Hail Caesar! | Problem Set 6 | Contenu du cours 6.00.1x

Hail Caesar!

Encryption is the process of obscuring information to make it unreadable without special knowledge. For centuries, people have devised schemes to encrypt messages - some better than others - but the advent of the computer and the Internet revolutionized the field. These days, it's hard not to encounter some sort of encryption, whether you are buying something online or logging into a shared computer system. Encryption lets you share information with other trusted people, without fear of disclosure.

A cipher is an algorithm for performing *encryption* (and the reverse, *decryption*). The original information is called *plaintext*. After it is encrypted, it is called *ciphertext*. The ciphertext message contains all the information of the plaintext message, but it is not in a format readable by a human or computer without the proper mechanism to decrypt it; it should resemble random gibberish to those for whom it is not intended.

A cipher usually depends on a piece of auxiliary information, called a *key*. The key is incorporated into the encryption process; the same plaintext encrypted with two different keys should have two different ciphertexts. Without the key, it should be difficult to decrypt the resulting ciphertext into readable plaintext.

This assignment will deal with a well-known (though not very secure) encryption method called the Caesar cipher. In this problem set you will need to devise your own algorithms and will practice using recursion to solve a non-trivial problem.

Caesar Cipher

The basic idea of the Caesar cipher is that you pick an integer for a key, and shift every letter of your message by the key. For example, if your message was "happy" and your key was 3, "h" becomes "k", "a" becomes "d", and so on, because we are shifting every letter three spots to the right. Here is what the whole alphabet looks like shifted three spots to the right:

```
Original:  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
 3-shift:  d e f g h i j k l m n o p q r s t u v w x y z a b c
```

Using the above key, we can quickly translate the message "happy" to "kdssb" (note how the 3-shifted alphabet wraps around at the end, so x -> a, y -> b, and z -> c).

Note!! We are using the English alphabet for this problem - that is, the following letters in the following order:

```
>>> import string
>>> print string.ascii_lowercase
abcdefghijklmnopqrstuvwxyz
```

In this problem, we will use a variant of the standard Caesar cipher where we will treat upper and lower case letters separately, so upper case letters will always be mapped to upper case letters, and lower case letters will always be mapped to lower case letters. Thus, if "a" maps to "c", "A" will map to "C". Characters such as the space character, commas, periods, exclamation points, etc will *not* be encrypted by this cipher - basically, all the characters within `string.punctuation`, plus the space (`' '`) and all numerical characters (0 - 9).

To learn more about the Caesar cipher, check out this Wikipedia article.

Wrapper Functions

Now that you are ready to start coding, you can look carefully at the code skeletons that we have provided for you. Do not be intimidated by the length of the function specifications we provide with the supplied code! Many of these problems rely on **wrapper functions**, an extremely useful coding concept that, when implemented correctly, often requires very little additional code. The idea of wrapper functions here is that the functions visible to a user take as arguments simple inputs, and then supply these arguments - plus other information - to functions visible only within the implementation to perform the computation.

Read the Wikipedia article on wrapper functions for more information.