CS 2302 – Project 3, Option B  
Professor Diego Aguirre  
Victor Huicochea – 80643271

**Project 3 – Trees**

**Introduction**

The purpose of this project is to practice the use of trees. First, the program should ask the user which type of tree to create: an AVL or a Red-Black tree. This tree should be populated with every word inside of the words.txt file. Then, a couple of functions should be created. First, the count\_anagrams function; this one receives a word and should return the number of valid anagrams, which are obtained by comparing every permutation of the word with the values inside of the tree. The second one should read a new .txt file that would contain a list of words. The function would check how many valid anagrams each word has and return the word with the most valid anagrams.

**Proposed Solution**

Five different files were written to solve this problem. Four of them consist of classes (AVL Node, AVL Tree, Red-Black Node and Red-Black Tree) and the last one is the Main program.

The Main program divides the problem into several functions. Two of them are dedicated to creating either an AVL or a Red Black tree. The third function prints every node in a tree; it was written with testing purposes. The next function generates a list with all possible permutations of a given word, which is used by two other functions. These two functions compare every permutation with the tree nodes, with the purpose of checking if it is a valid anagram; then one of the functions prints the valid anagrams and the other one adds the number of valid anagrams. The last function reads a new file with a list of words and returns the word that has the most anagrams.

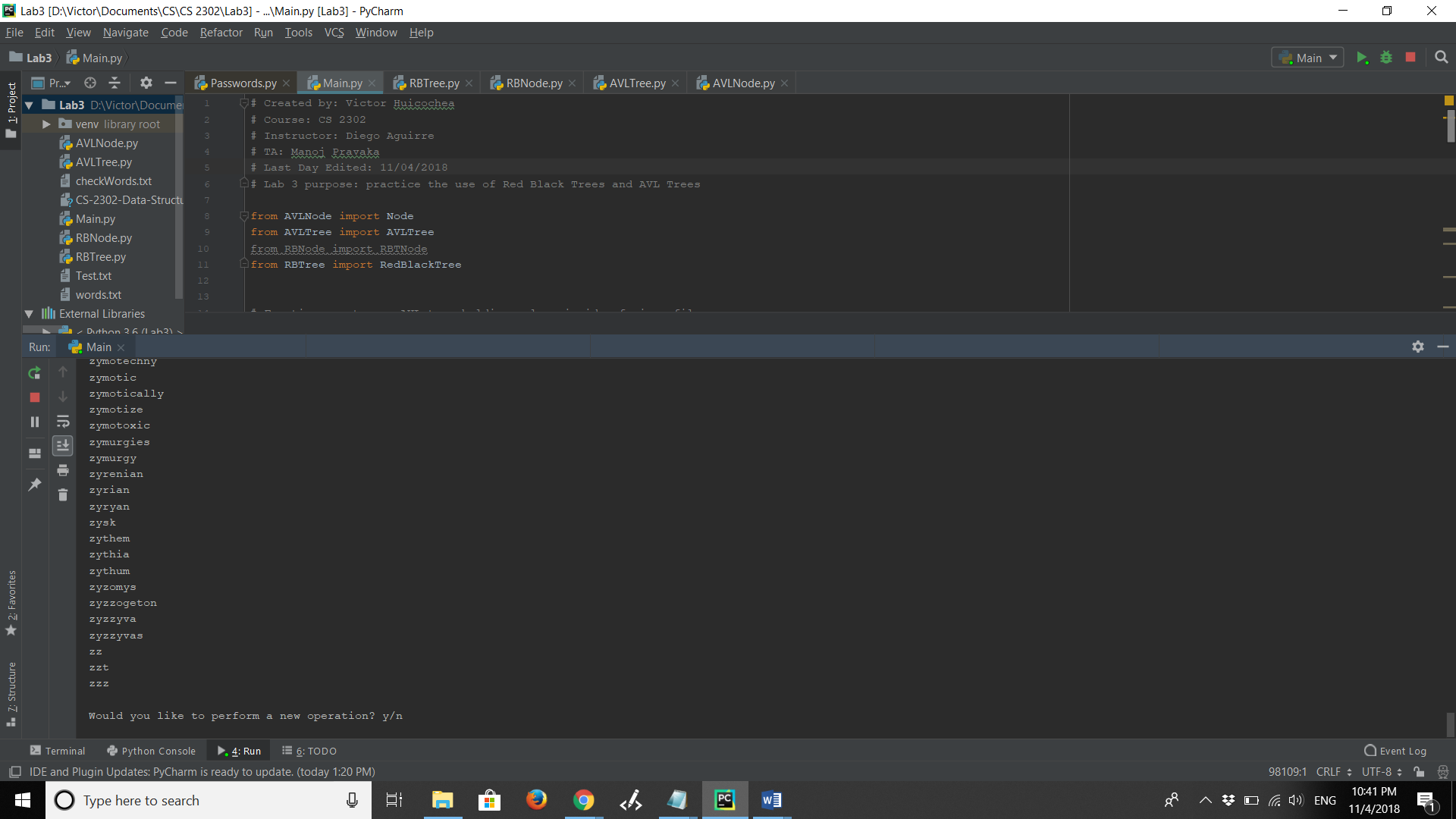
**Experimental Results**

1.- Testing with the words.txt file.

