Lab Programming Assignment #3

Points Possible: 20

Due Date: Friday, November 1 by 11:59 pm

Read the instructions at least twice, to ensure you follow it. Failure to follow the instructions will result in point deductions.

Instructions:

You can work in teams of up to three or individually, up to you. Submit just one file: **Programming Assignment 3-<Name of Submitter>.py**. Example: Programming Assignment 3-John Doe.py assuming John Doe is the person submitting the file on BeachBoard. Only submission via BeachBoard will be accepted.

That file should contain all of your answers (which should be in Python code).

That file should also be executable and display all of your answers via the following command in the Python command prompt:
exec(open("<Your Filename>.py").read())

Example:
exec(open("Programming Assignment 3-John Doe.py").read())

A part of my testing will be executing that command, so it is important that that command work with your Python file and display the answers.

Be sure to also type in your name and the names of your team members within the file, at the top. When typing your Python code, clearly indicate the question number above it so I know which question you're answering. You should write them in your file as Python comments.

Also, ensure the function names are EXACTLY that of what is listed in the problems below.

Do not use any print statements. Instead, it should use the **return** function.

Failure to do any of the above will result in point deductions. No exceptions.

(20 points)

Chinese Remainder Theorem (See Theorem 2 of Section 4.4 of *Discrete Mathematics and Its Applications* book by Kenneth Rosen 7th edition):

Write a function that will take in a list of lists. The list will contain a list of tuples, with the first value being the **a** and the second value being the modulus. For example, a_1 and m_1. Your function should then provide the final x value after it has been modded by m. The x value should be returned as an integer.

Assuming L is the list of lists, then L can be of any length. Note that each sub-list within L will have an exact length of 2, with the first value being **a** (e.g. a_1) and the second value being the modulus (e.g. m_1).

Also note that if the moduluses aren't relatively prime, then the output of the function should be "Cannot proceed, the modulus values are not relatively prime". Thus, you should also implement a relatively prime function to ensure the list of moduluses are relatively prime.

The function name must be ChineseRemainderTheorem.

See below for examples. Note that the first example is the same as that of Example 4 and 5 of Section 4.4 of *Discrete Mathematics and Its Applications* book by Kenneth Rosen 7th edition.

Examples:

```
>>> ChineseRemainderTheorem([[2,3],[3,5],[2,7]])
23
>>> ChineseRemainderTheorem([[1,5],[2,7],[3,9],[4,11]])
1731
>>> ChineseRemainderTheorem([[1,5],[2,7],[3,9],[4,11],[8,22]])
'Cannot proceed, the modulus values are not relatively prime'
```