Chapter 7 - Files

Files

Lab - Files

Lab 1

Lab 1

A previous lesson stated that the `seek` method can be used to move Python back to the beginning of a text file so it can be read again. Another way to do this is to read the contents of the text file to a list. Python can iterate over a list as many times as it wants without having to use the `seek` method.

Create the variable `text` and assign it the value of an empty list.

```
text = []
```

Open the text file `files-lab1.txt` in read mode. Read all of the text at once, and assign that to `text`.

```
with open("student_folder/.labs/files-lab1.txt", "r") as input_file:

    text = input_file.readlines()
```

The contents of the text file now reside in the variable `text`. To prove this, go to the next line and remove the indentation. This will close the text file. Next, print the first element of the variable `text`.

```
print(text[0])
```

You should see a passage from Bram Stoker's *Dracula*. You can access the paragraph again by referencing the variable `text`.

```
print(text[0].upper())
```

Storing the value of a text file in a list variable is another way to access text without having to close and then open the text file.

▼ **Code**

```
text = []

with open("student_folder/.labs/files-lab1.txt", "r") as input_file:

    text = input_file.readlines()

print(text[0])
print(text[0].upper())
```

Lab 2

Lab 2

This lab uses a comma delimited CSV file `files-lab2.csv`, which contains integers. There are three columns and four rows. The program below will print the sum for each row in the CSV.

Start by importing the `csv` module. Open the CSV file in read mode, and pass the file to a `csv.reader`.

```
import csv

with open("student_folder/.labs/files-lab2.csv", "r") as input_file:

    reader = csv.reader(input_file)
```

Next, add a nested loop. The outer loop relates to the rows of the CSV file, while the inner loop will go through the numbers in each row. Declare the variable `total` after the first loop, but before the second loop. Set `total` to 0.

```
    for row in reader:
        total = 0
        for num in row:
```

Add each number in the row to the variable `total`. Values in CSV files are stored as strings, so typecast `num` as an integer so you can perform addition.

```
        for num in row:
            total += int(num)
```

After the inner loop has finished running, print the value of `total` with a short explanation.

```
        for num in row:
            total += int(num)
        print("The total value is: {}".format(total))
```

Run the program. You should see the following output:

```
The total value is: 10
The total value is: 151
The total value is: 63
The total value is: 127
```

▼ Code

```
import csv

with open("student_folder/.labs/files-lab2.csv", "r") as input_file:

    reader = csv.reader(input_file)
    for row in reader:
        total = 0
        for num in row:
            total += int(num)
        print("The total value is: {}".format(total))
```

Lab 3

Lab 3

The following program will ask the user to enter the name of a superhero and then to enter their superpower. These two pieces of information will be written to the CSV file called `superheroes.csv`. When the user enters `stop` for the superhero name, the program will stop running.

Start by importing the `csv` modules. Open the `superheroes.csv` file in write mode as `output_file`.

```
import csv

with open("student_folder/csv/superheroes.csv", "w") as output_file:
```

Setup a `csv.writer`, and be sure to set the `lineterminator` to `\n` to avoid the mixed line endings warning. The first row in the CSV file should be headers for each column of data.

```
    writer = csv.writer(output_file, lineterminator="\n")
    writer.writerow(["Superhero", "Power"])
```

Now is the time to ask the user to input the name of the superhero and their superpower. You should also tell the user how to quit the program.

```
    print("Enter `stop` to quit the program")
    name = input("Enter the superhero's name: ")
    power = input("Enter their superpower: ")
```

The program should run again and again until the user enters `stop`. Since the for loop runs for a predetermined amount of time, an infinite while loop is a better choice. The stop condition is when `name` is equal to `stop`.

```
    while True:
```

Now, write a line to the CSV file with `name` and `power`. Then ask the user to enter another superhero name. Be sure to check and if it is equal to `"stop"`. Exit the loop with a `break` statement. Finally, ask the user to enter the superpower.

```
        writer.writerow([name, power])
        name = input("Enter the superhero's name: ")
        if name == "stop":
            break
        power = input("Enter their superpower: ")
```

Run the program and enter some information. Click the link below to view the CSV file.

Open superheroes.csv

▼ **Code**

```
import csv

with open("student_folder/csv/superheroes.csv", "w") as output_file:

    writer = csv.writer(output_file, lineterminator="\n")
    writer.writerow(["Superhero", "Power"])
    print("Enter `stop` to quit the program")
    name = input("Enter the superhero's name: ")
    power = input("Enter their superpower: ")
    while True:
        writer.writerow([name, power])
        name = input("Enter the superhero's name: ")
        if name == "stop":
            break
        power = input("Enter their superpower: ")
```

Lab 4

Lab 4

The lab will cover how to encrypt the contents of a file with a Caesar cipher. This cipher will work with any characters that are letters (uppercase and lowercase), the digits 0 to 9, and the symbols space, exclamation, question mark, and the period. Other symbols will not be encrypted. The Caesar cipher requires a key to work as well. The key is any number from 0 to 25.