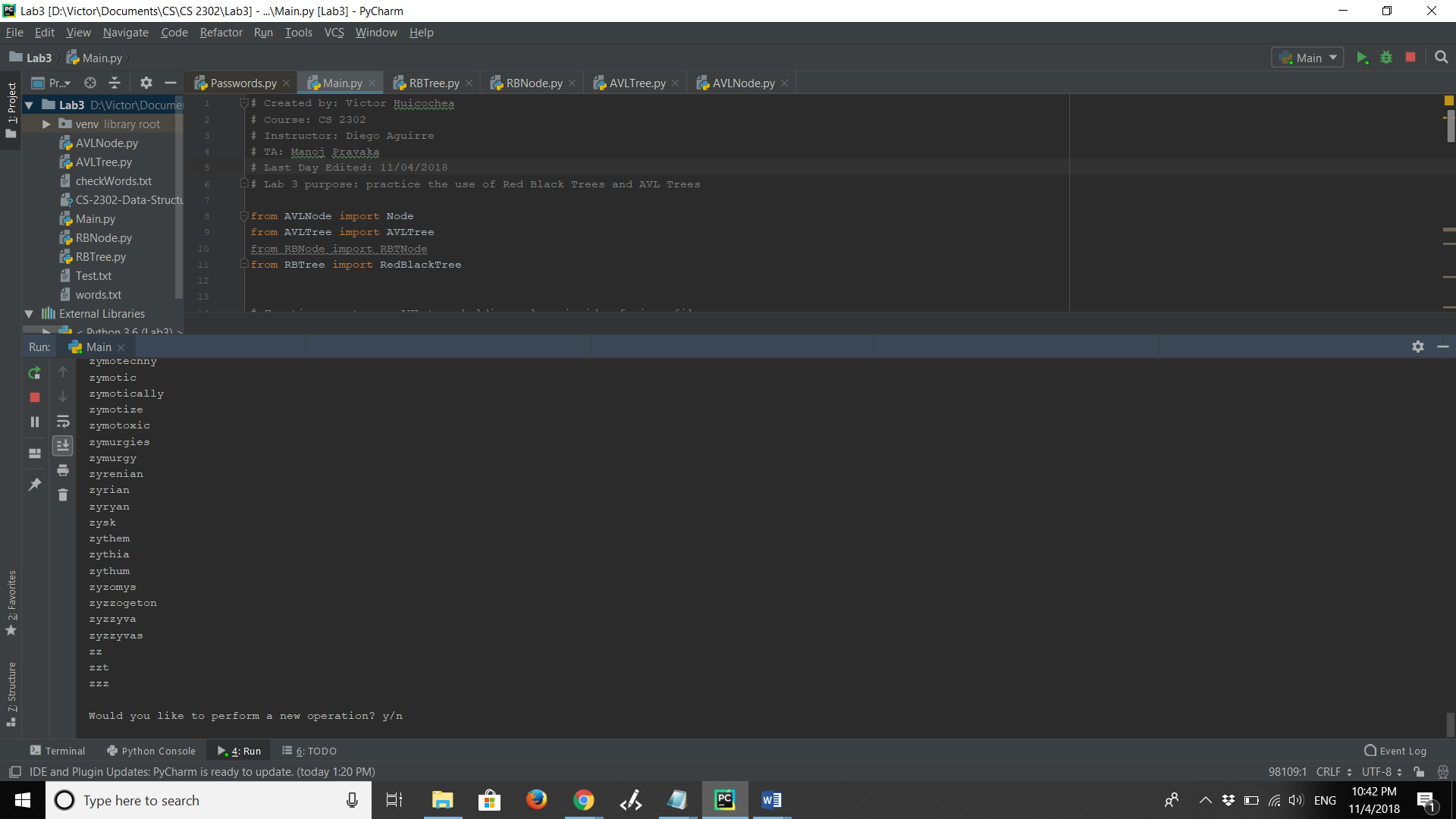
The provided file was used to create both trees.

Output after printing each tree:

AVL Tree



Red-Black Tree

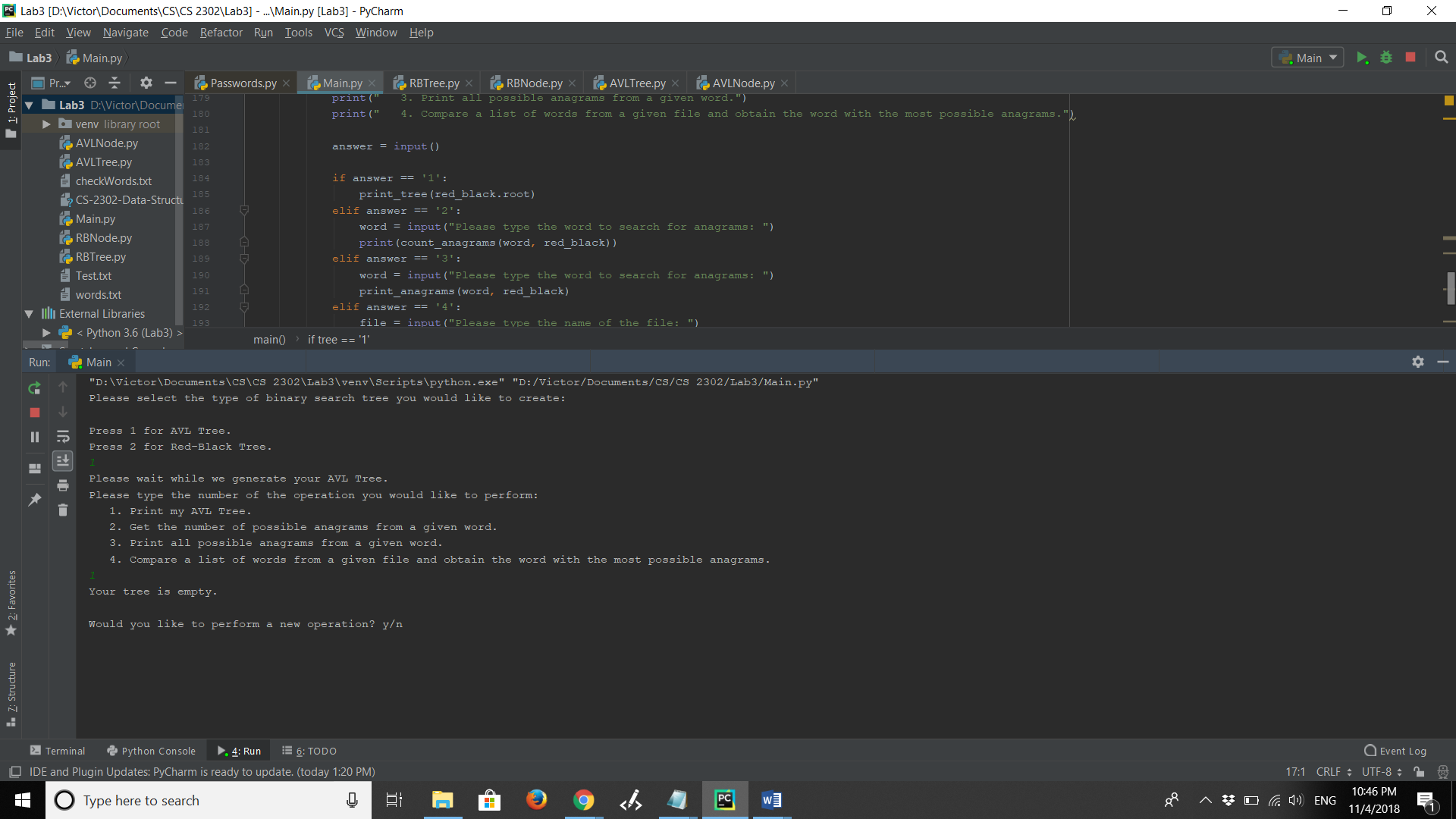


2.- Testing with an empty file.

