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 using "English" as an argument for the readFileForWords() function in the main() function.

Steps

1. In PyCharm (Community Edition), open your existing ITP115 project.
2. Under the Assignments directory, create a new directory called **a9_*last_first*** where *last* is your last/family name and *first* is your preferred first name. Use all lowercase letters.
3. In the directory, create a new Python file called **assignment9.py**.
4. 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, Fall 2022
# Section: number or nickname
# Assignment 9
# Description:
# Describe what this program does.
```

5. 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.
6. Define the **readFileForLanguages(filenameStr)** function.
   - Parameter: filenameStr is a string containing the name of a CSV file to read from
   - Return value: a list of strings containing the languages from the header row of the CSV file
   - Open the CSV file.
   - Get the header row using the file.readline() method. Make sure to strip the string to remove the new line ("\n") at the end.

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

7. Define the **getLanguageFromUser(languagesList)** function.
   - Parameter: languagesList is a list of the languages
   - Return value: a string containing the language to translate words into
   - Use slicing to create a list variable that has all of the languages from the languagesList parameter except English. (Do not remove "English" from the languagesList parameter since that change will exist outside of the function.)
   - Display to the user the languages that are available for translation using the variable. It should look like the following (the list can be displayed on one line):

```
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 except English. You need to remove whitespace at the beginning and end of the user's input. Their input is not case sensitive while the languages in the list start with a capital letter. You will need to use the str.title() or str.capitalize() method. Use a loop to repeat until they have entered a valid language. Use the appropriate list variable.

```
Enter a language: chinese
Chinese is not a valid language
Enter a language: English
English is not a valid language
Enter a language:   finnish
```

   - After the loop, display a message with the language. Here is an example:

```
You have entered Finnish
```

   - Return the language.
   - This function will be called in the main() function.

8. Define the **readFileForWords(filenameStr, languagesList, languageStr)** function.
   - Parameter 1: filenameStr is a string containing the name of a CSV file to read
   - Parameter 2: languagesList is a list of the languages
   - Parameter 3: languageStr is a string of containing the name of a language
   - Return value: a list of words in the language identified by the languageStr parameter
   - Create a list variable for the return value and 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 "," (a comma) for a separator. Get the correct word by using the index. Append it to the list.
   - Close the file and return the list.
   - This function will be called twice in the main() function. The first time to get a list of English words and the second time to get a list of words in another language.
9. Define the **createFileWithTranslations(languageStr, englishList, translationList)** function.
   - Parameter 1: languageStr is a string containing the language to translate to
   - Parameter 2: englishList is a list of words in English
   - Parameter 3: translationList is a list of words in the language to translate to
   - Return value: None
   - Open a file for writing text. The name of the file is the languageStr parameter appended by ".txt". You can use string concatenation. Open the file such that if there is an existing file with that name, it will be overwritten.
   - Write text to the file stating the language that English words will be translation to. Use the languageStr parameter. Here is an example:

```
Words translated from English to Finnish
```

- Ask the user to enter an English word to translate.

```
Enter a word to translate (return to quit): turtle
```

- Make sure to remove the whitespace from the beginning and end of the string as well as lowercase the string.
- Create a loop to allow the user to enter a word to translate until they press the return key. You will do this by checking if the string is equal to the empty string ("").
- In the loop, check if the word is in the English list.
- If the word is in the English list, then translate the word. To get the translated word, use the list.index() method to find the index of the word in the English list. Then use that index to access the value in the translation list.
- Check if there is a translation. A valid translation does not equal "-".
- If there is a translation, then display a message to the user and write the word and its translation to the file.
  - Here is an example of the message to user:

```
rabbit is translated to kani
```

  - Here is an example of the text that gets written to file:

```
rabbit -> kani
```

- If there is no translation (i.e., the value equals "-"), then display a message to the user. Do not write to the file. Here is a sample message to the user:

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

- If the word is not in the English list, then display a message to the user. Do not write to the file. Here is a sample message to the user:

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

- Get another word from the user. Make sure to remove whitespace and lowercase.

```
Enter a word to translate (return to quit): bird
```

- After the loop, 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.

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

```
Language Translator
```

- Create a string variable that holds the name of the CSV file.
- Call the readFileForLanguages() function to get the list of languages. Use the string variable holding the name of the CSV file for the argument. This function has a return value, so make sure to use a variable to hold the list of languages that is returned.
- Call the readFileForWords() function to read the CSV file for the English words. For the filenameStr, use the string variable you created in the main() function. For the languagesList parameter, use the list that was returned from calling the readFileForLanguages() function. For the languageStr, use "English". The readFileForWords() function has a return value, so make sure to use a variable to hold the list of English words that is returned.
- Call the getLanguageFromUser() function to get the language to use for translation from the user. For the languagesList parameter, use the list that was returned from calling the readFileForLanguages() function. The getLanguageFromUser() function has a return value, so make sure to use a variable to hold the returned language.
- Call the readFileForWords() function to read the CSV file for the translation language. For the languageStr parameter, use the variable that holds the language for translation. This 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 createFileWithTranslations() function to allow the user to enter words, translate those words, and save the translations to a text file. Use the variables you have created in the main() function as the appropriate arguments.

11. 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.
12. Follow coding conventions. You should use lowerCamelCase or snake_case for variable names. You are welcome to create any variables that you need.
13. Test the program. Look at the Sample Output below. Assignments that do not run are subject to 20% penalty.
14. 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.
15. 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 must include the languages.csv file in your project directory.
- 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	1
readFileForLanguages()	5
getLanguageFromUser()	8
readFileForWords()	10
createFileWithTranslations()	10
main()	6
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
Chinese is not a valid language
Enter a language: English
English is not a valid language
Enter a language:   finnish
You have entered Finnish

Enter a word to translate (return to quit): rabbit
rabbit is translated to kani

Enter a word to translate (return to quit): snake
snake did not have a translation

Enter a word to translate (return to quit): turtle
turtle is not in the list

Enter a word to translate (return to quit): bird
bird is translated to lintu

Enter a word to translate (return to quit):

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
Spain is not a valid language
Enter a language: SWAHILI
You have entered Swahili

Enter a word to translate (return to quit): earth
earth is translated to kiwanja

Enter a word to translate (return to quit): peace
peace did not have a translation

Enter a word to translate (return to quit): school
school is translated to shule

Enter a word to translate (return to quit): city
city is translated to mji

Enter a word to translate (return to quit): teacher
teacher is not in the list

Enter a word to translate (return to quit):

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
```