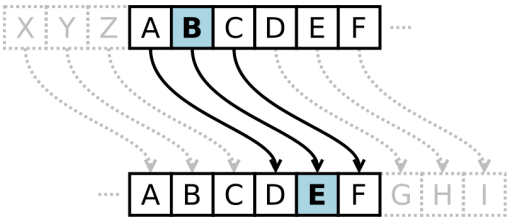
Programming Project 05

This assignment is worth 45 points (4.5% of the course grade) and must be **completed and turned in before 11:59 on Monday, February 20, 2017.** This assignment will give you more experience on the use of strings and functions.

Assignment Overview

The Caesar cipher is named after Julius Caesar who used this type of encryption to keep his military communications secret. A Caesar cipher replaces each plain-text letter with one that is a fixed number of places down the alphabet. The plain-text is your original message; the cipher-text is the encrypted message. The example shown below is a shift of three so that "B" in the plain-text becomes "E" in the cipher-text, a "C" becomes "E", and so on. The mapping wraps around so that "X" maps to "A" and so on.



Here is the complete mapping for a shift of three:

```
Plain:    ABCDEFGHIJKLMNOPQRSTUVWXYZ
Cipher:   DEFGHIJKLMNOPQRSTUVWXYZABC
```

To encrypt a message simply substitute the plain-text letters with the corresponding cipher-text letter. For example, here is an encryption of "the quick brown fox jumps over the lazy dog" using our shift-three cipher (case is ignored):

```
Plaintext:    the quick brown fox jumps over the lazy dog
Ciphertext: WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ
```

To decrypt the message simply reverse the process.

The encryption can also be represented using modular arithmetic after first transforming the letters into numbers according to the scheme: A = 0, B = 1, etc. (which is the index of the alphabet if it is in a string or a list). A shift-three cipher will take the number of each letter (`plainTextChar`), add 3 (the shift), and then find the remainder after dividing by 26 to get the cipherText:

```
cipherTextChar = (plainTextChar + 3) % 26
```

Program Specifications

Your task is to write a program that can decrypt a message that has been encoded using a Caesar cipher. Stated another way, you need to find the "shift" for the cipher. Once you determine the shift, you know the mapping so you can decrypt the message.

Caesar Cipher Cracking

To find the "shift" you need to know about cracking Caesar ciphers using a technique that has been around for over a thousand years. Any language such as English has a known distribution for each letter. For example, the letter "E" is the most common letter in English making up 12.702% of the letters on average (ignoring case). The letter "T" is next (9.056%), followed by "A" (8.17%), and so on (for all see Wikipedia). The order "E"-"T"-"A"-… is what matters for decryption, not the percentage.

The procedure begins by finding the most common letter. You can guess that the most common letter maps to "E." You can now find the "shift" from the most common letter in the cipher-text to the expected most common letter "E". For example, if the most common letter in the cipher-text is "H", you know that the shift from "E" to "H" is 3.

What about spaces between words and punctuation? In real world cipher-text there are no spaces or punctuation because those are useful clues for deciphering. Similarly, case can provide clues so case doesn't matter. In the cipher-text we provide for this project we have left in both the punctuation and spaces because they will be helpful for you to recognize that your deciphering is correct or not. You will need to ignore spaces and punctuation when counting letters in the cipher-text (if you forget to ignore them, beware that the space will be the most common character).

Your high level algorithm will be:

1. Input the cipher-text.
2. Find the most common character.
3. Find the shift from "E" to that most common character.
4. Use the shift to decode each character of the cipher-text and print the resulting plain-text.
5. (Hint: stop here for a simple version to start with.)
6. If the plain-text isn't readable English, try the next-most common character and continue checking successive next-most-common characters until the resulting plain-text is readable.

Your program must also meet the following specifications:

1. You must have at least these four functions—more are fine. A `proj05.py` file with function stubs is provided.
    a. `def get_char(ch,shift)` The parameters are `ch`, a string, and `shift`, an int. This function returns the shifted decoding of character `ch`, i.e. decode ch. If the shift is -3 and the cipher-text character is "H", this function will return the plain-text character "E"
    b. `def get_shift(s,ignore)` The parameters `s` and `ignore` are both strings. Return the shift (key) for string s, and the most common character (if there is more than one, return any). The most common character should be "E" so `shift` is the shift to get "E". For example, if the most common character in the cipher-text is "H", the shift will be -3, the number of characters to get to "E". In that case you would return -3 and "H". The `ignore` parameter is the string of cipher-text letters to ignore—they have been tried but didn't result in readable English text. The `ignore` parameter allows you to find the next-most-common character by ignoring more common characters. The simple starting point is to ignore the `ignore` parameter, i.e. only find the most common character.
    c. `def output_plaintext(s,shift)` The parameters are `s`, a string, which is the cipher-text and `shift`, an int. Output plain-text of the cipher-text using the key, i.e. using the shift. Print the output in upper case.

d. `def main()` The main program that calls the other functions—don't forget the call to main, i.e. `main()`. Having `main` allows us to more easily create a set of test programs to test the other functions because we need to comment out `main` for those tests.

Deliverables
The deliverable for this assignment is the following file:

`proj05.py` -- your source code solution

Be sure to use the specified file name and to submit it for grading via the **handin system** before the project deadline

Notes and Hints:
1. To clarify the project specifications, sample output is appended to the end of this document.
2. Items 1-9 of the Coding Standard will be enforced for this project—note the change to include more items.
3. You can test functions separately—that can be a huge advantage in developing correct code faster! If you cannot figure out how to do that, ask your TA for guidance.
4. I didn't use them, but one way to decrypt is to use the two functions that are inverses of each other: `ord()` and `chr()`. These work because our letters are based on the ASCII subset of Unicode.
5. You do not need to check for any input errors.
6. You may not use advanced data structures such as lists, dictionaries, sets or classes in solving this problem.