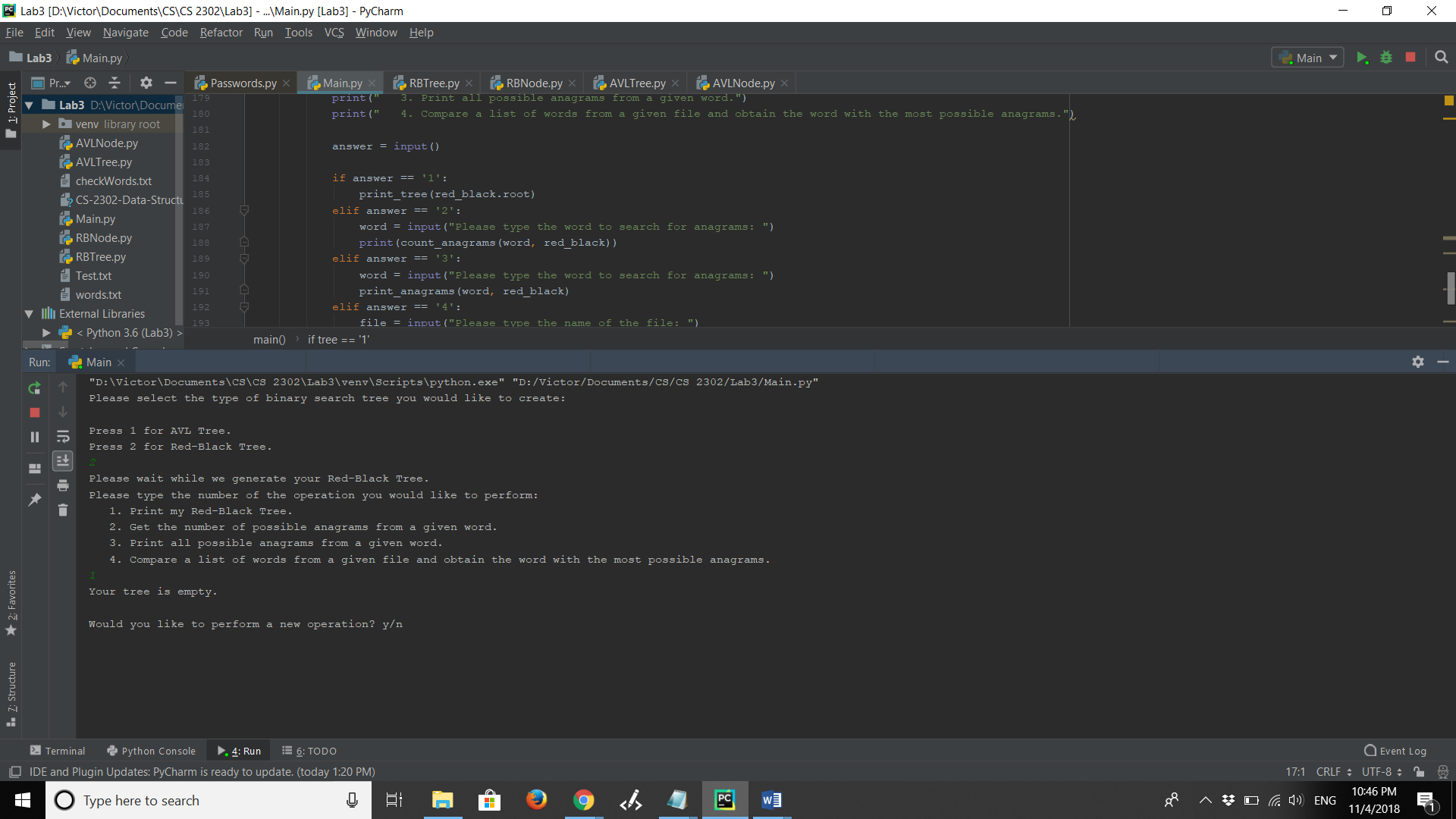
The words.txt file was deleted and replaced by an empty file.

Output after each tree:

