**Mark Eatough**

**CSIS 2430 9:00 Class**

**Programming Project 6**

**Caesar Cipher Program**

**Assignment objective:**

Implement a program that will take ANY such formula for the Caesar Cipher.  You will use/need this for the first Midterm exam.

**What Worked?:**

I used the list index to associate values with the letters idea again for the Caesar Cipher just like I did for the RSA encryption . This seemed to work well again for this program. I then called a method that converted any letter in the string to the corresponding letter index, or it ignores the character if it is not a letter. The program then increments the letter by the user entered offset.

**What did not work?:**

I had some trouble getting the letter to corresponding index method to work. I tried to write a statement that would have been the equivalent of if character is not a letter to help with the problem but I could not get that to work. My solution was finally to return the corresponding index if the character was a letter, and to return the character if the character was not a letter.

**Comments:**

This program was very similar to the RSA encryption program, I was able to use some of the same code, and then alter it for the Caesar Cipher specifications. I learned something new about python that I think will be very useful in certain situations; the same method can return two different types. For example one of my methods returns an integer value if certain conditions exist, and a string value if other conditions exist. This would not be possible in any other language I have used, which includes Java, C++ and C# as these languages all require the programmer to state what their return type will be if there is one.

1 '''  
 2 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
 3 \* Discrete Structures  
 4 \* Caesar Cipher Program  
 5 \* Programmer: Mark Eatough  
 6 \* Course: CSIS 2430   
 7 \* Created October 6, 2013  
 8   
 9 \*This program takes user input for an offset, and user  
10 \*input for a phrase, and then changes each letter in   
11 \*the phrase to as many letters in the alphabet as is  
12 \*necessary. The output is the Caesar Ciphered phrase   
13 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
14 '''  
15   
16 #Create list of characters  
17 characters = []  
18 #populate list of characters  
19 a = ord('a')  
20 z = ord('z')  
21 for letter in range(a, z+1):  
22 characters.append(chr(letter))  
23 #method to convert the characters into their associated values  
24 #ie a = 0, b = 1 etc.  
25 def charToIntVal(c):  
26 for i in range(len(characters)):  
27 if(c.lower() == characters[i]):  
28 index = i  
29 return index   
30 return c  
31 #method that parses out the string, converts the letters to numbers,  
32 #increments those numbers by the given offset, and then returns the  
33 #corresponding letter  
34 def caesarCipher(offset, phrase):  
35 tempString = ""  
36 for i in range(len(phrase)):  
37 if(isinstance(charToIntVal(phrase[i]), (int))):  
38 tempInt = offset + charToIntVal(phrase[i])  
39 tempString += characters[tempInt%26]  
40 else:  
41 tempString += charToIntVal(phrase[i])  
42 return tempString  
43   
44 #prompt user to enter offset  
45 offset = input("Please enter your offset: ")  
46 #prompt user to enter phrase  
47 phrase = raw\_input("Please enter your phrase you would like to be caesar ciphered:\n")  
48   
49 #print out users original phrase  
50 print "\n\n\nYour original phrase is: \n\n", phrase  
51 #print out users phrase after going through the cipher  
52 print "\n\n\nYour caesar ciphered phrase is: \n\n", caesarCipher(offset, phrase)

