CIS 41A - Lab 2: if statements, loops, functions, programming logic

Write a program that asks the user for a positive integer between 0 and 1 billion, exclusive, then prints the number as a text string. For example, the number 2,024 will be printed as: two thousand twenty-four.

Background info on how to form number strings:

* A large integer is divided into 3-digit triplets, often separated by a comma. For example: 12,078
* For a number up to 1 billion there are up to 3 triplets: millions, thousands, and ones. For example: 123,456,789 has 123 million as the first triplet, 456 thousand as the second triplet, and 789 as the third triplet.
* For each triplet there are 3 place values: hundreds, tens, and ones. For example: 235 has 2 hundreds, 3 tens, 5 ones.
* The tens are divided into 2 types:
  + From 20 - 90, the tens text strings are twenty, thirty, forty... and can optionally go with a ones value.  
    If there is a ones value, then there is a dash in between the tens and ones values. For example: twenty-three
  + From 10 - 19, the text strings are ten, eleven, twelve, thirteen... and there is no ones value because it's already 'built-in'

Download the file convert.py, which has 3 convert functions: toTens, toTeens, and toOnes

The function toTens accepts a number in the range 20 - 90 as input argument, and returns the equivalent text strings 'twenty' to 'ninety'.

The function toTeens accepts a number in the range 10 - 19 as input argument, and returns the equivalent text strings 'ten' to 'nineteen'.

The function toOnes accepts a digit in the range 0 - 9 as input argument, and returns the equivalent text strings 'one' to 'nine'.

Follow these steps to complete the lab assignment:

1. At the top of your source file, import the convert.py module with: import convert

(Note that there’s no .py in the import statement)

1. Write a function getInput that will:

* ask the user for a positive integer under 1 billion.
* keep prompting until you get an integer between 0 and 1 billion, exclusive
* if the user input is not an integer or is out of range, print an error message
* the user can optionally use commas in the input number. Quietly remove the comma (which means no printing to screen that the comma is removed). You don't have to check whether the comma is at the right location (in between 2 triplets).

1. Write a function toWords that will:

* check if there is the first triplet in the number (the millions), and if it exists, call the convertTriplet function to convert the millions triplet to a text string.
* check if there is the second triplet in the number (the thousands), and if it exists, call the convertTriplet function to convert the thousands triplet to a text string.
* check if there is the third triplet in the number (the ones), and if exists, call the convertTriplet function to convert the ones triplet to a text string.

1. Write a function convertTriplet that will:

* check if there is a hundreds value, and if it exists, call the toOnes function in the convert module to find the equivalent word for the hundreds. (Example: two hundred)
* check if there is a tens value between 20 - 90, and if it exists, call the toTens function in the convert module to find the equivalent word for the tens. (Example: thirty)
* check if there is a teens value between 10 - 19, and if it exists, call the toTeens function in the convert module to find the equivalent word for the teens. (Example: fifteen)
* check if there is a ones value between 0 - 9, and if it exists, call the toOnes function in the convert module to find the equivalent word for the ones.
* Don’t forget to add a dash (-) for values between 21 and 99 that are not divisible by 10. Example: thirty-eight
* Make your if statements efficient so that Python doesn’t have to evaluate any unnecessary if conditions.

1. Write a main function that will:

* call the getInput function to get a valid integer
* call the toWords function to convert the number to a text string
* print the text string
* ask the user if they want to continue
  + if the user answers 'y' or 'Y', go back to the first step and call getInput again
  + If the user answers 'n' or 'N', end the program
  + If the user enters anything else, print an error message and ask if they want to continue again

Additional requirements:

* The program must have the 4 functions as outlined above, and each should do their tasks shown above.
* There is no description of input arguments or return values for each function. It's up to you to decide what data to pass to the functions and what data to return from the functions.

This also means no global variables should be used in your program (or in any CIS 41A program for this quarter).

* Add any necessary word (such as 'thousand' or 'hundred') as needed to the output text string.
* Words in the output text string should be separated by 1 space. They should not be jammed together with no space in between, and they should not be separated by more than 1 space.
* You should not have to use any Python data containers (list, etc.) or any Python module that we haven't covered.
* Don't forget docstrings for each function, except main.
* Don't forget the beginning documentation block with your name and lab number.

When done, make sure your file name is lab2.py before uploading to Canvas.

Sample output (user input in blue):

Enter a positive integer less than 1 billion: 103

Number is: one hundred three

Another number? y/n: y

Enter a positive integer less than 1 billion: 23,400

Number is: twenty-three thousand four hundred

Another number? y/n: y

Enter a positive integer less than 1 billion: 999,999,999

Number is: nine hundred ninety-nine million nine hundred ninety-nine thousand nine hundred ninety-nine

Another number? y/n: y

Enter a positive integer less than 1 billion: 1001

Number is: one thousand one

Another number? y/n: y

Enter a positive integer less than 1 billion: 1234567

Number is: one million two hundred thirty-four thousand five hundred sixty-seven

Another number? y/n: y

Enter a positive integer less than 1 billion: 12a

positive integer less than 1 billion only

Enter a positive integer less than 1 billion: -5

positive integer less than 1 billion only

Enter a positive integer less than 1 billion:

positive integer less than 1 billion only

Enter a positive integer less than 1 billion: 123,456,789,012

positive integer less than 1 billion only

Enter a positive integer less than 1 billion: 1

Number is: one

Another number? y/n: 12

Another number? y/n: a

Another number? y/n: n