AVL Tree



Red-Black Tree

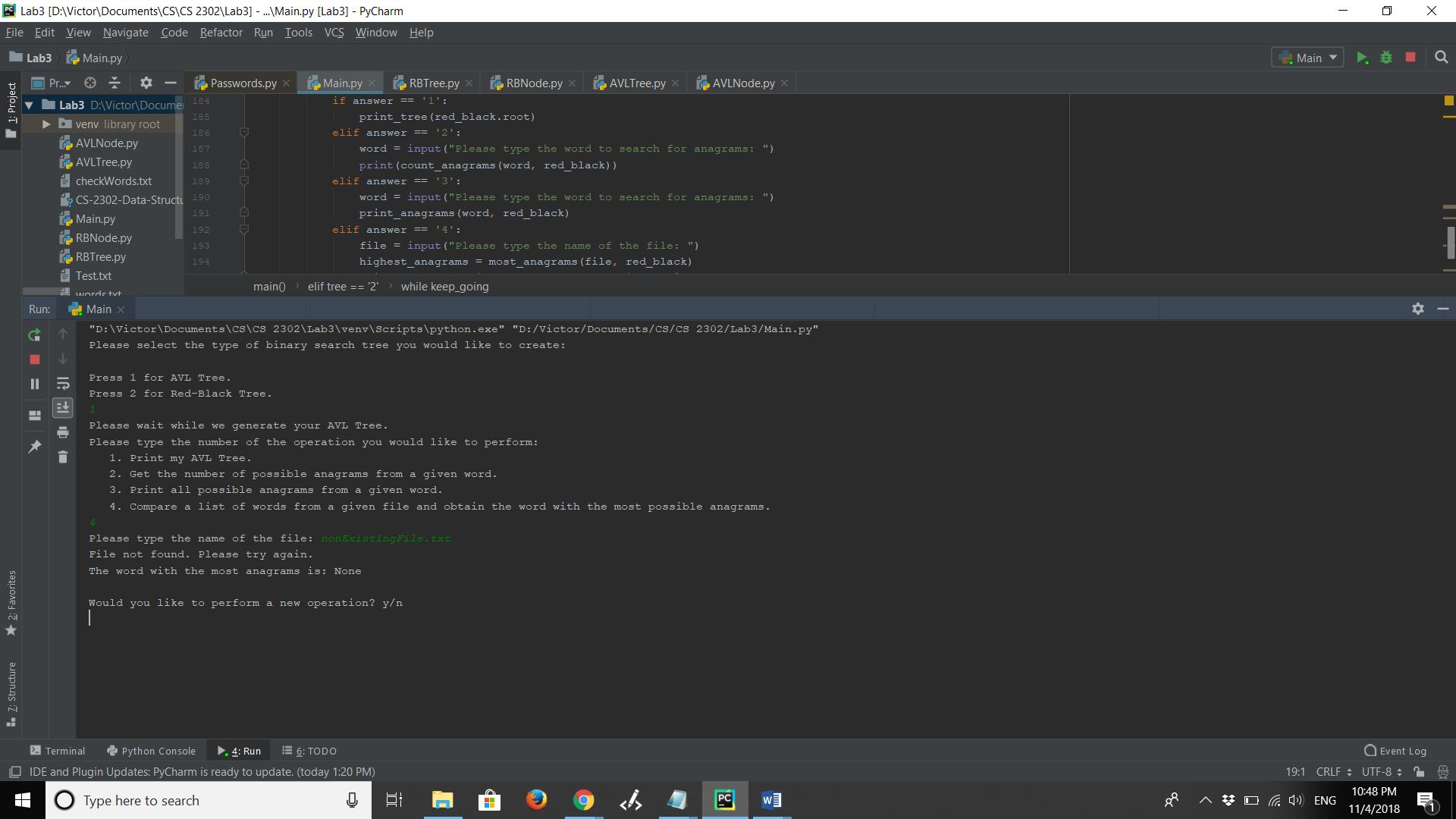


3.- Testing with a non-existing file.

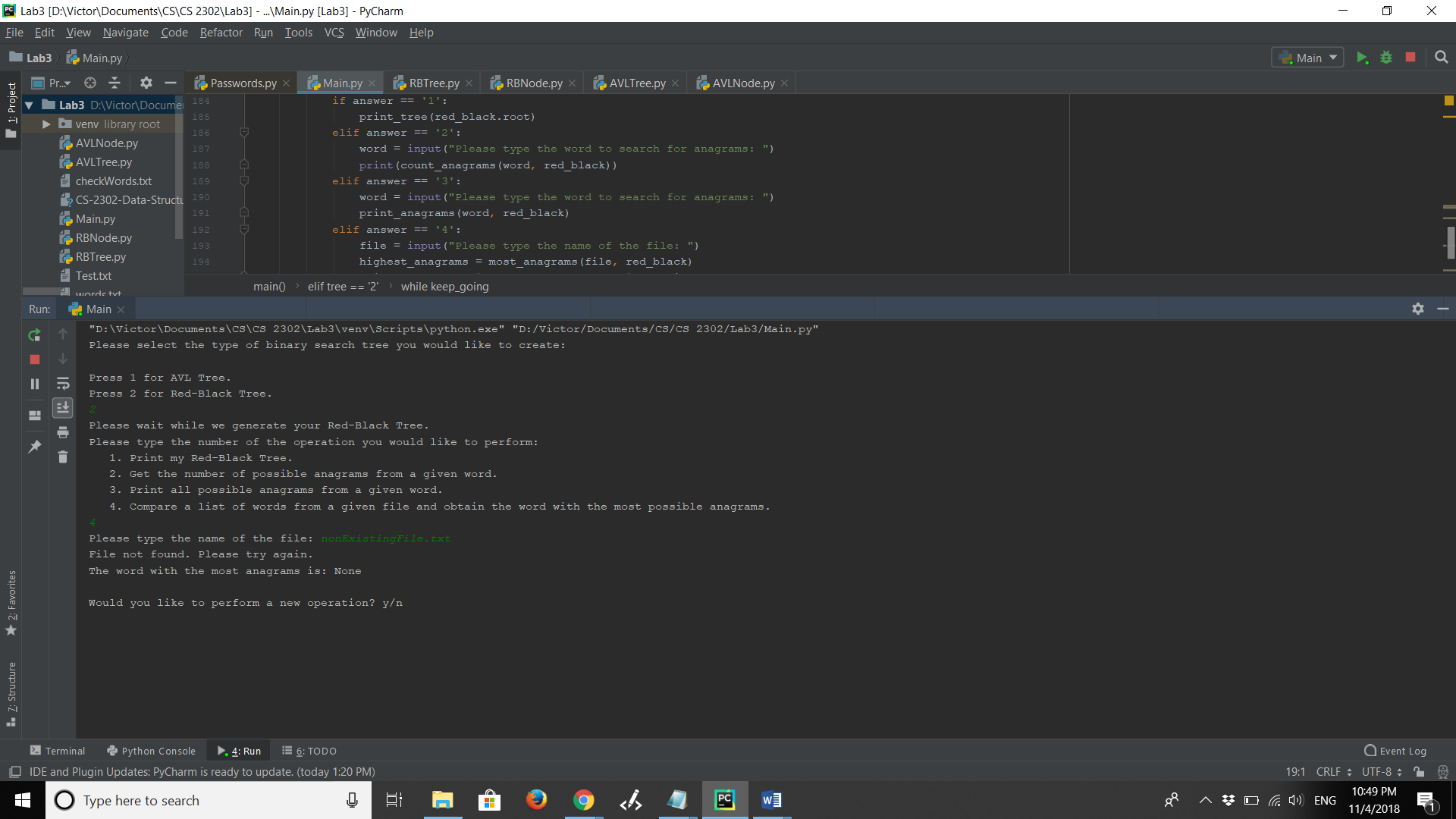
When reading the second file, the one with the words to compare and get the one with the most anagrams, a non-existing file name was given.

