## CS101. A10a - Fixed Substitution

Write a function to return the 5th english letter after the input.

**Tasks to do**
Implement the function f_substitute(letter)
letter (string): single letter from the english alphabet.
Return value: 5th letter from the input

**Examples**
- f_substitute(‘a’) returns ‘f’
- f_substitute(‘f’) returns ‘k’
- f_substitute(‘u’) returns ‘z’
- f_substitute(‘z’) returns ‘e’

**Code Template (f_substitute.py)**

```
'''
Fixed Substitute
'''

def f_substitute(letter):
     pass
```

**Test Cases**

| Function | Inputs | Output | Remarks |
|---|---|---|---|
| f_substitute | ‘a’ | ‘f’ | |
| f_substitute | ‘f’ | ‘k’ | |
| f_substitute | ‘u’ | ‘z’ | |
| f_substitute | ‘z’ | ‘e’ | |

********

## CS101. A10b - Text Substitution

Write a function to return the nth english letter after the input.

**Tasks to do**
Implement the function substitute(letter, n)
letter (string): single letter from the english alphabet.
n (int): integer (can be negative)
Return value: nth letter from the input

**Examples**
- substitute(‘a’ , 0) returns ‘a’
- substitute(‘a’ , 1) returns ‘b’

- substitute(‘a’ , 25) returns ‘z’
- substitute(‘a’ , 26) returns ‘a’
- substitute(‘a’ , -1) returns ‘z’
- substitute(‘a’ , -25) returns ‘b’
- substitute(‘a’ , -26) returns ‘a’

**Code Template (substitute.py)**

```
'''
Substitute letter
'''

def substitute(letter, n):
     pass
```

**Test Cases**

| Function | Inputs | Output | Remarks |
|---|---|---|---|
| substitute | ‘a’ , 1 | ‘b’ | |
| substitute | ‘a’ 25 | ‘z’ | |
| substitute | ‘a’ , 26 | ‘a’ | |
| substitute | ‘a’ , -1 | ‘z’ | |
| substitute | ‘a’ , -25 | ‘b’ | |
| substitute | ‘a’ , -26 | ‘a’ | |
| substitute | ‘z’ , 1 | ‘a’ | |
| substitute | ‘z’ 25 | ‘y’ | |
| substitute | ‘z’ , 26 | ‘z’ | |
| substitute | ‘z’ , -1 | ‘y’ | |
| substitute | ‘z’ , -25 | ‘a’ | |
| decrypt | ‘z’ , -26 | ‘z’ | |

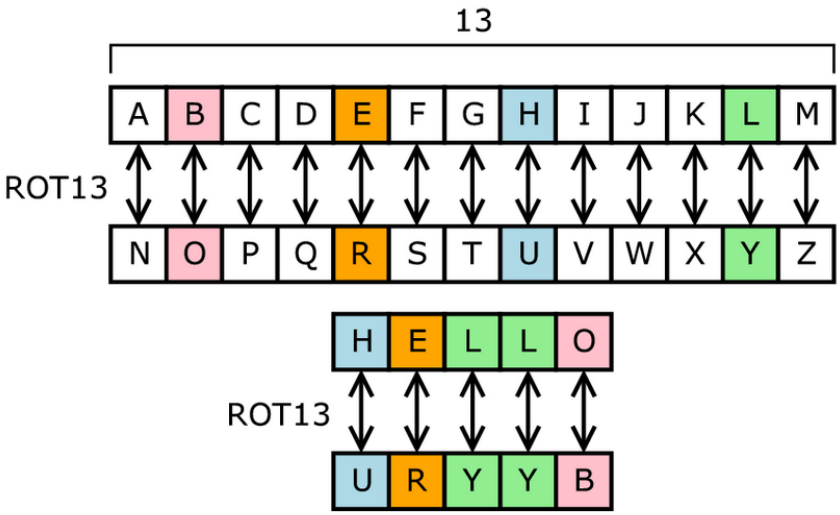
********

CS101. A10c - Encryption, Decryption through Substitution Cipher

Ref: http://en.wikipedia.org/wiki/Substitution_cipher

Encryption converts a string in plain english (known as *plaintext*), into another form known as *ciphertext*. Typically, *plaintext* is in plain readable English. The *ciphertext* is unintelligible and cannot be read without first decrypting it.

**Substitution cipher** is a method of encrypting in which units of plaintext are replaced with the ciphertext, in a well defined manner. Substitution can be done using a **key**. The encrypted message (*ciphertext*) can be read by the receiver after performing the inverse substitution process. The inverse substitution process is the decryption process here. Example process is shown in the picture below.




