

AIST 2120

Programming Assignment 6 (Ch. 17)

Encryptinator Plus: Shhh

*Created by Steve Weldon; revamped by Justin Henry and Misty Lawrence (Jan. 2022)
Modified by Jonathan Sloan (Sep. 2023)*

INTRODUCTION

This assignment presents another opportunity to connect some dots. Chapter 15 introduced the wonderful world of working with PDF documents, Chapter 16 JSON files, and Chapter 17 keeping time. Let's again put some pieces together.

ASSIGNMENT & DISCUSSION

The international spy agency KAOS has been working to intercept and decrypt documents sent from Maxwell Smart's employer, CONTROL – especially those being sent to the cunning Mr. Peabody. You have been hired to assist Mr. Smart in his efforts by unzipping the files in his directory, finding the PDF files, encrypting them, then rezip all the PDF files into a single archive file, ready to be forwarded to Mr. Peabody.

Your mission in this homework assignment is to practice working with PDF documents, creating JSON files, and keeping time.

NOTE: Some of this Programming Assignment is similar to Lab 9 (Ch 15). Consider if Lab 9 includes any code which could prove beneficial for this assignment.

Write a program to unzip the provided archive file and 'discover' the PDF documents in a folder containing a variety of file types. Apply encryption to the PDF documents; disregard all non-PDF documents. To make this interesting, keep tabs on how long it takes to process (read, encrypt, write) *each* PDF document. Also create a JSON file that keeps track of each PDF and the length of time it took to process it. Sounds super fun, doesn't it?

Of note, this assignment works with three seminal papers in the computing world:

- Alan Turing authored two of the papers. Turing is considered the Father of Computer Science. His 1936 paper introduced such topics as Turing Machines and Universal Turing Machines. His 1950 paper introduced the Turing Test.
- Claude Shannon, the Father of Information Theory, contributed the third paper, which describes the basic logic functions at the heart of all electronic digital computer designs.

All three are worth adding to your library and reading!

Tasks

- **IMPORTANT:** Start with the provided **PA6_Files ZIP** archive. Manually create a Prog6 folder on your desktop and place PA6_Files.zip in that folder. This zip file holds information and files needed for Programming Assignment 6.
 - Note the folder contents: a 00 – READ_ME.txt file, **Secret_In** subfolder, and **Secret_Sent** subfolder.

- The *Secret_In* folder contains a 00 - READ_ME.txt file along with other .docx, .pdf, .txt, and .xlsx files. Do not move, remove, or alter these files.
 - The *Secret_Sent* folder only contains a 00 – READ_ME.txt file. This folder will hold your processed files. In other words, write the encrypted files to this folder.
- Name your source code (solution) file **encrypt_plus6_YourName.py**.
- Write your program to accomplish the following steps:
1. Change directory to your afore-mentioned Prog6 folder. To verify the change, print the CWD to the screen. Thereafter, *use relative paths* in your source code; i.e., *do not use absolute paths*.
 2. Inside the program, open the **PA6_Files ZIP** archive, decompress (unzip) and save the files to your Prog6 folder.
 3. Your program should “look” through the *Secret_In* folder considering all files in it. Remember, no hardcoding of filenames included inside this folder.
 4. Process all **.pdf** files found in the *Secret_In* folder, apply encryption with the password noted below, and write the resulting .pdf files to the *Secret_Sent* folder. For any folder names you use more than once, assign them to a variable near the beginning of the program and use the variable thereafter.
 - a. Print to the screen the names of all files in the *Secret_In* folder.
 - b. Open all **.pdf** files.
 - c. Set the password “**enigma**” to a variable near the beginning of the program and use the variable thereafter.
Apply encryption to each **.pdf** file using the provided password.
 - d. Write the newly encrypted PDF file to the *Secret_Sent* folder using the original filename with a prefix of “**encrypted_**”.
Example: A file named *abc.pdf* should have the new filename of *encrypted_abc.pdf*.
 - e. Keep track of how long it took to process each PDF (steps *a* through *d*).
Hint: Refer to Chapter 17 treatment of the **time** module.
Write the processing times to a dictionary using the original filenames as the keys and the lengths of time to process each file as the values. Round the times to *two* digits to the right of the decimal point, ex. 1.2684543 rounds to 1.27.
 9. Write the filenames of the encrypted PDFs in the *Secret_Sent* folder to the screen and write the dictionary to the screen.
 10. After document processing is complete, convert the dictionary from Step 4e to JSON data format and write a JSON file into the **Prog6** folder. Name the JSON file **time_fileyourInitials.json**, e.g., *time_fileML.json*.
 11. Compress the *Secret_Sent* folder, ready for transmission to Mr. Peabody, and notify the user.

12. Notify the user that all program processing is complete.
13. Execute a clean exit from the program as in previous assignments.

Hints

- The PDF work in this assignment is rather straightforward. Remember note re: Lab 9!
- Remember previous work and chapters regarding file operations.
- Consider where *Walking a Directory Tree* covered in Chapter 10 might be useful in determining which files to process.
- An interesting point: what extension do PDF files always '*end with*'? Could come in handy as you determine which *Secret_In* folder files to treat as PDFs.

