CS 111 – Introduction to Computer Science – Spring 2018

Programming Assignment #4

Encryption and Decryption

Due Date: at 11:59pm on Saturday, April 21.

For this assignment, you will design and implement a program that requires the use of string and file processing.

Remember that assignments are to be done individually, you should not share code, share design ideas, or look at anyone's code. If you need help, you may use the tutor, TA, or professor. If you have questions about this policy, refer to the syllabus or ask the professor.

Before getting started with the project, create a folder named `proj4` within your `U:\cs111` folder and use it to save your source file for this project.

Background

The Caesar cipher, named after the dictator and war general of the Roman empire Julius Caesar, is an encryption technique once used to easily obscure military messages. The method is also known as a shift cipher, because it shifts the alphabet a number of steps. For instance, Caesar was rumored to use a shift of 3, so A would become D, B would become E, C would become F, etc. In order to decrypt the message, you simply shift the alphabet the same number of steps in the opposite direction.

encryption shift of 3	decryption shift of 3
`ABCDEFGHIJKLMNOPQRSTUVWXYZ`	`ABCDEFGHIJKLMNOPQRSTUVWXYZ`
`DEFGHIJKLMNOPQRSTUVWXYZABC`	`XYZABCDEFGHIJKLMNOPQRSTUVW`

Example:

Original Text:	`Hello World.`
Encrypt, shift 3:	`Khoor Zruog.`
Decrypt, shift 3:	`Hello World.`

The Caesar cipher was broken hundreds of years ago and can no longer be used for security, but it remains an important technique in the study of cryptography.

Julius Caesar

Program Description

You will implement a program `encryption.py` containing several functions that, together, can read a file, encrypt the text, and write the encrypted text into another file. The program will be able to do the same with decryption.

Your program will include functions to encrypt and decrypt characters, digits, entire strings, and entire files. Code the functions in the order given, and you will be able to use earlier functions to help the later functions.

There are simple formulas to encrypt and decrypt a number. For encryption, the general formula is

$$e = (x + s)\%n$$

where x is the starting number, s is the shift, and n is the number of options (10 for digits, 26 for characters). For decryption, the general formula is

$$d = (x - s)\%n$$

with the same pieces as encryption.

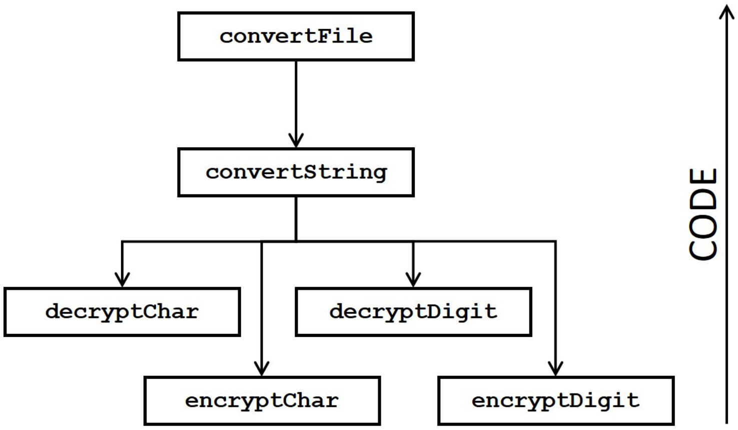
You will be able to encrypt digits directly, but for letters, it is easier to use a string containing all the letters and encrypt the index of the letter. For instance, if your letter is stored at index 8, and the shift is 7, the encrypted letter is stored at index 15. A constant containing all the (lowercase) letters has already been created for you.

Finish the following functions:

- `encryptDigit(digit, shift)` - Encrypt the digit by the shift. Use the general encryption formula and return the encrypted digit.
- `decryptDigit(digit, shift)` - Decrypt the digit by the shift. Use the general decryption formula and return the decrypted digit.
- `encryptChar(char, shift)` - Encrypt the character by the shift. If the character is lowercase, encrypt to a lowercase character, and if the character is uppercase, encrypt to an uppercase character. Use the general encryption formula on the index of the character and return the character at the encrypted index. Note that the number of possible characters is different than the number of possible digits.
- `decryptChar(char, shift)` - Decrypt the character by the shift. If the character is lowercase, decrypt to a lowercase character, and if the character is uppercase, decrypt to an uppercase character. Use the general decryption formula on the index of the character and return the character at the decrypted index.
- `convertString(string, shift, encrypt)` - Encrypts the entire string by the shift if `encrypt` is true, and decrypts the entire string by the shift if `encrypt` is false. Returns a new string containing the encrypted or decrypted string. Characters that are letters or digits should be encrypted or decrypted appropriately, and any other character should be kept the same.

- `convertFile(inputName, outputName, shift, encrypt` - Encrypts everything in the file named `inputName` by the shift and writes the results into the file named `outputName`, if `encrypt` is true. Otherwise, it decrypts everything in the file named `inputName` by the shift and writes the results into the file named `outputName`. Does not return anything.

The top-down design of this project is given below. Don't forget to code bottom-up!! Use `testEncryption.py` to test your functions as you code. The file `tisket.txt` is required to test the `convertFile` function.



Program Requirements

Your program must be well structured and meet the following specifications:

- Stick to using variables instead of literals as much as possible. On that note...
- Use constants if appropriate.
- Your prompts and output should look identical to those provided in this document.
- Your program must be commented appropriately, specifically you must:
  - Include an appropriate file prolog at the top of the source file.
  - Include a comment above each function including a description of the parameters, function, and return statement.
  - Include appropriate comments throughout the program.
  - Use meaningful variable names.
- Your program should not include a main function or any floating code.

What to Submit

Please submit the following file to Canvas by the due date and time.

- `encryption.py`

Remember, all of the files must be named exactly as indicated above, with the same case and with no spaces or special characters.