The Caesar cipher works by having a list of all of the symbols that can be encrypted. Take the letter you want to encrypt and find its place in the list. Add the value of the key to the position to get the encrypted letter. If the new position is greater than the end of the list, keep counting from the beginning of the list. The example below assumes a key of 13, and shows that a `"K"` becomes `"X"` with the Caesar cipher.

Caesar Cipher

Reading the Source File

This program will read from a file called `source.txt` with the path of `student_folder/.labs`. The encrypted text will be written into a file called `encrypted.txt` with the path of `student_folder/text`.

```
with open("student_folder/.labs/source.txt", "r") as source, open("stu
```

Next, you need set the key (a number from 0 to 25), the cipher mode (encrypt or decrypt), the list of symbols and an empty string of the new characters (either encrypted or decrypted).

```
key = 13
mode = "encrypt"
SYMBOLS = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz123

new_text = ""
```

▼ **Why is `SYMBOLS` in all caps?**

There is a type of variable called a constant. This variable should never change its value. The Python community represents constants by using all caps when writing the variable name.

Read the first line from the `source` file. If it is not an empty string (the end of the text file), then you are going to loop through each character of the line.

```
line = source.readline()
while line != "":
    for char in line:
```

The Caesar Cipher

The Caesar cipher can only encrypt those characters that are in the `SYMBOLS` variable. Check to see if `char` is in `SYMBOLS` and then find its index if true. If the program is encrypting the text, add the value of the key to the index. If the program is decrypting the text, subtract the value of the key from the index.

```
            if char in SYMBOLS:
                char_index = SYMBOLS.find(char)

                if mode == "encrypt":
                    new_index = char_index + key
                elif mode == "decrypt":
                    new_index = char_index - key
```

It is possible that `char_index` is less than 0 or greater than the length of `SYMBOLS`. These indexes will cause a problem. If `char_index` is negative, add it to the length of `SYMBOLS`. If `char_index` is greater than the length of `SYMBOLS`, subtract the length of `SYMBOLS`.

```
                if new_index >= len(SYMBOLS):
                    new_index = new_index - len(SYMBOLS)
                elif new_index < 0:
                    new_index = new_index + len(SYMBOLS)
```

Now that the character has been transformed according to the key, add the character to `new_text`. Once the for loop has finished running, write the new text to `destination`. Finally, read the next line in `source.

```
                new_text += SYMBOLS[new_index]

        destination.write(new_text)
        line = source.readline()
```

Open the encrypted file

▼ **Code**

```
with open("student_folder/.labs/source.txt", "r") as source, open("s

    key = 13
    mode = "encrypt"
    SYMBOLS = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1

    new_text = ""

    line = source.readline()
    while line != "":
        for char in line:
            if char in SYMBOLS:
              char_index = SYMBOLS.find(char)

              if mode == "encrypt":
                  new_index = char_index + key
              elif mode == "decrypt":
                  new_index = char_index - key

              if new_index &gt;= len(SYMBOLS):
                  new_index = new_index - len(SYMBOLS)
              elif new_index &lt; 0:
                  new_index = new_index + len(SYMBOLS)
              new_text += SYMBOLS[new_index]

        destination.write(new_text)
        line = source.readline()
```

Decrypting the File

To decrypt the file, a few changes need to be made to your code. The source file should be the encrypted file, and the destination file will be the file `decrypted.txt`.

```
with open("student_folder/text/encrypted.txt", "r") as source, open("s
```

Finally, change the mode of the cipher to `"decrypt"`.

```
mode = "decrypt"
```

Run the program and take a look at the output below.

Open the decrypted file

▼ **Source Text**

The original text for this lab is the opening paragraph from L. Frank Baum's *The Wizard of Oz*.

Lab Challenge

Lab Challenge

Write a program that reads the text file `student_folder/.labs/myanmar.txt`. The file contains several instances of the word `Burma`. Replace each instance of `Burma` with `Myanmar`, and print the results of this transformation. The final output of your program should be:

```
Myanmar is a country in Southeast Asia.
The capital of Myanmar is Naypyidaw.
Its population is about 54 million people.
Myanmar is a former British colony.
```

▼ **Where is the code visualizer?**

Unfortunately, the code visualizer does not work with the `open` command, so it cannot be used for this problem.