Message	Error Message	Pass
9	123+-*	Error Message	Error Message	Pass
10	(2+2) - 3	Error Message	Error Message	Pass

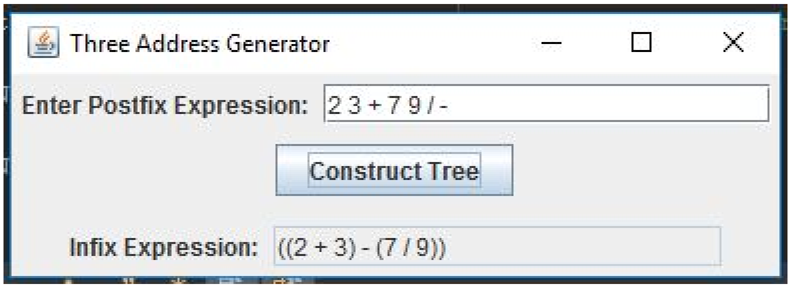
Case 1 - testing regular post fix expression



Enter Postfix Expression: 3 5 9 + - 2 3 * /

Infix Expression: ((3 - (5 + 9)) / (2 * 3))

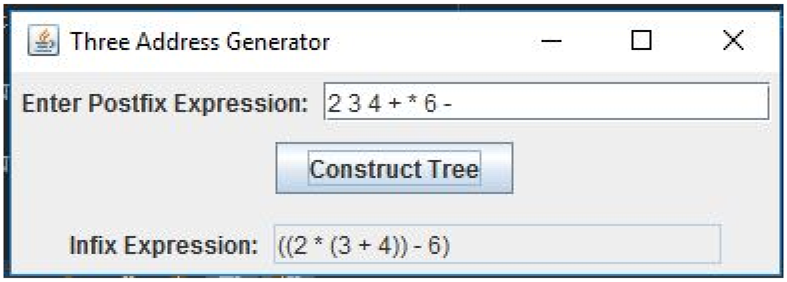
Case 2 - testing regular post fix expression



Enter Postfix Expression: 2 3 + 7 9 / -

Infix Expression: ((2 + 3) - (7 / 9))

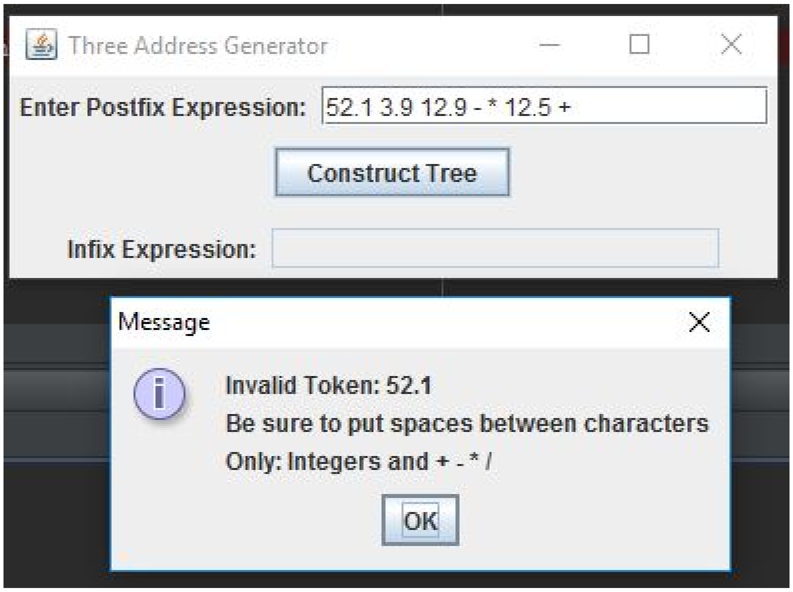
Case 3 - testing regular post fix expression



Enter Postfix Expression: 2 3 4 + * 6 -

Infix Expression: ((2 * (3 + 4)) - 6)

Case 4 - testing if non integer numbers would cause error

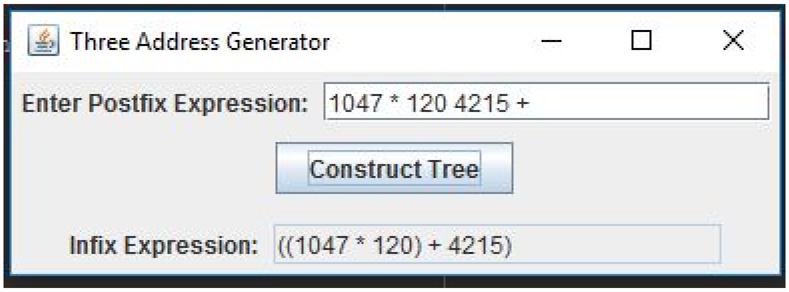


Enter Postfix Expression: 52.1 3.9 12.9 - * 12.5 +

Infix Expression:

Invalid Token: 52.1
Be sure to put spaces between characters
Only: Integers and + - * /