RECOMMENDATIONS

- It will help to break down your project into phases ... one might call high level algorithm steps ... and then break it down further into more detailed/specific algorithmic steps.
- Pay attention to the purpose of the code.
- Include informative comments, including *docstring(s)*, and whitespace to aid understanding and legibility.
- Remember that screen output is for the user; make it visually pleasing, clear, and informative.

RUBRIC

The general rubric for this assignment is:

Header and filename:	5
Comments:	10
White space/legibility:	10
Works & coding technique:	75
Total	100

REQUIREMENTS

☐ Application

- Write your program in Python 3.
- There is no need to turn in the algorithm for this assignment.
- Ensure your file is named properly: **encrypt_plus6_YourName.py**.
- If you choose to use a different file editor, note that your program will be evaluated in IDLE. Remember to run your finished program in IDLE to ensure no unfortunate surprises. If it doesn't run in IDLE, it doesn't run.

☐ Submission

- Submit your source code file via D2L; i.e., submit only one deliverable to D2L. Neither email nor hardcopy submissions will be accepted.

- Ensure you retain a complete copy of your program source code file(s).

DUE DATE: Per D2L instructions.

- **Late Penalty:** The penalty for late submissions is generally 10% per day for up to four days (96 hours from assignment due date and time). I will not accept submissions more than one week, 7 calendar days (168 hours), after the original due date ***nor after the day of the Final*** (whichever comes first). Ensure your work is complete and on time!





STYLE GUIDE

- **File headers.** Include a header in the below format at the top of all .py source code files.










```
# Your Name  
# AIST 2120, Pgm. Asgn. 6    #! Python 3  
# Submission Date
```

- **Sample Output.** The below output is provided as a sample of how your output could look. Your output is not required to look exactly like the sample; however, this sample meets assignment requirements for content and screen output.
 - Keep in mind that the output is to inform the user and should be both clear and pleasing to those reading it.
 - See next page



Proj6 Folder Contents after PA6_Files.zip unzip

Prog6 >	
<input type="checkbox"/>	Name
	PA6_Files.zip
	00 - READ_ME.txt
	Secret_Sent
	Secret_In

Provided Secret_In Folder Contents

Prog6 > Secret_In	
<input type="checkbox"/>	Name
	.DS_Store
	00 - READ_ME.txt
	distractor_1.txt
	distractor_2.txt
	Nada.xlsx
	Nothing To See Here.docx
	Shannon_1938.pdf
	Turing_Paper_1936.pdf
	Turing_Paper_1950.pdf

Provided Secret_Sent Folder Original Contents

Prog6 > Secret_Sent	
<input type="checkbox"/>	Name
	.DS_Store
	00 - READ_ME.txt

Example Screen Output

```
-----
---          AIST 2120          ---
---   Programming Assignment 6   ---
---   Encryptinator Plus        ---
-----

Ensured current dir is "Desktop\Prog6": C:\Users\jonsloan\Desktop\Prog6

Unzipping files ...

=> Input files sent by CONTROL:
    .DS_Store
    00 - READ_ME.txt
    distractor_1.txt
    distractor_2.txt
    Nada.xlsx
    Nothing To See Here.docx
    Shannon_1938.pdf
    Turing_Paper_1936.pdf
    Turing_Paper_1950.pdf

=> Encrypted PDF files ready to send to Mr. Peabody:
    encrypted_Shannon_1938.pdf
    encrypted_Turing_Paper_1936.pdf
    encrypted_Turing_Paper_1950.pdf

    Time to process each file:
{
    'Shannon_1938.pdf': 0.39,
    'Turing_Paper_1936.pdf': 2.49,
    'Turing_Paper_1950.pdf': 0.06}

=> Secret Out files compressed and ready for transmission

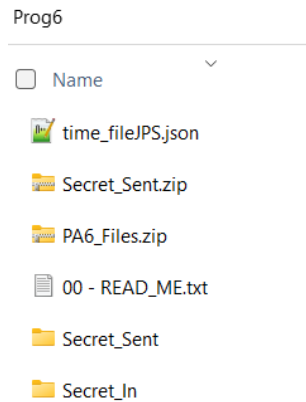
>> JSON time_file created and placed in C:\Users\jonsloan\Desktop\Prog6

*** All CONTROL file processing COMPLETE ***
***   KAOS has been thwarted again!   ***

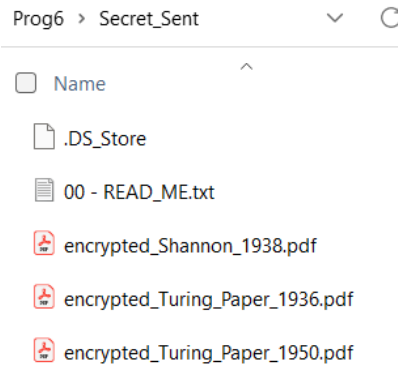
-----
---   Programming Assignment 6   ---
---   Encryptinator Plus        ---
---   Complete                  ---
-----

Press the enter key to exit:
```

Example Prog6 Folder Post Run Contents



Example Secret_Sent Folder Post Run Contents



Example timefile.JSON File Contents

