Lab 7 - Profiling and Optimization

Rahul Kukreja & Jeffrey Yim

jyim3@bcit.ca

Welcome!

In today's lab, you will:

1. Utilize cProfile to profile code that takes a while to run. You will identify which parts of the code are inefficient and change them.
2. Use comprehensions, built-in functions, data structures and other tricks to optimize code.
3. Get first hand experience working with not-so-good code. This is going to happen a lot once you work. The standard isn't very high out there.

Grading

This lab and all future labs will be marked out of 10

For full marks this week, you must:

1. (5 points) Improve the given code using profiling to make it run faster . (Part 1)
2. (5 point) Further optimize the improved code after you have reduced the run-time. (Part 2)

Requirements

Make sure you **read both parts** before starting on Part 1.

Part 1: Profiling

Clone the project: https://classroom.github.com/a/uiWnviA6

Run the code in driver.py.

The `BookAnalyzer` class is responsible for reading a text file and providing a method to extract all the **unique words** that appear

- For instance in this sentence, “hello hello dog dog dog cat”, the unique words are “hello”, “dog”, “cat”

This code is not written very well and takes a few seconds to run. This will be different for each machine. On my machine it took approximately 10 seconds.

When faced with a situation like this, profilers are a super convenient tool to help us identify what is taking so long to run. We can then edit and change only those parts of the code that contribute to the run time.

1. Duplicate this module, or copy the code into a new module called `driver_profiled.py`.
2. In this new module, you will repeat the following steps until you have improved the code as much as you can without re-designing the code and changing the data types. :
    1. Profile the code and identify which parts of the code take the longest to run/ are inefficient.
    2. Interpret the results and identify the parts of the code that are contributing to this long run time. Change only those parts so that the code runs faster.
        1. Restructure the code by **identifying functions that are called too often unnecessarily**.
        2. **Cannot re-design the code or change data structures**. Not yet anyway

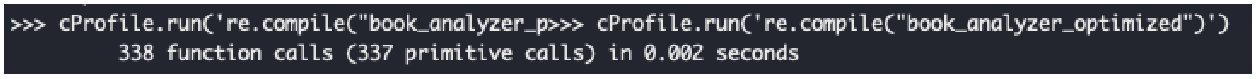
By the end of this part, My code was running at approximately 3-4 seconds. This is a lot better but still isn't the best. Remember the goal here is to improve the performance of the program, not to re-design our code. Once this done move on to Part 2!

Part 2: Optimizing and Re-designing

1. Duplicate your code from `driver_profiled.py` into a new module called `driver_optimized.py` .
2. Go through the `Lab 07 Optimization Slides` on D2L to learn how to optimize python code.
3. Optimize and re-design how the Book Analyzer class reads and processes data. Also re-write the code in `find_unique_words(self)` to run in O(N) time instead of O(N^2).
    - You can save a lot of time and energy by creating a new attribute called word_count when reading data from the file. What data structure would you use for this?).
    - Remember to re-profile and check how your modifications affect the results.
    - What built-in methods can you use to improve the code? Can generators and/or comprehensions be used? Can you reduce the number of times you iterate over the data?
4. By the end of this part your code should run in less than a second.

- Ensure you push your work to github classroom. I'd like to see sensible git commits comments, and commits must take place at logical points in development.

Just for fun, this was the fastest optimization I've seen from one of my past students



```
>>> cProfile.run('re.compile("book_analyzer_p>>> cProfile.run('re.compile("book_analyzer_optimized")')
         338 function calls (337 primitive calls) in 0.002 seconds
```

That's it. Good luck, and have fun!

FAQ

Does the output of the optimized and profiled version need to match the original driver.py?

