Algorithm Design

LAB 06 : ADVANCED SEARCH PROGRAM

Due Saturday at 5:00 PM MST

This week, we will implement in Python a design we created previously.

Program Description

This program will function exactly the same as Ponder 02. It prompt the user for a filename. We will then read the contents of the file into a list. The program will then prompt the user for a name. Finally, we will tell the user whether the name is in the list.

Read From a File

The first part of the program will be to read the contents of a file into a list. This data will be in JSON. For example, the contents of the file `languages.json` might be:

```
{
  "array": [
      "C",
      "C#",
      "C++",
      "Java",
      "JavaScript",
      "Kotlin",
      "PHP",
      "Perl",
      "Python",
      "Swift",
      "VB"
  ]
}
```

Note that the contents of this file are guaranteed to be in sorted order. Of course, if the file does not exist, then a user-friendly message will be displayed to the user.

Advanced Search

Next the program will use the advanced search to determine if the name is in the list. This should behave the same as it did for the linear search in Ponder 02. The only difference is that we will use a much more efficient algorithm to perform the search. In other words, the functionality is the same but the algorithm used to produce the functionality will be much more efficient.

When finished, the program will display a message indicating whether the name was found.

Note, please do not just use Python's built-in `in()` function to see if a word is in the list. The whole point of the assignment is to implement this yourself.

Example

The following example is a single run-through of the program.

```
What is the name of the file? languages.json
What name are we looking for? C++
We found C++ in languages.json.
```

Assignment

To submit this assignment, two things are needed: your source code and a demonstration video.

Source Code

Submit your source code as a file attachment in I-Learn. At the top of your program, include a comment answering these five questions:

```
# 1. Name:
#      -your name-
# 2. Assignment Name:
#      Lab 06: Advanced Search
# 3. Assignment Description:
#      -describe what this program is meant to do-
# 4. Algorithmic Efficiency
#      -Identify the algorithmic efficiency and tell why it is as you say it is-
# 5. What was the hardest part? Be as specific as possible.
#      -a paragraph or two about how the assignment went for you-
# 6. How long did it take for you to complete the assignment?
#      -total time in hours including reading the assignment and submitting the program-
```

Notice that Question 4 is new. You need to compute the algorithmic efficiency for this advanced search. You also need to provide some evidence that it is what you say it is.

Demonstration Video

Record a short video demonstrating the execution of your program.. The video must be very short. No video longer than two minutes will be accepted. This means you might need to practice once or twice before recording the video to make sure that you demonstrated everything that is necessary. As the video is recorded, mention the test case you are covering.

Your demonstration video must cover the following test cases:

Case	File	Search Word	Found?
Empty list	Lab06.empty.json	Empty	No
Single item found	Lab06.trivial.json	trivial	Yes
Single item not found	Lab06.trivial.json	missing	No
Small list found	Lab06.languages.json	C++	Yes
Small list not found	Lab06.languages.json	Lisp	No
Big list found	Lab06.countries.json	United States of America	Yes
Big list not found	Lab06.countries.json	United States	No

After the video is recorded, provide a voice-over mentioning what test case you are covering.

Assessment

Your grade for this activity will be according to the following rubric:

	Exceptional 100%	Good 90%	Acceptable 70%	Developing 50%	Missing 0%
Code Quality 30%	Perfection! The code is extremely easy to understand and very straightforward	Professional and efficient	A few obvious mistakes were made	Readable	Little effort was spent on style. The code looks thrown together or is missing
Efficiency 20%	Algorithm is efficient and the efficiency is correctly computed	Algorithmic efficiency was not correctly computed but an efficient algorithm was used	The algorithm is not efficient but was correctly computed	Incorrect algorithmic efficiency and an inefficient algorithm was used	Algorithmic efficiency was not computed for this problem
Functionality 40%	All the test cases execute perfectly	Everything works but there are minor cosmetic defects	One test case fails to execute as expected	At least one test case works as expected	Code does not run, is missing, or does not resemble the advanced search algorithm
Reflection 10%	The reflection component of the video completely and concisely describes what went well and what went poorly	It is clear that thought went into the reflection component of the video	Each reflection question is addressed, but there is no evidence of introspection	At least one reflection question is answered	There is no voice-over in the video or the voice-over fails to address any of the required points