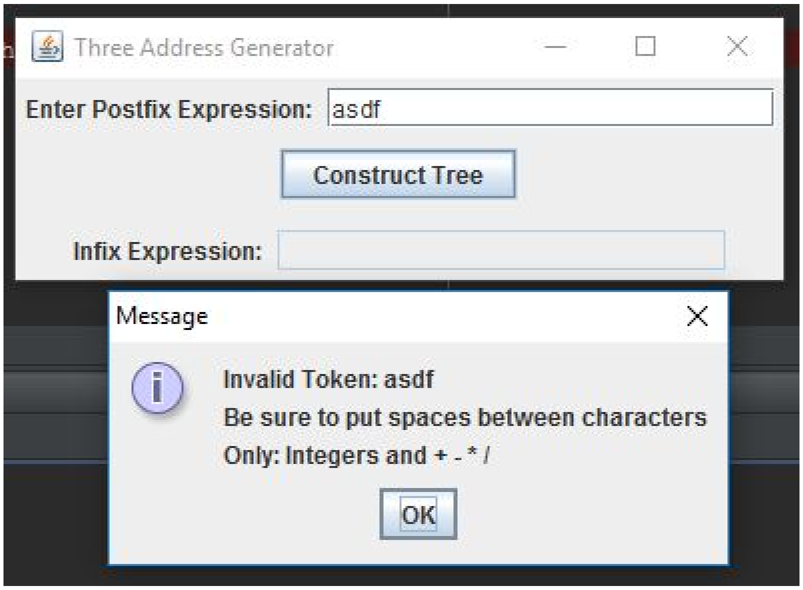
Case 5 - testing if larger numbers would cause errors



Enter Postfix Expression: 1047 * 120 4215 +

Infix Expression: ((1047 * 120) + 4215)

Case 6 - testing for wrong type of input errors

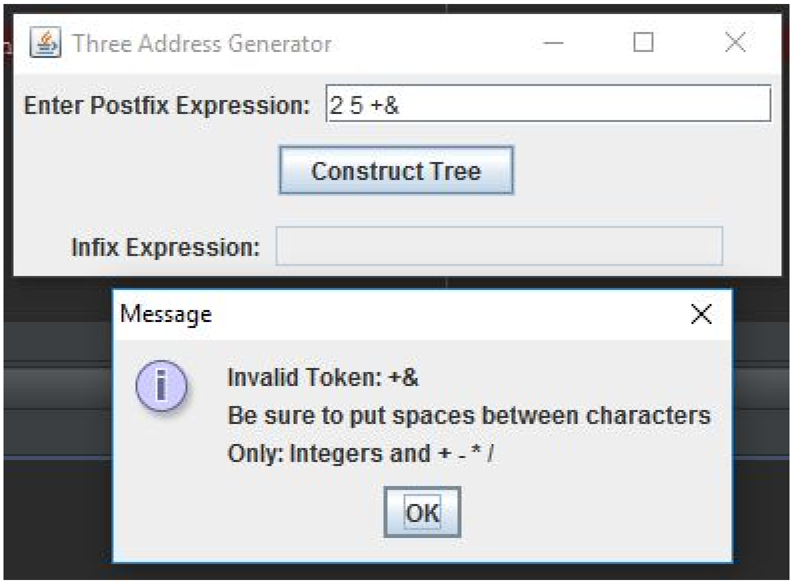


Enter Postfix Expression: asdf

Infix Expression:

Invalid Token: asdf
Be sure to put spaces between characters
Only: Integers and + - * /

Case 7 - testing for errors with a single bad token placed in last position

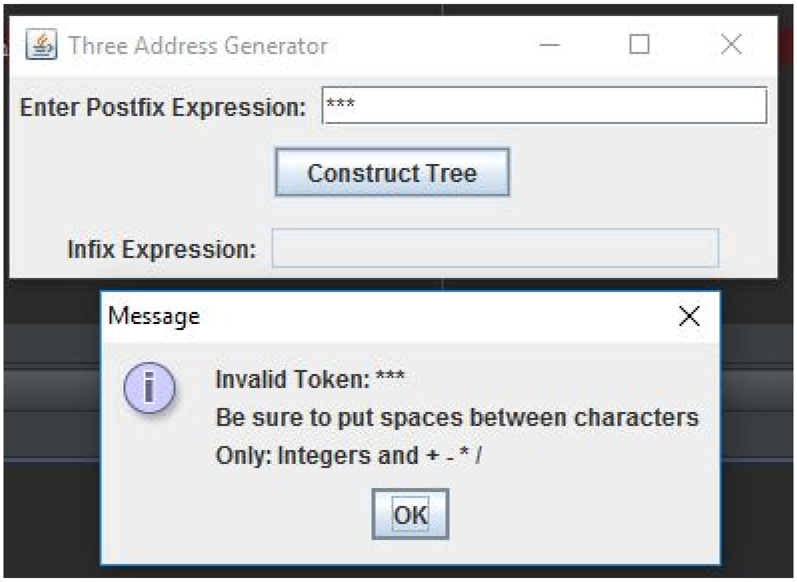


Enter Postfix Expression: 2 5 +&

Infix Expression:

