Large Assignment #3: Ruby
Due: Friday, 6 December 2024 by 11:59 PM

Instructions.
Implement the program described below, following all the guidelines, in a file called `la3.rb` and submit it on lectura using the following command:

```
turnin csc372la3 la3.rb
```

Implement a standard trie in Ruby. Write your solution in a file called *la3.rb.* See the slides for details on what a standard trie is, as well as suggestions for the implementation.

In addition, your file should contain an interactive program for using the trie class to search for strings in files. The program must meet the following requirements.

- The program should start by asking the user for a file name. When that filename is given, the program should open it and build a trie from that file.
- Then the user should be able to input strings to search for (this should be on a loop that only ends if the user types :q).
    - Typing `:q` should quit the program.
    - If the string that is entered is not in the text (and therefore not in the trie), print a statement saying that the string is not found.
    - If the string is a full word in the text, the program should print out all locations at which the string occurs. The location should indicate line number and word number.
    - If the string is a prefix for any words in the text, the program should print out all the strings that occur in the text that start with that prefix. (I recommend doing a DFS of that subtree.)
    - If the input string is both a full word and a prefix, you just need to print the locations of the full word.
- Please just print this information to standard output using `puts` or `print.` There is no need to write it to a file.

The following shows an example using a simple text file.

simple.txt:

```
sue sells
seashells
on the seashore
at the seashore
by the sea
```

```
> ruby Trie.rb
Type the name of your file.
> simple.txt
Type a string to search for.
> susan
Not found.
> sue
(0, 0)
> sea
(4, 2)
> seas
seashells
seashore
> seashore
(2, 2)
(3, 2)
> :q
```

A couple more things…

- You do not have to worry about punctuation. It is fine if your trie treats “`seashore,`” , “`seashore.`” , and “`seashore`” as three different words.

- The actual messages that are output in the program and the format of how you print out locations is up to you as long as it is clear, but make sure `:q` is the command to quit.
- I have provided several files that you can use to test your code.
- To grade this, the TAs will run your program and provide various inputs. They will also look at your code to make sure you are implementing the structure correctly.