Output after each tree:

AVL Tree



Red-Black Tree



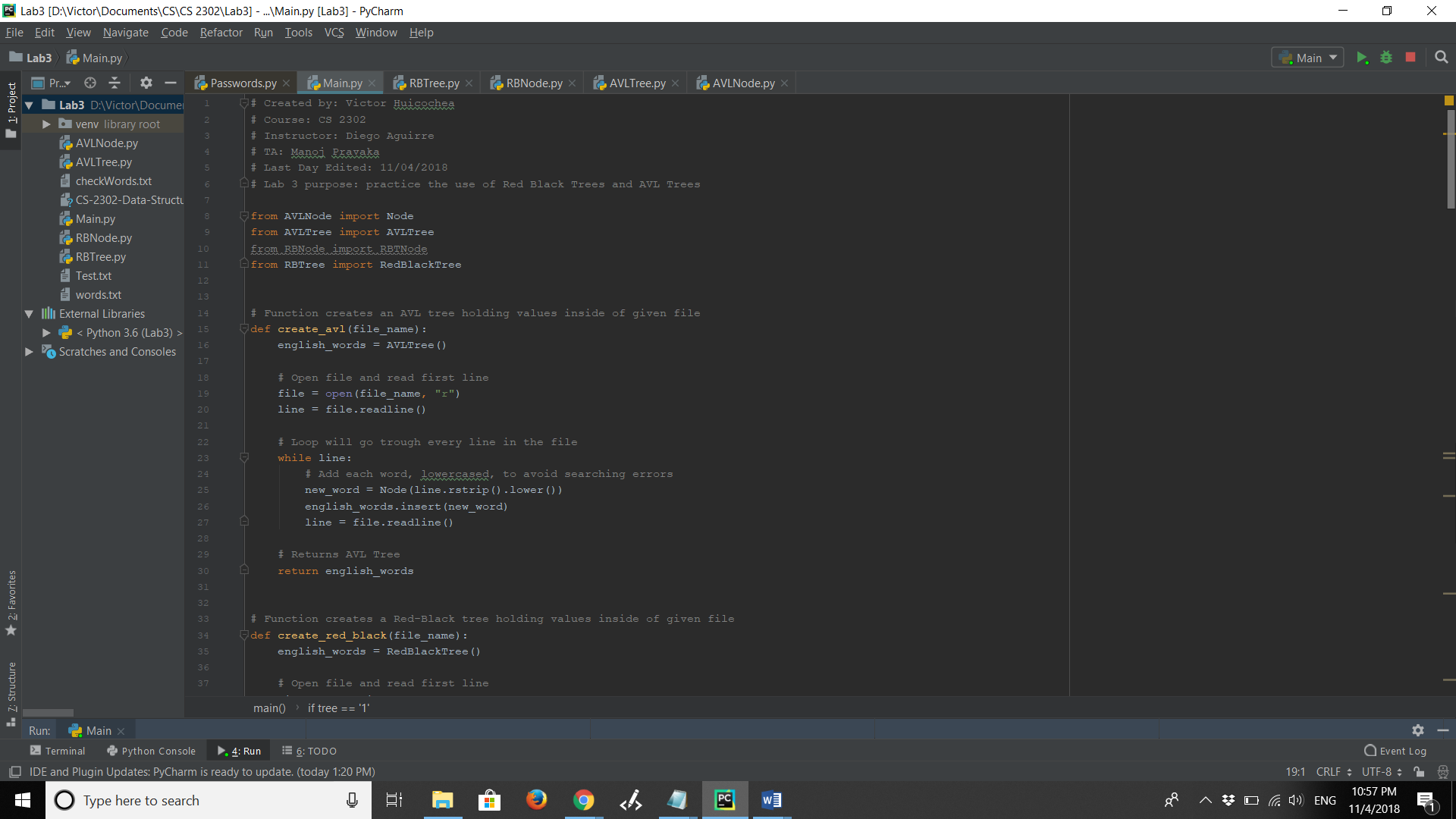
**Conclusions**

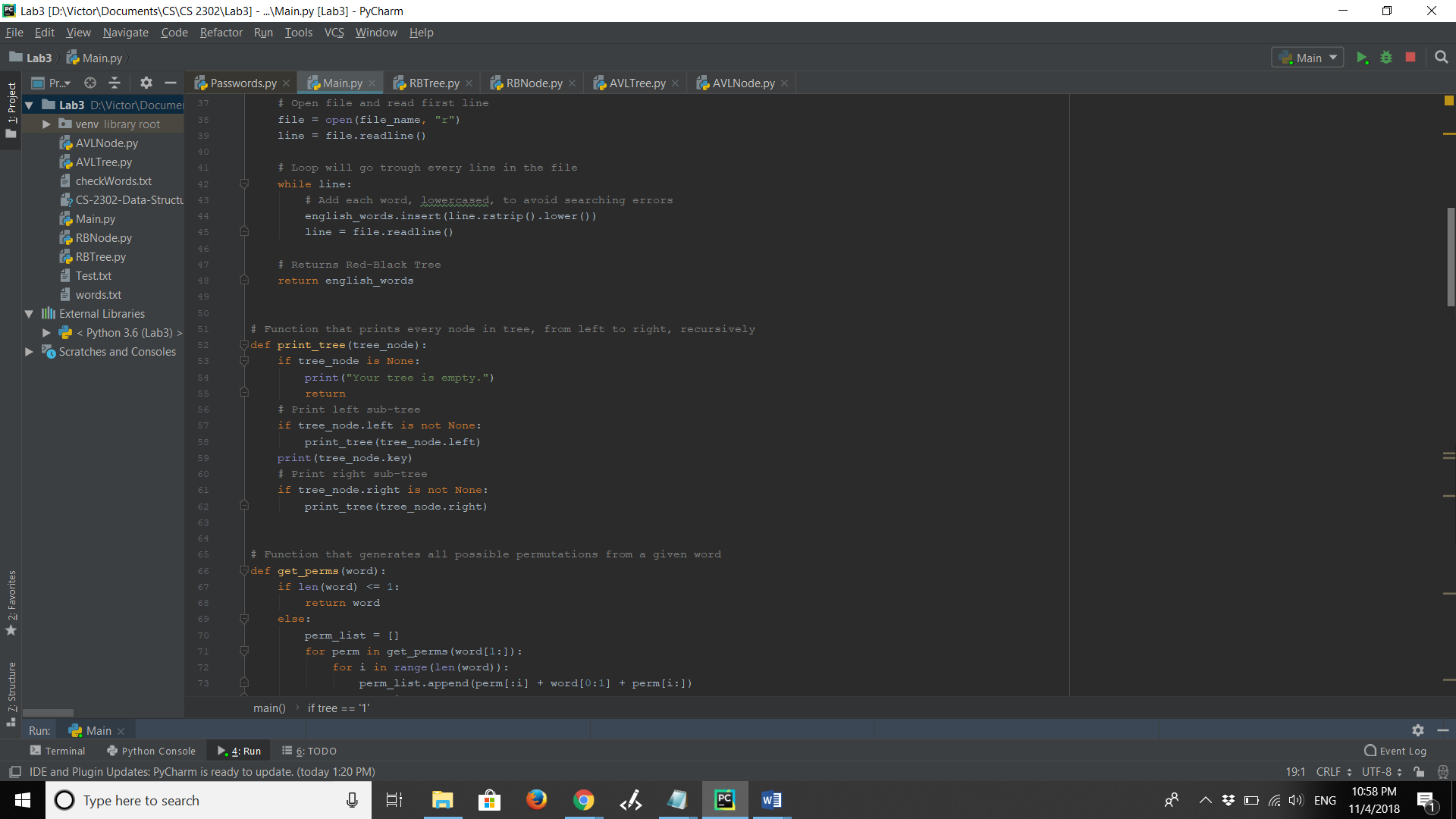
This project helped me to understand the use of two types of Binary Search Trees: the AVL and Red-Black trees. Now