Sample Interaction:
(Of course, our test cases come from Shakespeare's *Julius Caesar*!)
Test Case 1

```
Cracking a Caesar cypher.


Input cipherText: Frzdugv glh pdqb wlphv ehiruh wkhlu ghdwkv; Wkh
ydoldqw qhyhu wdvwh ri ghdwk exw rqfh. Ri doo wkh zrqghuv wkdw L bhw
kdyh khdug, Lw vhhpv wr ph prvw vwudqjh wkdw phq vkrxog ihdu; Vhhlqj
wkdw ghdwk, d qhfhvvdub hqg, Zloo frph zkhq lw zloo frph.

COWARDS DIE MANY TIMES BEFORE THEIR DEATHS; THE VALIANT NEVER TASTE
OF DEATH BUT ONCE. OF ALL THE WONDERS THAT I YET HAVE HEARD, IT
SEEMS TO ME MOST STRANGE THAT MEN SHOULD FEAR; SEEING THAT DEATH, A
NECESSARY END, WILL COME WHEN IT WILL COME.

Is the plaintext readable as English? (yes/no): yes
```

Test Case 2

```
Cracking a Caesar cypher.


Input cipherText: Wlv d frpprq surri, Wkdw orzolqhvv lv brxqj
dpelwlrq'v odgghu
```

```
JYI Q SECCED FHEEV, JXQJ BEMBYDUII YI OEKDW QCRYJYED'I BQTTUH

Is the plaintext readable as English? (yes/no): no
PEO W YKIIKJ LNKKB, PDWP HKSHEJAOO EO UKQJC WIXEPEKJ'O HWZZAN

Is the plaintext readable as English? (yes/no): no
FUE M OAYYAZ BDAAR, FTMF XAIXUZQEE UE KAGZS MYNUFUAZ'E XMPPQD

Is the plaintext readable as English? (yes/no): no
XMW E GSQQSR TVSSJ, XLEX PSAPMRIWW MW CSYRK EQFMXMSR'W PEHHIV

Is the plaintext readable as English? (yes/no): no
KZJ R TFDDFE GIFFW, KYRK CFNCZEVJJ ZJ PFLEX RDSZKZFE'J CRUUVI

Is the plaintext readable as English? (yes/no): no
ETD L NZXXZY ACZZQ, ESLE WZHWTYPDD TD JZFYR LXMTETZY'D WLOOPC

Is the plaintext readable as English? (yes/no): no
MBL T VHFFHG IKHHY, MATM EHPEBGXLL BL RHNGZ TFUBMBHG'L ETWWXK

Is the plaintext readable as English? (yes/no): no
LAK S UGEEGF HJGGX, LZSL DGODAFWKK AK QGMFY SETALAGF'K DSVVWJ

Is the plaintext readable as English? (yes/no): no
UJT B DPNNPO QSPPG, UIBU MPXMJOFTT JT ZPVOH BNCJUJPO'T MBEEFS

Is the plaintext readable as English? (yes/no): no
TIS A COMMON PROOF, THAT LOWLINESS IS YOUNG AMBITION'S LADDER

Is the plaintext readable as English? (yes/no): yes
```

Scoring Rubric

```
Computer Project #05                                Scoring Summary

General Requirements

______    7 pts    Coding Standard 1-9
        (descriptive comments, function header, etc...)


Implementation:

__0__    (12 pts)  Pass test1: correctly decodes a message that has "E"
                   as the most common character.


__0__    (14 pts)  All three functions work as specified.
                   (4 pt) get_char
                   (6 pt) get_shift
                   (4 pt) output_plaintext
```

```
__0__    (12 pts)  Pass test2: correctly decodes a message that does not
                   have "E" as the most common character by prompting the
                   user for successive tries.


TA Comments:
```

Optional Testing

This test suite has a test program for each of the three functions and two for the entire program. They are handled slightly differently.

1. Testing functions using `function_test1.py, function_test2.py, and function_test3.py` simply run these programs.

2. Testing the entire program using `run_file.py`.
   You need to have the files `test1.txt` and `test2.txt` from the project directory.
   Make sure that you have the following lines at the top of your program (only for testing):

```
import sys
def input( prompt=None ):
    if prompt != None:
        print( prompt, end="" )
    aaa_str = sys.stdin.readline()
    aaa_str = aaa_str.rstrip( "\n" )
    print( aaa_str )
    return aaa_str
```

Educational Research

When you have completed the project insert the 5-line comment specified below.

For each of the following statements, please respond with how much they apply to your experience completing the programming project, on the following scale:

1 = Strongly disagree / Not true of me at all
2
3
4 = Neither agree nor disagree / Somewhat true of me
5
6
7 = Strongly agree / Extremely true of me

****Please note that your responses to these questions will not affect your project grade, so please answer as honestly as possible.****

Q1: Upon completing the project, I felt proud/accomplished

Q2: While working on the project, I often felt frustrated/annoyed

Q3: While working on the project, I felt inadequate/stupid

Q4: Considering the difficulty of this course, the teacher, and my skills, I think I will do well in this course.

Q5: I ran the optional test cases (choose 7=Yes, 1=No)

Please insert your answers into the bottom of your project program as a comment, formatted exactly as follows (so we can write a program to extract them).

```
# Questions
# Q1: 5
# Q2: 3
# Q3: 4
# Q4: 6
# Q5: 7
```