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?:

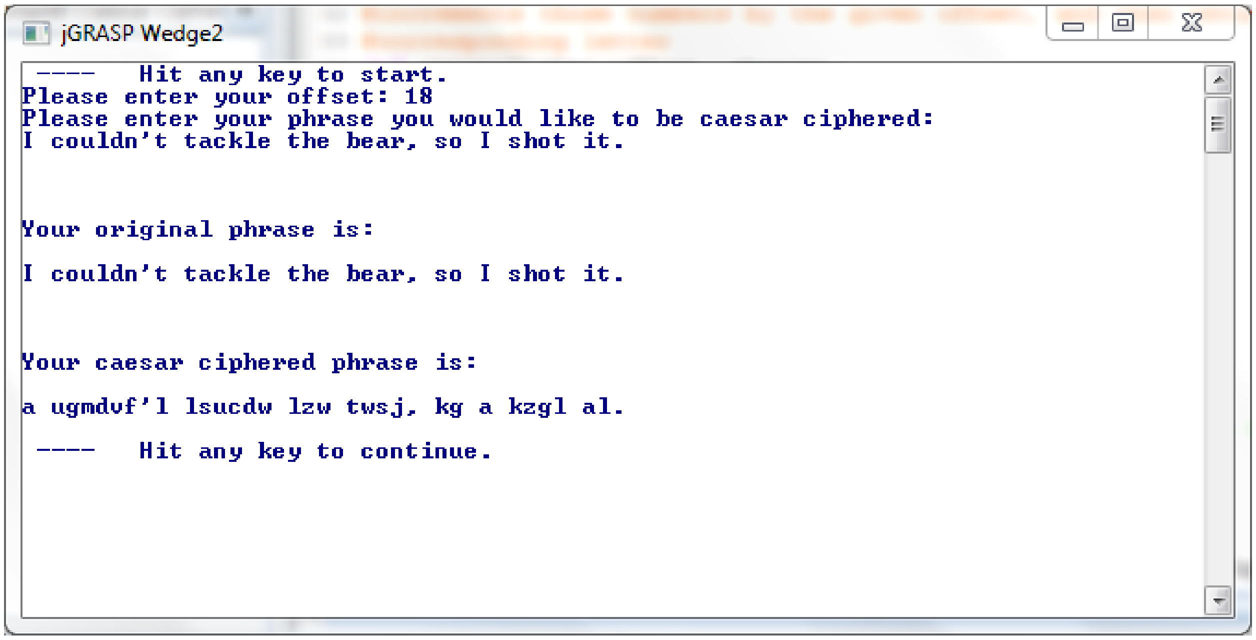
I had some trouble getting the letter to corresponding index method to work. I tried to write a statement that would have been the equivalent of if character is not a letter to help with the problem but I could not get that to work. My solution was finally to return the corresponding index if the character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

```python
'''
***********************************************************
 * Discrete Structures
 * Caesar Cipher Program
 * Programmer: Mark Eatough
 * Course: CSIS 2430
 * Created October 6, 2013

 *This program takes user input for an offset, and user
 *input for a phrase, and then changes each letter in
 *the phrase to as many letters in the alphabet as is
 *necessary.  The output is the Caesar Ciphered phrase
 ***********************************************************
'''

#Create list of characters
characters = []
#populate list of characters
a = ord('a')
z = ord('z')
for letter in range(a, z+1):
   characters.append(chr(letter))
#method to convert the characters into their associated values
#ie a = 0, b = 1 etc.
def charToIntVal(c):
   for i in range(len(characters)):
      if(c.lower() == characters[i]):
         index = i
         return index
   return c
#method that parses out the string, converts the letters to numbers,
#increments those numbers by the given offset, and then returns the
#corresponding letter
def caesarCipher(offset, phrase):
   tempString = ""
   for i in range(len(phrase)):
      if(isinstance(charToIntVal(phrase[i]), (int))):
         tempInt = offset + charToIntVal(phrase[i])
         tempString += characters[tempInt%26]
      else:
         tempString += charToIntVal(phrase[i])
   return tempString

#prompt user to enter offset
```

```
offset = input("Please enter your offset: ")
#prompt user to enter phrase
phrase = raw_input("Please enter your phrase you would like
to be caesar ciphered:\n")

#print out users original phrase
print "\n\n\nYour original phrase is: \n\n", phrase
#print out users phrase after going through the cipher
print "\n\n\nYour caesar ciphered phrase is: \n\n",
caesarCipher(offset, phrase)
```


jGRASP Wedge2

```
 ----   Hit any key to start.
Please enter your offset: 18
Please enter your phrase you would like to be caesar ciphered:
I couldn't tackle the bear, so I shot it.



Your original phrase is:

I couldn't tackle the bear, so I shot it.



Your caesar ciphered phrase is:

a ugmdvf'l lsucdw lzw twsj, kg a kzgl al.

 ----   Hit any key to continue.
```