I feel more comfortable working with trees since I am now more capable of visualizing how the data moves after every method.

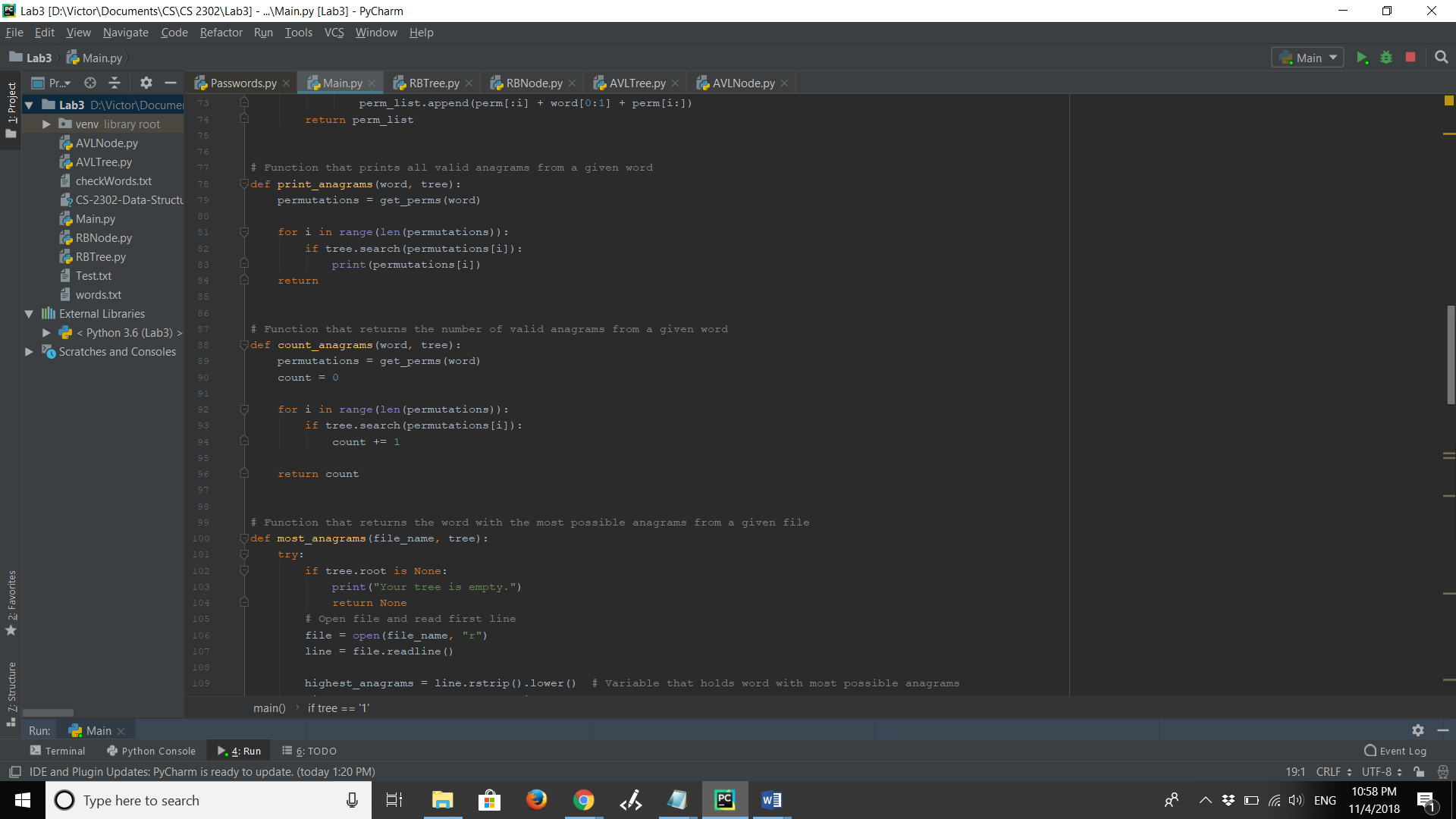
I also realized which operations are faster and slower depending on the type of tree I am using.

