CSC 1301 – FALL 2022
HOMEWORK #02
DUE 10/04/2022 11.59 PM
FINDING FACTORS

Design and Test Cases Due: 10-4-22 @11.59PM
Source Code Due: 10-4-22 @11.59PM

Use the I-college drop box for submitting both files. Both files should be .py files.

Note about collaboration: As stated in the syllabus and on the first day of class, you are allowed to choose ONE other student from this class and work with them as a partner. Either student must NOT have already worked with someone else in the class on this assignment. Pairs are allowed, not triples nor chains or rings! Choose partners BEFORE the due date. You do not want to make your partner late too! You must reference your partner AND they must reference you in the submitted work of both of work, regardless of which direction the "help" went. . **All program source code and designs will be checked for similarity. If similarity is found and no references are given, it will be investigated for plagiarism.**

Assignment total points = Design (20 points) + Test Case Answers (15 points) + Runs given test cases successfully (10 points) + Implementation correctly written (60 points) = 105 points using formatting in print statements

Problem Description:

Definition: an integer **a** is divisible by a number **b** iff there exists an integer **c** so that ac=b

(Note: this means that every number a is divisible by +-1 and +-a. It also means that 0 is divisible by every integer)

Definition: an integer a>1 is called a prime if it is divisible by no integers except +1 or -1 and +a or-a

The specifics regarding negative numbers differ between textbooks. In many cases only positive numbers are considered, but we will consider them neither prime/not prime. Given user input of an integer and return if prime, not prime or negative in the following string. You must use ***sympy's isprime function*** to find if the integer is prime or not.

Demo "***sympy's isprime function***"

```
# Python program to check prime number
# using sympy.isprime() method

# importing sympy module
from sympy import *

# calling isprime function on different numbers
isprime(30)
isprime(13)
isprime(2)
```

Sample Run

```
     ***  isPrime  ***

Please provide an integer: 2

The integer 2 is prime.
```

Sample Run

```
     ***  isPrime  ***

Please provide an integer: -4

The integer -4 is negative.
```

Test Plan (25 points) Make sure your program gives these results. Part of our grading process will be to run your program with these cases. You get 15 points for working the results for A., B., and C. by hand. You get 10 points when your program gives the same results when we run them.

Description	**Inputs Integer**	**Expected Outputs Prime/Not Prime/is Negative**
Normal and boundary cases		
Normal case	2	A.
Normal case	6	B.
Normal case	657	C.
Boundary case, the number is negative	-1	Is Negative
Boundary case, the number is zero	0	Not Prime

(20 points) Design:

Write the design for the program in pseudocode as comments and submit it to I-college as "design2.py". **NOTE that we do not want Python code in this file!** Just comments which can be used in the implementation later.

- Give the three P's (purpose, pre- and post-conditions) and author info as usual. **The steps do NOT have to be numbered**.

```
·    # supply program prolog (3 P's)
·    #  main function
·        # Display introductory message
·        Your design here
```

(60 points) Implementation:

Write a Python program to implement your design. Start by making a copy of the Python file you have that has the design in it (possibly modified with improvements you or your partner came up with) and write your Python code between the commented lines of the design. Make sure you eliminate any syntax and semantics errors. Here is where test cases are important!

Specifications for the implementation

- This program uses input; you will have to prompt the user for the inputs. The inputs can be assumed to be integers.
- You must import sympy
- You must use at least two functions and use sympy.isprime() from sympy to find if the number is prime.
- Find a prime number should be its own function and should be called 'isItPrime' that prints the desired output message
- You should have a main function and you should call isItPrime to find out if the number is prime or not.
- Make sure you format the lines of the output as described. The line breaks and the punctuation should be as shown. The output messages should be **exactly** as given.