Assignment 9 – Files

Learning Objective

Read data from a CSV (comma-separated value) file and write data to another file. Practice defining and calling functions and working with lists.

Assignment Description

Write a language translator program that translates English words to another language using data from a CSV file. Read in a CSV file with words in 15 languages to create a list of words in English. Ask the user to select a language and read the CSV file to create a list of words in that language. Ask the user for a word, translate the word, display to the user, write it to an output file, and repeat until the user is done. Since the data is in the same file, the index from the English list will match the index from the other language list.

You are provided a CSV with words in English and 14 other languages.

- Each row represents one line in a table, and commas separate each column.
- The first line in the CSV file represents the header which contains the supported languages. Each subsequent row represent an English word and its translations.
- If there is no translation for a word, then there is a dash (-).

CSV File (Example)

```
English,Danish,Dutch,Finnish,French,German
animal,dyr,dierlijk,eläin,bête,tier
cat,kat,kattekop,kissa,chat,katze
dog,hund,hond,koira,clebs,hund
snake,slange,slang,-,guivre,schlange
```

CSV Data in a Table (Example)

English	Danish	Dutch	Finnish	French	German
animal	dyr	dierlijk	eläin	bête	tier
cat	kat	kattekop	kissa	chat	katze
dog	hund	hond	koira	clebs	hund
snake	slange	slang	-	guivre	schlange

All operations must be done using techniques in course materials. You may NOT import and use the CSV or other modules to process the CSV file.

You are not allowed to create any global variables. You need to get the names of the languages from the CSV file. You cannot hard-code any of the names of the languages except when assigned "English" as the default language.

Steps

1. In PyCharm (Community Edition), open an existing project (such as ITP115) or create a new project.
   - If you open an existing project, then create a new directory (probably under the Assignments directory) named **a9_*last_first*** where *last* is your last/family name and *first* is your preferred first name. Use lowercase letters.
   - If you create a new project, then name it **a9_*last_first*** where *last* is your last/family name and *first* is your preferred first name. Use lowercase letters.
2. In the project or directory, create a new Python file called **assignment9.py**. At the top of the file, put comments in the following format and replace the name, email, and section with your actual information:

```
# Name, USC email
# ITP 115, Spring 2022
# Section: number or nickname
# Assignment 9
# Description:
# Describe what this program does.
```

3. Put the **languages.csv** file in your a9_*last_first* directory.
   - Download the file from Blackboard under the item for this assignment.
   - Add it to the correct directory by dragging to a9_*last_first* directory in PyCharm.
4. Define the **getAllLanguages(fileName)** function.
   - Parameter: fileName is a string containing the name of a CSV file to read from and it has a default value of "languages.csv"
   - Return value: a list of strings representing the languages in the header row
   - Open the CSV file and get the header row with the languages.

- Use the str.split() method to get the list of languages from the header row.
- Close the file and return the list.

5. Define the **getTranslationLanguage(languagesList)** function.
   - Parameter: languagesList is a list of the languages
   - Return value: a string for the language to translate words into
   - Display to the user the languages that are available for translation, which are all of the languages in languagesList parameter except English. You can use slicing to get a list without English. Do not hard-code the languages.

```
Translate English words to one of the following languages:
Danish Dutch Finnish French German Indonesian Italian
Japanese Latin Norwegian Portuguese Spanish Swahili Swedish
```

   - Get input from the user for the language for translation. They must enter in a valid language. The user's input is not case sensitive.

```
Enter a language: chinese
This program does not support Chinese
Enter a language: finnish
```

   - Return the language such that the first letter is capitalized and the remaining letters are lower case. You can use the str.title() or str.capitalize() method.

6. Define the **readDataFile(languagesList, languageStr, fileName)** function.
   - Parameter 1: languagesList is a list of the languages
   - Parameter 2: languageStr is a string of containing the name of a language and it has a default value of "English"
   - Parameter 3: fileName is a string containing the name of a CSV file to read from and it has a default value of "languages.csv"
   - Return value: a list of words in the language identified by the languageStr parameter
   - Create a variable for the return value. Initially set it to an empty list.
   - Open the CSV file and read the header row to skip it.
   - Use the languagesList and languageStr parameters to determine the index of the language. Use the list.index() method.

- Loop through the rest of the file. Each line of data will have an extraneous new line at the end, so use the str.strip() method to remove it. Use the str.split() method to break up the line into a list of strings using "," for a separator. Get the correct word and append it to the list.
- Close the file and return the list.

7. Define the **createTextFile(language)** function.
   - Parameter: language is a string containing the name of the language for translation
   - Return value: None
   - Create a variable for the results text file. Use the language with ".txt". If the language is Italian, then the default file name is "Italian.txt".
   - Open the results text file such that if there is an existing file, it will be overwritten.
   - Write text to the file stating the language that English words will be translation to. Here is an example:

```
Words translated from English to German
```

   - Close the file.

