COSC 175 Dierbach
Spring 2011

PROGRAMMING ASSIGNMENT

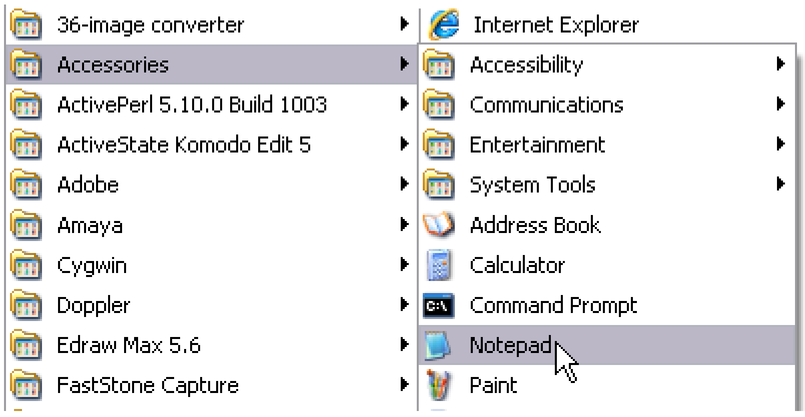
Encrypted Message Program

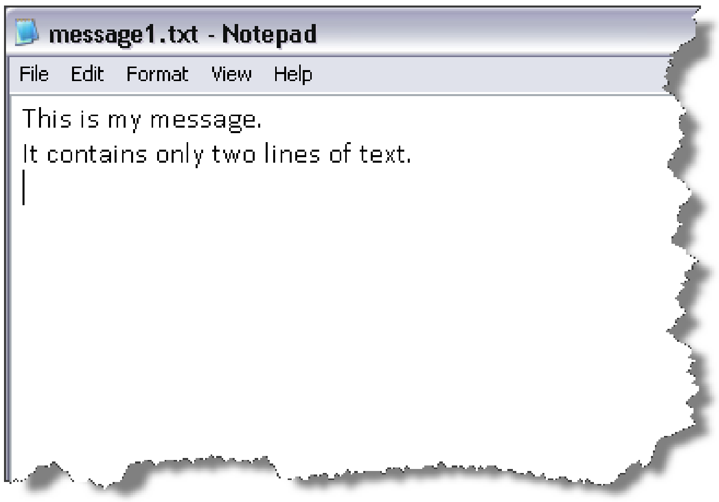
Due Tuesday, May 17th

For this programming assignment, you are to develop a Python program that can both encrypt a message (with a corresponding encryption key) as well as decrypt messages (given an encrypted message and the corresponding encryption key).

The Creation of Message Files

This program should be written to encrypt any given text file with file extension .txt (e.g., message1.txt) and produce a corresponding encrypted message with the _coded appended to the original file name (e.g., message1_coded.txt). The original text message is to be created using the notepad simple text editor provided in Windows (or a similar text editor provided on Macs).





```
This is my message.
It contains only two lines of text.
```

The Reading and Writing of Message Files

Both the encryption and decryption of message text files involve the same general process. The process for the encryption of a given file is given below:

```
inFile = open(filename, 'r')                      e.g., filename = 'message1.txt'
outFile = open(filename + '_coded', 'w')

line = inFile.readline()
while line != '':
    encrypt letter characters in string line
    outFile.write(line)
    line = inFile.readline()

outFile.close()
```

The specific process of the decryption of a given file is given below:

```
inFile = open(filename, 'r')                      e.g., filename = 'message1_coded.txt'
outFile = open(filename.rstrip('_coded'), 'w')

line = inFile.readline()
while line != '':
    decrypt letter characters in string line
    outFile.write(line)
    line = inFile.readline()

outFile.close()
```

Note that **rstrip** is used to remove a given trailing set of characters of a given string. That is, for any given string *str*, *str*.rstrip(*chars*) will remove the trailing chars *chars*.

The Encryption/Description of a Line of Text

In order to encrypt a given line of text, a simple key will be used, as shown below.

A	Z
B	Y
C	X
D	W
E	V
F	U
G	T
H	S
I	R

J	Q
K	P
L	O
M	N
N	M
O	L
P	K
Q	J
R	I

S	H
T	G
U	F
V	E
W	D
X	C
Y	B
Z	A

By this key, all occurrences of the letter 'A' are replaced with the letter 'Z' in the encrypted message, all occurrences of the letter 'B' are replaced with the letter 'Y', etc.

