Instructions:

This assignment is a Pair Programming assignment.

You MUST use the function names indicated. Points will be deducted for any change in function name because of the extra effort the TA will require to grade your function.

Extra Credit: There is no specific extra credit problem. Instead each of the eight problems is worth 10 points (6 points for correctness, 4 points for documentation/readability), and the assignment is out of 70 points. If you answer all eight problems, you can earn up to 10 extra credit points.

What to submit (upload) to Sakai (four files):

Upload one file named 'lab4Problem1-7.py' containing the functions required in Problems 1-7.

Upload two ".png" files that you name 'Lab4Problem7piechart.png' and 'Lab4Problem7histogram.png' resulting from your solution to Problem 7.

Upload one file named 'lab4Problem8.py' containing the function required in Problem 8.

Problem 1: md5 Hash Code

Surf the Internet to learn what an 'md5 hash' is. Here are two starting web sites to consider:

http://en.wikipedia.org/wiki/MD5

http://www.go4expert.com/forums/showthread.php?t=319

Write a function 'computeMD5hash' that takes in a string parameter, and returns its md5 hash (also a string). (Hint: Unless you are an incredibly smart person and computer programmer, find existing Python md5 software, and have your function import that software.) No argument-parameter testing is required. Include at least 3 assertEqual tests. You may surf the Internet to help make up your assertEqual tests.

Problem 2: Base64 Encoding

Surf the Internet to learn about 'Base64' encoding and decoding. Here is one web site to consider: http://en.wikipedia.org/wiki/Base64

Write a function 'convertToBase64' that takes in a string, and returns its representation in Base64 encoding (also a string). (Hint: Your program may import an existing Python function that performs Base64 encoding.) No argument-parameter testing is required. Include at least 3 assertEqual tests. You may surf the Internet to help make up your assertEqual tests.

Problem 3: Base64 Decoding

Write a function 'convertFromBase64' that takes in a string encoded in Base64, and returns the string in its original form. (Hint: Your program may import an existing Python function that performs Base64 decoding.) No argument-parameter testing is required. Include at least 3 assertEqual tests. You may surf the Internet to help make up your assertEqual tests.

Problem 4: Roots of a Quadratic Equation

Write a function "quadraticRoots" that takes in the 3 real coefficients (a,b,c) of a quadratic equation, and returns a list of the equation's real roots. Depending on the values of a,b,c, the list may have 2 , 1, or no real roots. (Do not consider imaginary roots for this question.) If there are no real roots, then the function should return an empty list. Even though when a=0, the equation is linear and not quadratic, your function should return a list with the 1 real root in this special case. When there are two roots, the smaller value should be first in the list. IMPORTANT: the function returns a list even in the case of only 1 root. Required argument-parameter testing: verify that all arguments passed to the parameters are numbers (any combination of int or float). If not, have your function return None. Include at least 3 assertEquals testing valid arguments, and at least 2 assertEquals testing invalid arguments.

Problem 5: Decimal to Hexadecimal (base 16) Conversion using Recursion

Hexadecimal (herein called 'hex') is a system of counting with a base of 16 as opposed the base of 10 which is the base for our decimal system. Hexadecimal is popular in computer science because 4 bits can exactly hold one hexadecimal digit. We often want to convert between base 10 (decimal) and base 16 (hex). The hex system, being base 16, counts as follows:

0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F,10,11,12,13,14,15,16,17,18,19,1A,1B,1C,1D,1E,1F,20,21,22,23,…

These hex numbers represent the decimal numbers, respectively:

0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,…

So, 14 hex is equal to 20 decimal; 27 decimal is equal to 1B hex. (If you do not understand hex at this point, go surf the Internet and learn about it. Do not continue unless you definitely understand hex.)

To convert from decimal to hex, keep dividing the decimal number by decimal 16 converting the remainder to a hex digit, and keeping track of the quotient. Then repeat the process on the quotient.

For example, convert 483 (base 10) to hex:

483 / 16 has quotient 30 and remainder 3

30/16 has quotient 1 and remainder 14

1/16 has quotient 0 and remainder 1

Hence 483 base 10 equals 1-14-3 base 16 which is written as 1E3.

Write a RECURSIVE function, 'convert10to16' that takes in a base 10 integer, and returns a base 16 string. Required argument-parameter testing: verify the passed argument is a non-negative base 10 integer. If not, return None. Your function MUST be recursive. Include at least 3 assertEquals testing valid arguments, and at least 2 assertEquals testing invalid arguments.

Problem 6: Hexadecimal to Decimal Conversion

Write a function, 'convert16to10' that takes in a base 16 string and returns a base 10 integer. Required argument-parameter testing: verify that the base 16 string is valid (both a string and made up only of the characters 0-9,A-F). If not, return None. Your function may or may not be recursive. Include at least 3 assertEquals testing valid arguments, and at least 2 assertEquals testing invalid arguments.

Problem 7: Plotting in Python

Read about plotting in Python: http://matplotlib.sourceforge.net/

The distribution of grades on a Chem Eng 110 midterm exam is: 21-90's, 25-80's, 15-70's, 6-60's, and 3-50's. Use Python plotting software to produce two figures displaying these grades: a meaningful pie chart and a meaningful histogram. The pie chart should have a title and labels on each portion of the pie. The histogram should have a title and labels on both axes. Save and upload the figures to Sakai using file names: 'Lab4Problem7piechart.png' and 'Lab4Problem7histogram.png'.

Problem 8: File IO (put this answer in a separate file - 'lab4Problem8.py'

Look over the Python File IO examples used in class (see the Sakai class web site.)

Write a function "inputRoster" that prompts a user for the name of a class roster. This roster has names in the format "lastName,firstName\n" (no spaces). Your function should read (i.e., input) this file, and create (i.e., write out to disk) a new roster file with the format "firstName lastName\n" (one space between names, no comma). Your program MUST be written to prompt the user for the name of the roster file to be read. The new roster that your program creates MUST be named 'rosterFirstLast.txt' (Note: there are no blanks in this file name).

To help you test your program, a sample class roster named 'rosterLastFirst.txt' is an attachment to this assignment. Test your program by running it on this sample roster and several different rosters that you make up. Verify that the output file looks correct by opening it with an editor (e.g., notepad, vi, emacs) No assertEqual tests for this function are expected. The TA will run your program by hand to verify it works correctly.