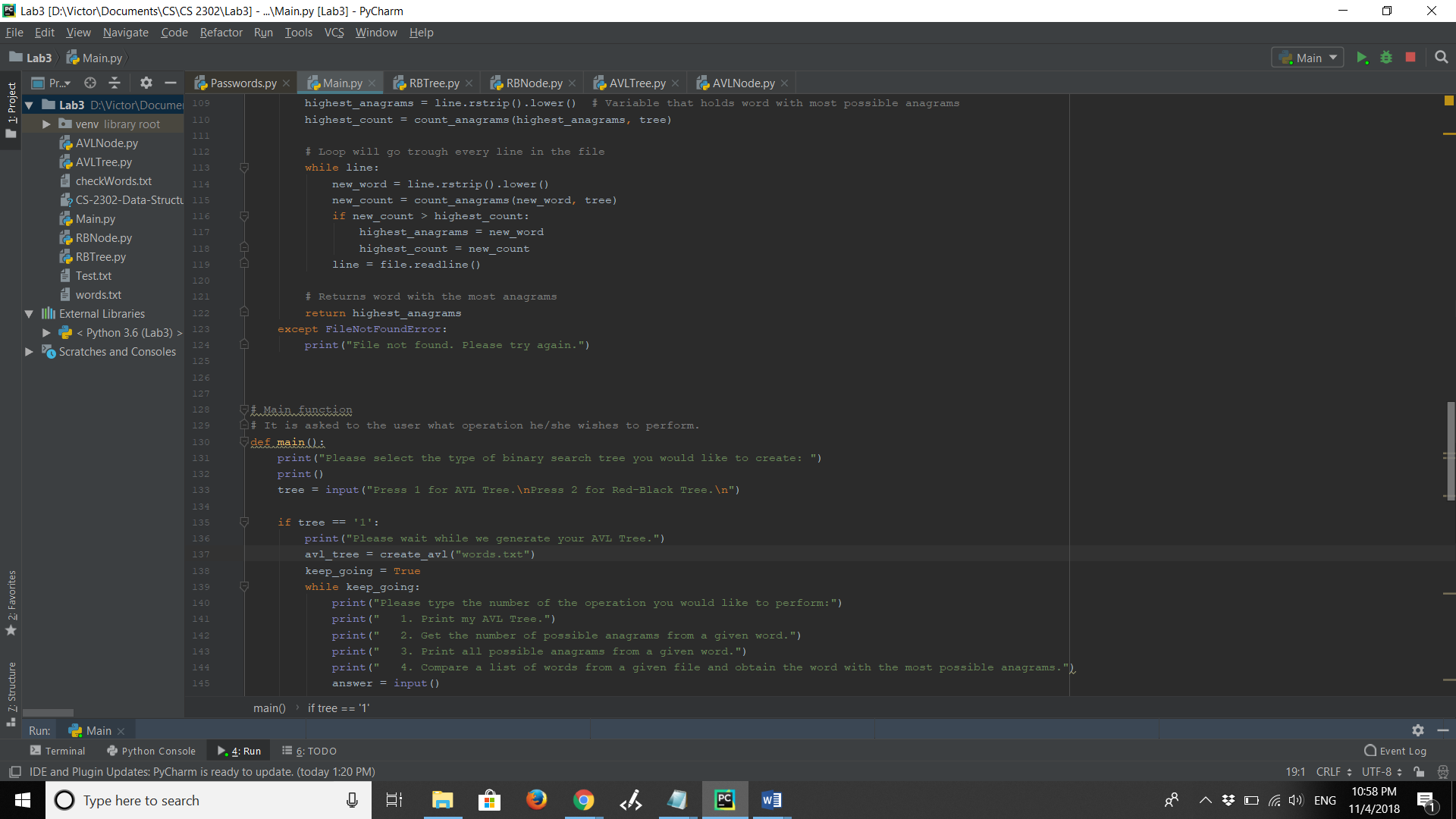
Finally, I learned some new Python syntax, since I tried new things that I had not tried before, such as using a switch statement, which apparently does not exist in Python.