String in Python are an immutable type. Thus, like tuples, they cannot be altered, and therefore we cannot directly substitute one letter with another in a given string. Instead, we can create a new string with from the original string containing the desired substitution as follows:

line.replace(*currentChar*, *newChar*)

Thus, if variable line contained the string 'Book', then the following use of replace would created the new string 'Look',

```
line = 'Book'
line.replace('B', 'L')
```

It is important to realize that the replace command does NOT change the value in variable line. Thus, after the above is executed, line still contains 'Book'. Therefore, if we want to change variable line, we must do the following,

```
line = line.replace('B', 'L')
```

To decrypt a message would involve the reverse use of the key. Since when decrypting a message we receive both the message and the key as (separate) emailed files from another person, the program must open each of the files to descript the message.

The Generation of Random Keys

Each time a message is encrypted, it should be encrypted with a new key. Otherwise, if we used the same key to encrypt every message, if someone got a hold of the key, they would be able decrypt every encrypted sent.

First, a key will be represented as a list of each letter in the alphabet, in some random order as shown below,

```
['D', 'H', 'A', 'F', 'R', …]
```

Thus, the fact that 'D' is in the first position of the list indicates that all occurrence of letter 'A' should be replaced by the letter 'D' in encrypted messages, all occurrences of 'B' should be replaced with the letter 'H', etc.

To generate a random key, we can make use of the randint function of the random module,

```
import random

random.randint(1,10)
```

The import statement should only appear once at the start of the program. Function randint(j,k) will generate a random number between j and k inclusive each time called. So we start with a list of all letters of the alphabet, we can created a random key as follows,

```
letters = [‘A’, ‘B’, ‘C’, ‘D’, ‘E’, ‘F’, …]

key = []
```

We then call random.randint twenty-six times, once for each letter of the alphabet. The first time, we would call

```
whichOne = random.randint(0,25)
```

This will give a random number in the range 0-25, since the letters list is indexed 0-25. We then append to the currently empty key list the letter from the letters list as given by whichOne,

```
char = letters[whichOne]

key.append(char)
```

Let’s say that whichOne was 2, and therefore the selected character from letters was ‘C’. Since we do not want to add letter ‘C’ a second time to the key, we delete that letter from the letters list so that it cannot be selected again. The remove command can be used for this,

```
letters.remove(char)
```

assuming that char contains ‘C’. Letters would now contain the following,

```
letters = [‘A’, ‘B’, ‘D’, ‘E’, ‘F’, …]     ,   letter ‘C’ removed from list
```

Note the we do NOT use letters = letters.remove(char) here.

Now to get the next random letter from list letters, we need to generate a random number in the range of 0-24, not 0-25 as before, since the letters list contains only 25 letters at this point. Thus, it would make sense here to generate a random number in the range related to the *current* length of the list,

```
whichOne = random.randint(0, len(letters) – 1)
```

Designing and Implementing the Program

You are to use a top-down approach to the design of your program. Thus, you will first create the “broad outline” of the program in the main section of the program, and defer other details to a set of functions that you plan to develop later. Thus, for example, the main section of your program may begin as follows,

```
# MAIN

# variable initialization
( to be later completed )

# program welcome
programWelcome()

# prompt user for encryption or decryption
selection = raw_input('Do you wish to <E>ncrypt or <D>ecrypt a message file: ')
( Note: modify this so that it re-prompts the user of E or D not entered )

if selection == 'E':    # file encryption
    fileName = getFileName()
    inFile = open(fileName, 'r')
    ( use the code from the image manipulation program to re-prompt the use for the file name
      if the file cannot be found )
    outFile = open(fileName + '_coded', 'w')

    line = inFile.readline()
    while line != '':
        line = ( encrypted line )
        outfile.write(line)
        line = inFile.readline()

        codedFileName = (fileName with '_coded' appended)

    inFile.close()
    outFile.close()


else:                   # file decryption

    ( to complete )
```

WHAT TO SUBMIT

- A copy of your program submitted to Blackboard
- A printout of a message file of at least five lines, a copy of the corresponding key, and a copy of the resulting encrypted message file submitted to Blackboard

PROGRAMS MUST BE TURNED IN AT THE START OF CLASS TIME TUESDAY, May 17th.
LATE PROGRAMS WILL BE MARKED DOWN 10% PER DAY.