8. Define the **translateWords(englishList, translationList, language)** function.
   - Parameter 1: englishList is a list of words in English
   - Parameter 2: translationList is a list of words in another language (not English)
   - Parameter 3: language is a string containing the language of the words in the translationList
   - Return value: none
   - Open the results file in order to append text into it. The name of the file is the language parameter appended by ".txt".
   - Ask the user to enter an English word to translate.

```
Enter a word to translate: turtle
```

   - If the word is not in the English list, then display a message to the user. Here is an example message:

```
turtle is not in the English list
```

- Translate the word. If there is a translation, then display a message to the user and write the word and its translation to the file.
    - Example message to user:

```
rabbit is translated to kani
```

    - Example text to file:

```
rabbit = kani
```

- If there is no translation (i.e., the value is "-"), then display a message to the user.

```
snake did not have a translation
```

- Ask the user if they want to translate another word and repeat if they answer "y" or "Y".

```
Another word (y or n)? y
```

- Display a message to the user with the name of the results file. Here is an example:

```
Translated words have been saved to Finnish.txt
```

- Close the file.

9. Define and call the **main()** function.
    - Print the following message to the user:

```
Language Translator
```

    - Call the getAllLanguages() function to get the list of languages. If your CSV file is named "languages.csv", then you do not need to have an argument. Let the function use the default value for the file name. The getAllLanguages() function has a return value, so make sure to use a variable to hold the list of languages that is returned.
    - Call the readDataFile() function to read the CSV file for the English words. For the languagesList parameter, use the list that was returned from calling the getAllLanguages() function. For the languageStr, you do not need a matching

argument since you want to use the default value of "English". If your CSV file is named "languages.csv", then you do not need to have an argument for fileName. The readDataFile() function has a return value, so make sure to use a variable to hold the list of English words that is returned.

- Call the getTranslationLanguage() function to get the language to use for translation. For the languagesList parameter, use the list that was returned from calling the getAllLanguages() function. The getTranslationLanguate() function has a return value, so make sure to use a variable to hold the returned language.
- Call the readDataFile() function to read the CSV file for the translation language. For the languageStr parameter, use the variable that holds the language for translation. The readDataFile() function has a return value, so make sure to use a variable to hold the list of words (in the translation language) that is returned.
- Call the createTextFile() function to create and write a message to the results file. For the language parameter, use the variable that holds the language for translation.
- Call the translateWords() function to allow the user to enter words, translate those words, and save the translations to the results text file. Use the variables you have created in the main() function as the appropriate arguments.

10. Be sure to comment your code. This means that there should be comments throughout your code. Put a comment block before each function stating the parameters, return values, and what that function does. Points will be deducted for not having comments.
11. Follow coding conventions. You should use lowerCamelCase or snake_case for variable names. You are welcome to create any variables that you need.
12. Test the program. Look at the Sample Output below. Assignments that do not run are subject to 20% penalty.
13. Prepare your submission:
    - Find the **a9_*last_first*** folder on your computer and compress it. This cannot be done within PyCharm.
    - On Windows, use **File Explorer** to select the folder. Right click and select the Send to -> Compressed (zipped) folder option. This will create a zip file.

- On Mac OS, use **Finder** to select the folder. Right click and select the Compress "*FolderName*" option. This will create a zip file.

14. Upload the zip file to your Blackboard section:
    - On Blackboard, navigate to the appropriate item.
    - Click on the specific item for this assignment.
    - Click on the **Browse Local Files** button and select the zip file.
    - Click the **Submit** button.

Grading

- This assignment is worth 40 points.
- Make sure that you the program runs. Points will be taken off if the graders have to edit the source code to test your program.
- Make sure to submit your assignment correctly as described above. Points will be taken off for improper submission.
- You may NOT import and use the CSV modules or other modules to process the CSV file. Use the techniques from the course materials.

Item	**Points**
languages.csv in directory	2
getAllLanguages()	5
getTranslationLanguage()	5
readDataFile()	10
createTextFile()	3
translateWords()	10
main()	5
Total	**40**

Sample Output

Example 1 – Screen Output:

```
Language Translator
Translate English words to one of the following languages:
Danish Dutch Finnish French German Indonesian Italian
Japanese Latin Norwegian Portuguese Spanish Swahili Swedish
Enter a language: chinese
This program does not support Chinese
Enter a language: finnish

Enter a word to translate: rabbit
rabbit is translated to kani
Another word (y or n)? Y

Enter a word to translate: snake
snake did not have a translation
Another word (y or n)? y

Enter a word to translate: turtle
turtle is not in the list
Another word (y or n)? y

Enter a word to translate: bird
bird is translated to lintu
Another word (y or n)? q

Translated words have been saved to Finnish.txt
```

Example 1 – Finnish.txt File:

```
Words translated from English to Finnish
rabbit = kani
bird = lintu
```

Example 2 – Screen Output:

```
Language Translator
Translate English words to one of the following languages:
Danish Dutch Finnish French German Indonesian Italian
Japanese Latin Norwegian Portuguese Spanish Swahili Swedish
Enter a language: spain
This program does not support Spain
Enter a language: SWAHILI

Enter a word to translate: earth
earth is translated to kiwanja
Another word (y or n)? y

Enter a word to translate: peace
peace did not have a translation
Another word (y or n)? y

Enter a word to translate: school
school is translated to shule
Another word (y or n)? Y

Enter a word to translate: city
city is translated to mji
Another word (y or n)? y

Enter a word to translate: teacher
teacher is not in the list
Another word (y or n)? N

Translated words have been saved to Swahili.txt
```

Example 2 – Swahili.txt File:

```
Words translated from English to Swahili
earth = kiwanja
school = shule
city = mji
```