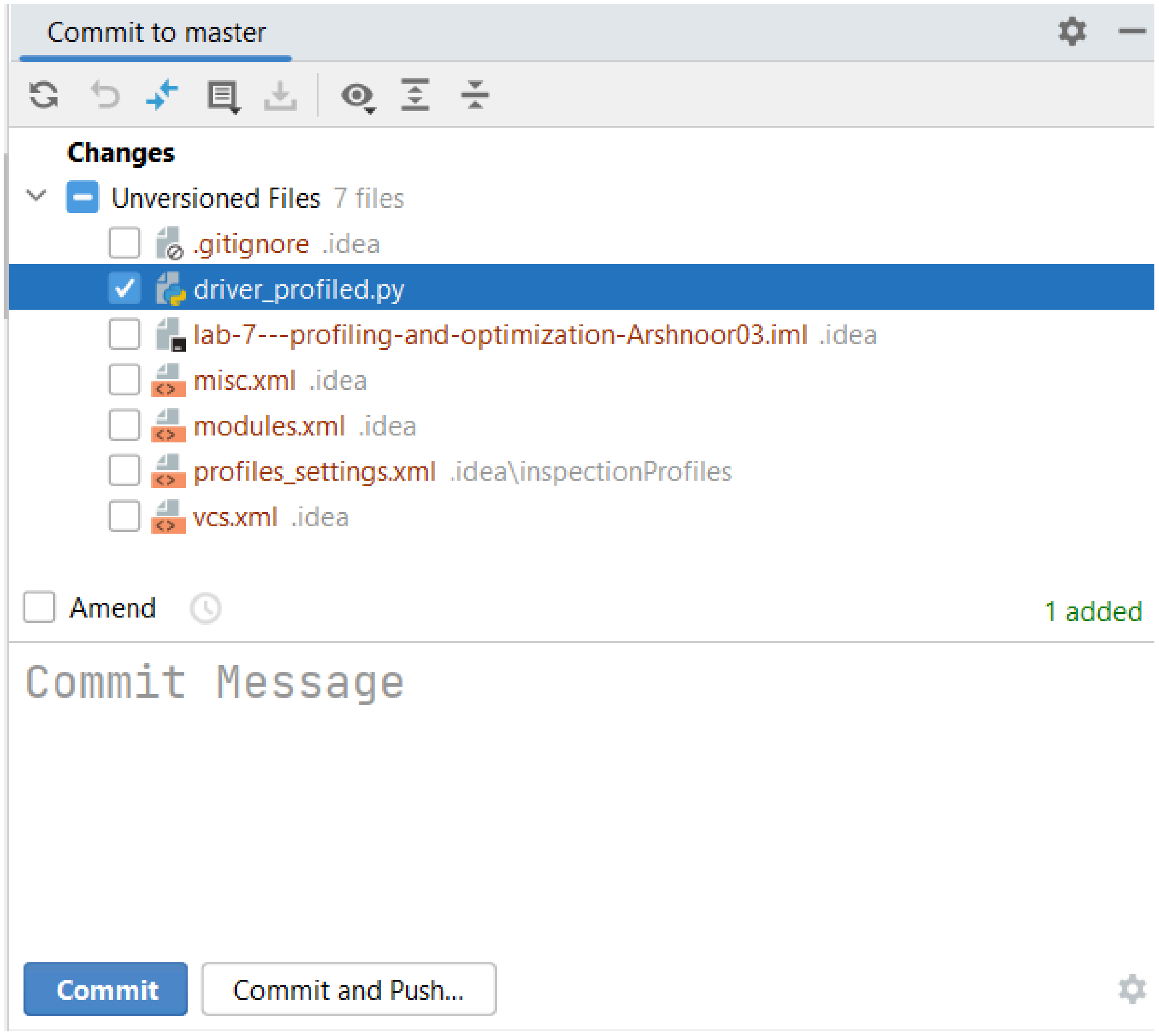
Yes, the casing of the words needs to match, but "List of unique words" count can differ by a few words

Notice the profiled/optimized output has all lowercase, this will **not be accepted**

Original driver.py	Profiled/Optimized
Usher House the of Fall	**u**sher **h**ouse the of **f**all **method lower of str objects** **method replace of str objects** **method append of list objects** **method pop of list objects** **method split of str objects** **driver.py: 55 (is_unique)**

Why aren't `driver_profiled.py` and `driver_optimized.py` appearing on github?

These newly created files are unversioned, in other words, they're not being tracked by Git. When committing your files, make sure to expand the "Unversioned files" area, and check the newly created files. Then commit and push them to GitHub.



[ ] .gitignore .idea
[✓] driver_profiled.py
[ ] lab-7---profiling-and-optimization-Arshnoor03.iml .idea
[ ] misc.xml .idea
[ ] modules.xml .idea
[ ] profiles_settings.xml .idea\inspectionProfiles
[ ] vcs.xml .idea