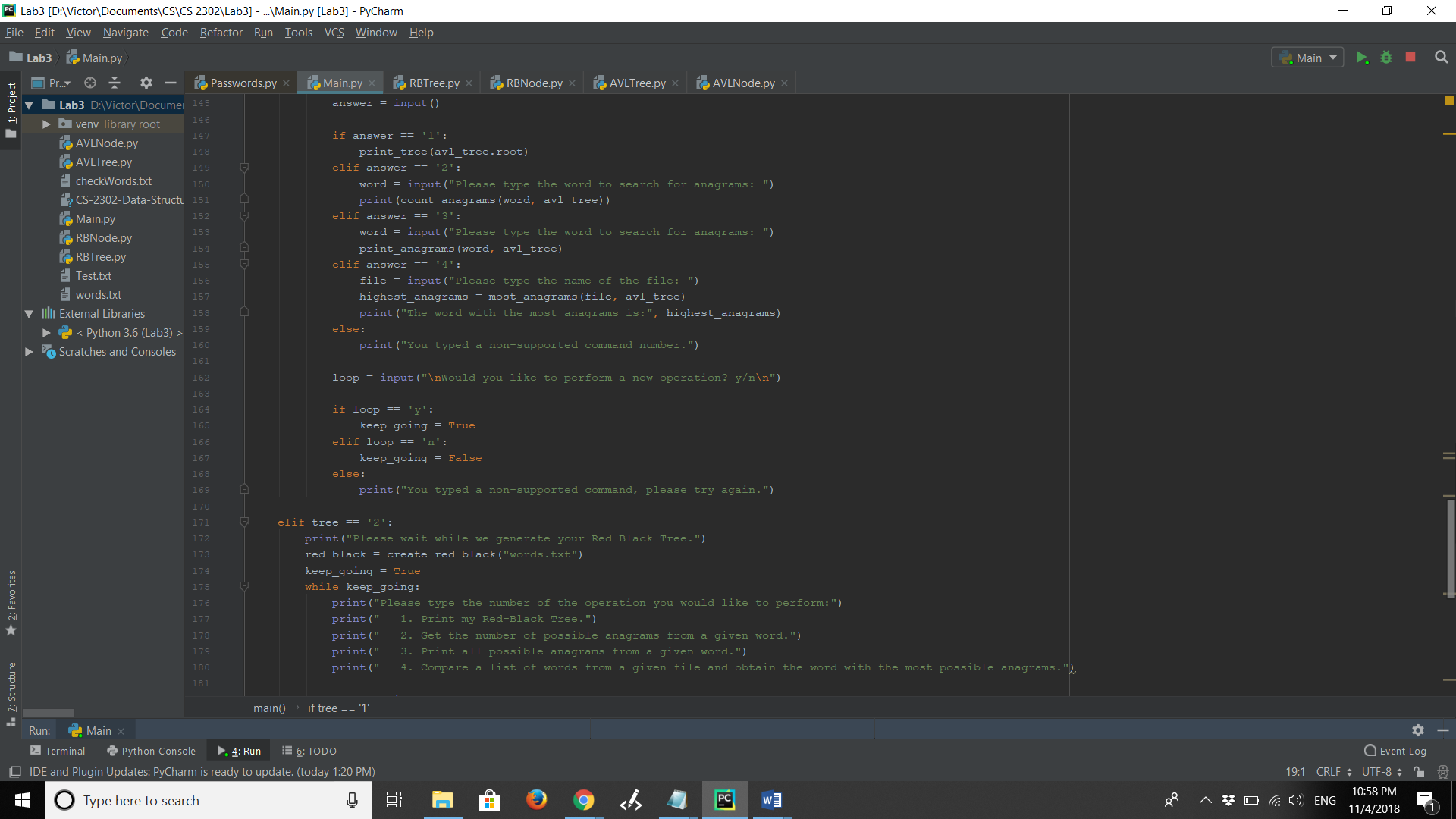
**Appendix – Source Code**

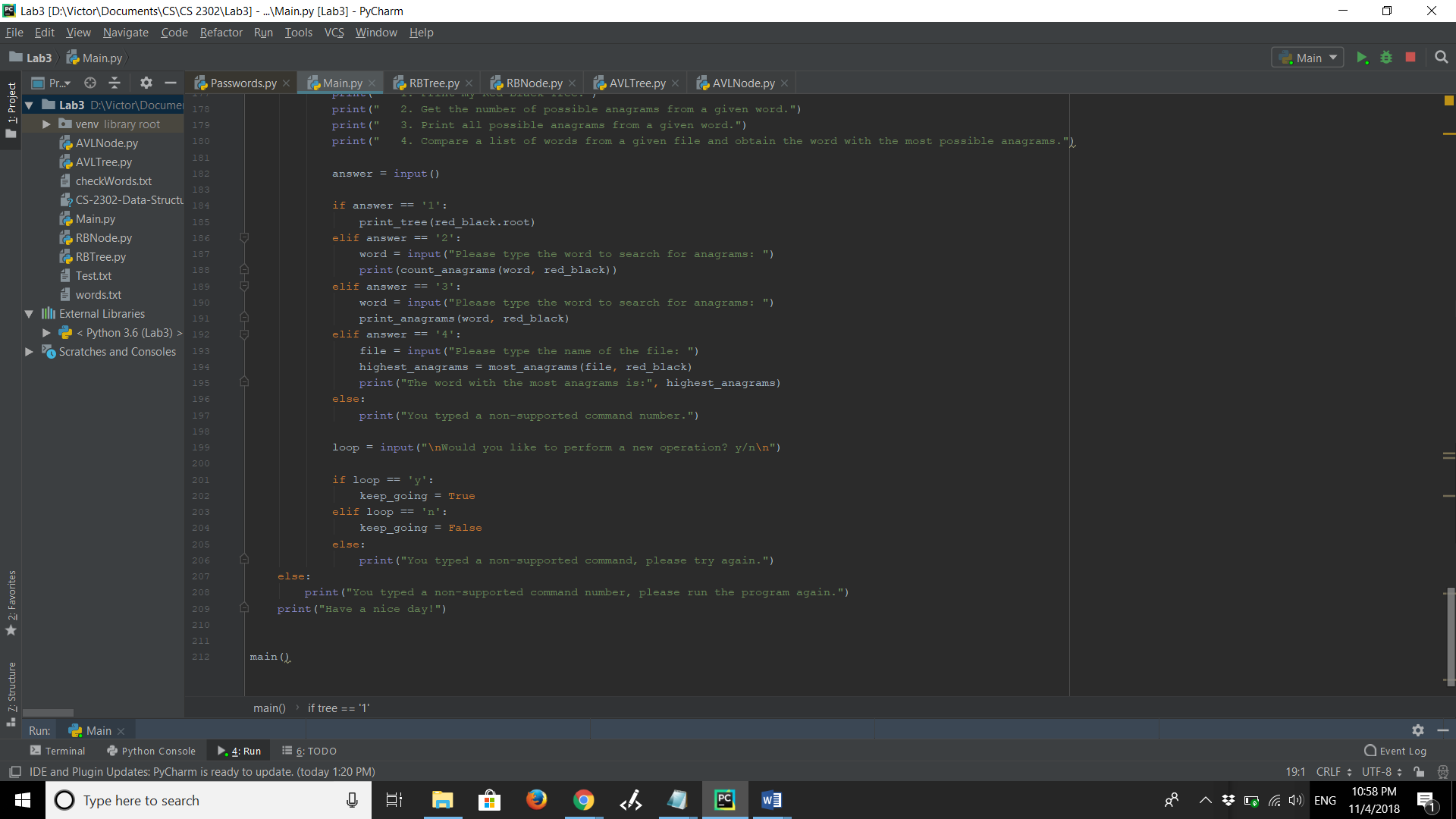












**Academic Honesty Certification**

****I certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class.