**COMP130 HW8: String processing  
instructor: John MacCormick**

Question 1. (5 points) Consider the following code.

quote = 'Ifemelu liked the tranquil greenness of the many trees.'

s1 = quote[0]

s2 = quote[6]

s3 = quote[len(quote) - 6]

i = 10

s4 = quote[i + 2]

s5 = quote[:4]

s6 = quote[40:]

s7 = quote[10:13]

s8 = quote[44:len(quote)]

s9 = quote[6:6]

s10 = quote[-5]

What string is assigned to each of the variables s1 to s10?

Question 2. (5 points) Consider the statements given below:

print(str(letter))

while index > 0:

index = len(s)

s = "Philaldelphia had the musty scent of history."

index = index - 1

letter = s[index]

Reorder and add indentation to the above statements, producing a program that prints out every character in s in reverse order.

Question 2. (5 points) Consider the statements given below:

index = index + 1

index=0

if char=='A' or char=='E' or char=='I' or char =='O' or char=='U':

def get\_vowel(s):

return 'X'

char=s[index]

while index < len(s):

return char

Reorder the above statements (adding indentation) into a function that returns the first uppercase vowel that appears in the string s, or the letter X if s contains no vowels.

Question 3. (10 points) Write a function that returns the number of words that appear in a string. You may assume that there is a single space between any two words.

Question 4. (10 points) Write a test function, as described in our course, for your function from the previous question. Employ sufficient tests to have high confidence that the function is working correctly for all valid inputs.

Question 5. (5 points) Simplify the following program by using a method or methods from the str class.

choice = input('Enter Yes or No: ')

if choice == 'Yes' or choice == 'YES' or choice == 'yes':

print('You chose yes :)')

elif choice == 'No' or choice == 'NO' or choice == 'no':

print('You chose no :<')

Question 5. (5 points) Simplify the following program by using a method or methods from the str class.

line = input('Enter a line of text: ')

index=0

while line[index] == ' ':

index=index+1

line=line[index:]

print(line)

**Substitution Ciphers**

The remaining questions in this homework assignment use an encryption technique known as a *substitution cipher*. In a substitution cipher, there is an *alphabet* and a *key*:

alphabet **=**'ABCDEFGHIJKLMNOPQRSTUVWXYZ .?0123456789'

key **=**'DEFGHIJKLMNOPQRSTUVWXYZ .?0123456789ABC'

To encode a message, each letter in the message is translated into the corresponding letter in the key. Using the alphabet and key above, for example, CAT is encoded to FDW. The C is looked up in the alphabet and then matched to the F in the key, which is at the same location. Similarly for the A to D and T to W.

To decode a message, the process is reversed. For example, GRJ decodes to DOG. The G is looked up in the key and then matched to the D that appears at the same location in the alphabet. Similarly for the R to O and J to G.

Substitution ciphers such as the one shown above, where the *key* is a rotation of the alphabet, are known as *Caesar ciphers.* They were used by Julius Caesar to encode messages he sent to his military commanders. Due to their simplicity, Caesar ciphers are essentially useless today as an encryption technique. But they are still good for providing programming exercises.

In this series of questions, you'll develop some functions that can encode and decode messages using a Caesar cipher.

Question 6. (15 points) Write a function rotate(s, n) that when given a string s will rotate the characters in it left by n positions. As an example, using the alphabet from above, the call rotate(alphabet,3) would return the key. Values of n greater than the length of s should be prohibited by a guardian. You can use the test function below to check that your function is working correctly. Hint: Use string slices.

def test\_rotate():

assert rotate('ABCDEFG', 1) == 'BCDEFGA'

assert rotate('ABCDEFG', 2) == 'CDEFGAB'

assert rotate('ABCDEFG', 6) == 'GABCDEF'

assert rotate('ABCDEFG', 7) == 'ABCDEFG'

print('test\_rotate succeeded')

Question 7. (15 points) Write a function encode(s, alphabet, key) that encodes s using the given alphabet and key. You can write this function incrementally. Write it first assuming s is a single character. The test function below will test this case. Then modify that solution to handle the case where s contains multiple characters.

def test\_encode\_single\_char():

assert encode('C', alphabet, key) == 'F', 'C did not encode correctly'

assert encode('A', alphabet, key) == 'D', 'A did not encode correctly'

assert encode('T', alphabet, key) == 'W', 'T did not encode correctly'

assert encode(' ', alphabet, key) == '0', 'space did not encode correctly'

assert encode('3', alphabet, key) == '6', '3 did not encode correctly'

assert encode('9', alphabet, key) == 'C', '9 did not encode correctly'

print('test\_encode succeeded')

Question 8. (10 points) Write a test function for your encode function, checking that it works correctly for strings that contain more than one character. Create sufficient tests to have high confidence that the function is working correctly.

Question 9. (10 points) Write a decode(s, alphabet, key) function that decodes the string s using the given alphabet and key. Hint: The decode function can be implemented very simply using a call to the encode function.

Question 10. (10 points) Write a test function for your decode function, checking that it works correctly for strings that contain more than one character. Create sufficient tests to have high confidence that the function is working correctly.

Total points on assignment: 105