Programming in Python: Message Encryptor

This project will practice working with lists and reading and writing text files.

You are working as a secret agent at a top secret government firm. For your current mission, you have been assigned a partner to correspond with over your super secure email server. However, you do not want your message to be read if hackers happen to intercept it, as it contains extremely sensitive information. You must find a way to encrypt your messages so only those with the secret key will be able to read them.

Some terminology:
Plaintext – the readable, English text to be encoded
Encoded text – the scrambled text, unable to be read unless you have the secret key
Key – a secret word that is used to decode some text

For this project, assume the text files and the keys will all be **capital letters only**. Ignore (do not encode) punctuation and spaces in the files. Keys will only consist of **alphabetic characters**.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Your algorithm should work as follows:

- Pick a secret word to be your key
- Pick a file to encode
- Line up successive copies of your key with the plaintext
- Run encode/decode on the text file

For example, if your key is **KEY** and your text is **TOP SECRET**

T	O	P		S	E	C	R	E	T
K	E	Y		K	E	Y	K	E	Y

To **encode** the text, look at each plaintext character in the file and it's matching key letter, **add** their numeric values together, and take the sum **modulo 26**. The result is the numeric value for the **encoded letter**.

Ex: **T** = 19, **K** = 10, **T** + **K** = 29 % 26 = 3 → new letter is **D**

O = 14, **E** = 4, **O** + **E** = 16 % 26 = 18 → new letter is **S**

D	S	N		C	I	A	B	I	R
K	E	Y		K	E	Y	K	E	Y

To **decode** the text, do the opposite of encode: get the numeric value of the letter in the encoded text, **subtract** the value of the matching key letter, and take the difference **modulo 26**.

Ex: **D** = 3, **K** = 10, **D** - **K** = -7 % 26 = 19 → new letter is **T**
S = 18, **E** = 4, **S** - **E** = 14 % 26 = 14 → new letter is **O**

In the file **message_encryptor.py**, finish the following functions:

encrypt()
This function will ask the user for a **file to encrypt** and the **key to encrypt it with**. The encrypted file should be written to a new file called "encoded_PREVIOUSFILENAME.txt". For example, if the file to encrypt is "message.txt", the new file should be called "encoded_message.txt".

Your function output should look something like this:

```
ENCRYPT MESSAGE
What is the key you'd like to use?: KEY
What is the name of the file to be encrypted?: secret.txt
Encoded text has been written to encoded_secret.txt.
```

Three sample files (tale.txt, moby.txt, and rachel.txt) have been included in the project folder for testing. Encrypt() must be written before you can write decrypt().

Sample:
Encrypting **tale.txt** with the key **HELLO**
IT WAS THE BEST OF TIMES, IT WAS THE WORST OF TIMES, IT WAS THE AGE OF WISDOM, IT WAS THE AGE OF FOOLISHNESS, IT WAS THE EPOCH OF BELIEF, IT WAS THE EPOCH OF INCREDULITY, IT WAS THE SEASON OF LIGHT, IT WAS THE SEASON OF DARKNESS, IT WAS THE SPRING OF HOPE, IT WAS THE WINTER OF DESPAIR, WE HAD EVERYTHING BEFORE US, WE HAD NOTHING BEFORE US, WE WERE ALL GOING DIRECT TO HEAVEN, WE WERE ALL GOING DIRECT THE OTHER WAY – IN SHORT, THE PERIOD WAS SO FAR LIKE THE PRESENT PERIOD, THAT SOME OF ITS NOISIEST AUTHORITIES INSISTED ON ITS BEING RECEIVED, FOR GOOD OR FOR EVIL, IN THE SUPERLATIVE DEGREE OF COMPARISON ONLY.

Encrypted

PX HLG ALP MSZX ZQ HPQPD, WA ALD HOI HZFZX ZQ HPQPD, WA ALD HOI LRS VJ HTGKSX, TH DED EVL ERP CM JZZZPWSYSZW, TE KHW ESS LTZNV VJ MPZPIQ, TH DED EVL IAZQO SQ TBJVPOISMEJ, WA ALD HOI DPOZSY ZT SMRSH, PX HLG ALP DSHWZY CM HLCYUIDD, WA ALD HOI DAFPRR ZT OSAP, WA ALD HOI HTBAIC ZT KIDAOPV, HP VHH PGSYCESWUK MPTVVP FG, DI SLR USESWUK MPTVVP FG, DI HPFL EWW UVMYR RPVPNH AS SPOCIY, HS DICP OSP RZWUK OTFLGE EVL SESSY ALJ – WU WSZFA, XSP DLVTZR DED DC MEC WWRI ESS WVPDSUX APFPSO, EVHX DZAL SQ THZ RZTGPIDE OBXSZFPXTPG PRDTGAIO ZB PXD MSPRR CSJITGSK, JZC UVSO ZF MSC PJPP, TY HOI DFDLVWLHPZP OSNVPP CM GZXDHVTDCU SYWM.

decrypt()

This function will ask the user for a file to decrypt and the key to decrypt it. It should work in a similar manner to the encrypt function. The decrypted file should be written to a new file called "decoded_PREVIOUSFILENAME.txt".

Your output should look something like this:

```
DECRYPT MESSAGE
What is the key you'd like to use?: KEY
What is the name of the file to be decrypted?: encoded_secret.txt
Decoded text has been written to decoded_secret.txt.
```

You can test decrypt() by calling it on your encrypted sample files.

print_menu()

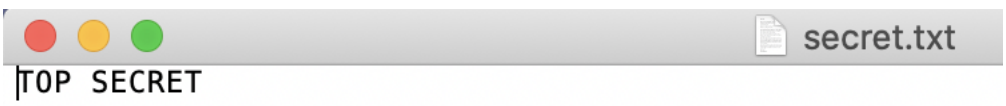
This function should run when the program starts and give the user 2 options: 1. Encrypt a file or 2. Decrypt a file. If the user types 1, the encrypt function should be called, if they type 2 decrypt should be called. Assume the user will enter a valid option.

TIPS:

- The starter file has imported "ascii_letters". You can check if a character is a valid letter by checking if it is **in ascii_letters**.
- You can use **.index()** on a list to get the index of a specific character.

EXAMPLE RUN:

secret.txt

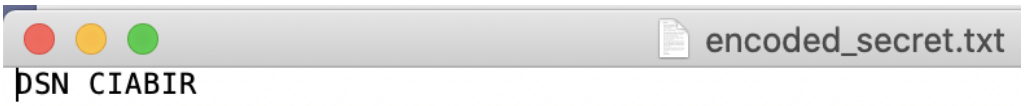


message_encryptor.py

```
Welcome to the message encryptor.
MENU
1. Encrypt message
2. Decrypt message
What would you like to do? Please enter the option number: 1

ENCRYPT MESSAGE
What is the key you'd like to use?: KEY
What is the name of the file to be encrypted?: secret.txt
Encoded text has been written to encoded_secret.txt.
```

encoded_secret.txt



message_encryptor.py

```
Welcome to the message encryptor.
MENU
1. Encrypt message
2. Decrypt message
What would you like to do? Please enter the option number: 2

DECRYPT MESSAGE
What is the key you'd like to use?: KEY
What is the name of the file to be decrypted?: encoded_secret.txt
Decoded text has been written to decoded_secret.txt.
```

decoded_secret.txt