In the above example, plaintext is “HELLO” and the ciphertext is “URYYB”. The key here is ROT13. Details about ROT13 are below. The receiver receives “URYYB” and the key. Using the key, the receiver can decrypt the message and obtain the original message “HELLO”.

**The Key**
The ROT13 key can be written as “nnnnnnnnnnnnn”. Each position in the Key is the letter to be substituted with the character ‘a’ in that position. Examples.
Key is “aaaaa”. “hello” ==> “hello”
Key is “ddddd”. “hello” ==> “khoor”
Key is “nnnnn”. “hello” ==> “uryyb”
Key is “abcde”. “hello” ==> “hfnos”
Key is “abc”. “hello” ==> “hfnlp”. Key should be interpreted as “abcabcabc…” (long enough to encrypt the plaintext)

**Tasks in the Assignment**

- Implement two functions to achieve Substitution Cipher
    - encrypt() and decrypt()
- encrypt(plaintext, key)
    - Creates the ciphertext using key
    - Returns the ciphertext
- decrypt(ciphertext, key)
    - Reverses the ciphertext into the plaintext using key
    - Returns the plaintext

- Assume that the inputs do not have any characters apart from a-z (no upper case, no numerals, no punctuation marks)

**Examples**

- encrypt(plaintext, key="ddddddddddddddddddddd"):
  - plaintext: plain text to be encrypted
  - Key: key to use to encrypt. Keep default as ROT3.
- encrypt() implements Simple Shift Cipher using the key (Eg. Caesar Cipher)
- encrypt("dontlookup") should return "grqworrnxs"
- encrypt("dontlookup", "abcdefghij") should return "dppwpturcy"
- decrypt(ciphertext, key="ddddddddddddddddddddd"):
  - ciphertext: cipher text to be decrypted to plaintext
  - Key: key to use to encrypt. Default is ROT3.
- decrypt() implements Simple Shift Cipher using the key (Eg. Caesar Cipher)
- decrypt("grqworrnxs") should return "dontlookup"
- decrypt("dppwpturcy", "abcdefghij") should return "dontlookup"

**Code Template (encrypt_decrypt.py)**

```
'''
Substitution Cipher
Encrypt / Decrypt
'''

def encrypt(text, key="ddddddddddddddddddddd"):
     pass

def decrypt(text, key="ddddddddddddddddddddd"):
     pass
```

**Sample output**

```
>>> text="jurassicpark"
>>>key="ankylosaur"
>>>print(encrypt(text,key))
jhbydgacjrrx
>>>print("jhbydgacjrrx",key))
jurassicpark
```

**Test Cases**

| Function | Inputs | Output | Remarks |
|---|---|---|---|
| encrypt | text="aaaaaaaaaa"<br>key="aaaaaaaaaa" | aaaaaaaaaa | Empty String |
| decrypt | text="aaaaaaaaaa"<br>key="aaaaaaaaaa" | aaaaaaaaaa | |
| encrypt | text="jurassicpark"<br>key="ankylosaur" | jhbydgacjrrx | |
| decrypt | text="jhbydgacjrrx"<br>key="ankylosaur" | jurassicpark | |

| | | | |
|---|---|---|---|
| encrypt | text="abcdefghij"<br>key="abcdefghij" | acegikmoqs | |
| decrypt | text="acegikmoqs"<br>key="abcdefghij" | abcdefghij | |
| encrypt | text="aaaaaaaaaa"<br>key="abcdefghij" | abcdefghij | |
| decrypt | text="abcdefghij"<br>key="abcdefghij" | aaaaaaaaaa | |
| encrypt | text="zzzzzzzzzz"<br>key="abcdefghij" | zabcdefghi | |
| decrypt | text="zabcdefghi"<br>key="abcdefghij" | zzzzzzzzzz | |
| encrypt | text="rainbowcolors"<br>key="vibgyor" | mijtzcnxwmupg | |
| decrypt | text="mijtzcnxwmupg"<br>key="vibgyor" | vibgyor | |
| encrypt | text="panthera"<br>key="leo" | aebelsce | |
| decrypt | text="aebelsce"<br>key="leo" | panthera | |
| encrypt | “hello”, “abc” | hfnlp | |
| decrypt | “hfnlp”, “abc” | hello | |
| encrypt | text=* (any string)<br>key="" | raise ValueError(“Key cannot be empty.”) | |
| decrypt | text=* (any string)<br>key="" | raise ValueError(“Key cannot be empty.”) | |

********