Invalid Token: +&
Be sure to put spaces between characters
Only: Integers and + - * /

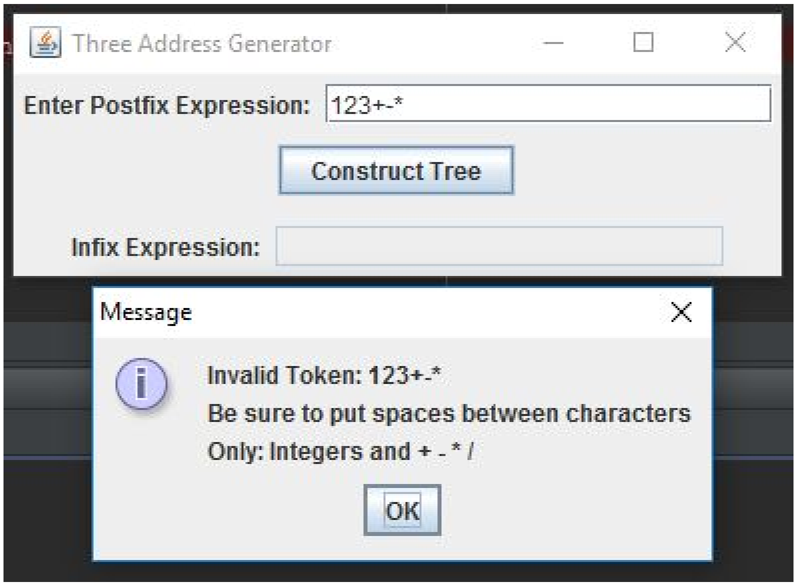
Case 8 - testing for errors with no numbers



Enter Postfix Expression: ***

Invalid Token: ***
Be sure to put spaces between characters
Only: Integers and + - * /

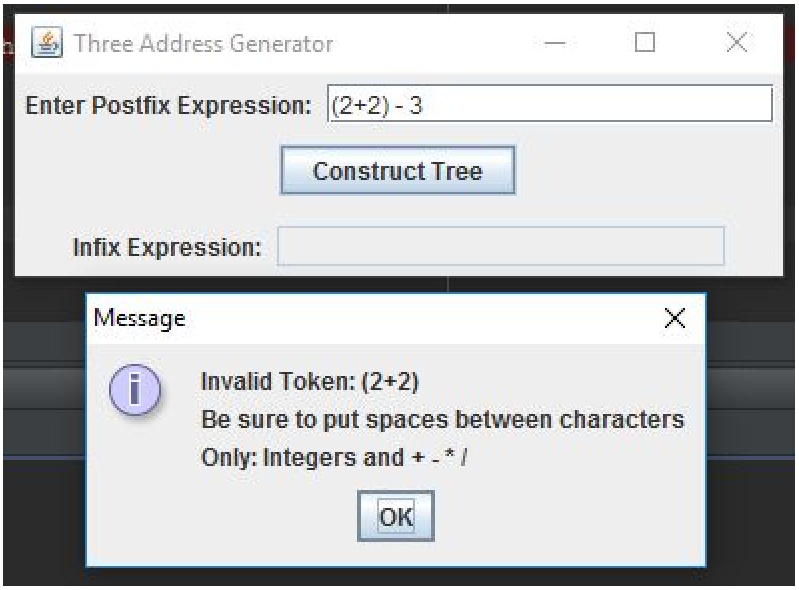
Case 9 - testing for errors when no spaces are included



Enter Postfix Expression: 123+-*

Invalid Token: 123+-*
Be sure to put spaces between characters
Only: Integers and + - * /

Case 10 - Testing for errors if an infix expression was entered



Enter Postfix Expression: (2+2) - 3

Infix Expression:

Invalid Token: (2+2)
Be sure to put spaces between characters
Only: Integers and + - * /

C.

This program was very challenging for me but I did learn a lot about how to build a tree using multiple classes so that an arithmetic function that was in postfix form could be converted to infix form. The GUI was the easy part, then the parts I struggled with were the logic in my Token2 class that was meant to build the tree, but eventually I landed on the right idea. The next part I struggled with was the building of the three assembly code, but you helped me with that and now I understand better how a parameter set in an interface class can carry through other classes and be used. I was able to review also how to print information to a text file which was good, although I couldn't figure out how to get it to display in the text file as it was showing in netbeans. In NetBeans I was seeing it correctly with a new line for each assembly instruction and a space between each completed program output. In the text file, I got a new line between each output but not a next line after each instruction, so they are lined up. The data was correct though so I let this go. Overall, I found this project to be a very good learning process which I struggled with a lot, but in that struggle came out with much more useful programming knowledge I did not previously have.