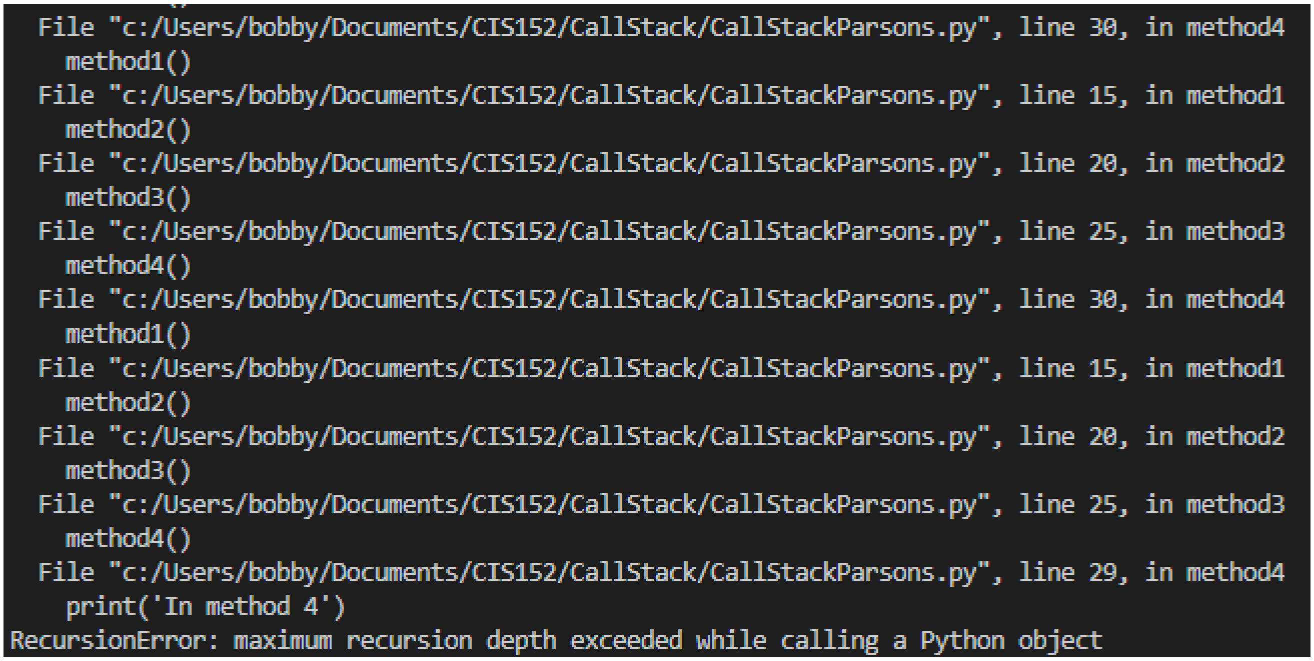
First, I set it up as defined by the assignment. As I expected, it entered main, ran all 4 functions, then exited main. Next, I set it up so that method 1 would call method 2 and so on until we reached method 4. While this functionally did the same thing of running all four methods in order, the output was dramatically different because the next function would be called before the last one exited. Finally, I completed the loop by setting method 4 to call method 1. I expected it to continuously loop forever (or until my computer ran out of memory) and never print an exit statement, but it appears Python has a limit to how how many layers of recursion you can have without exiting. I'm not sure how deep it was able to get before exiting, but it is interesting that Python explicitly stops infinite recursion before it gets out of hand.



```
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 30, in method4
    method1()
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 15, in method1
    method2()
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 20, in method2
    method3()
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 25, in method3
    method4()
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 30, in method4
    method1()
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 15, in method1
    method2()
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 20, in method2
    method3()
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 25, in method3
    method4()
  File "c:/Users/bobby/Documents/CIS152/CallStack/CallStackParsons.py", line 29, in method4
    print('In method 4')
RecursionError: maximum recursion depth exceeded while calling a Python object
```