|  |  |
| --- | --- |
| **NAME** | **DUE** |
| **Tomoki Koike** | **31 March 2019** |

**Every Boiler Engineering Code – Entry Level Programming**

**Week 4 – Programming Exercises**

1. **(15 points, Falling Distance)** When an object is falling because of gravity, the following formula can be used to determine that the distance the object falls in a specific time period:

**d = 1/2gt2**

The variables in the formula are as followings: d is the distance in meters, g is 9.8, and t is the amount of time, in seconds, that the object has been falling.

Write a function name **falling\_distance** that accepts an object’s falling time (in seconds) as an argument. The function should return the distance, in meters, that the object has fallen during that time interval. Write a program that calls the function in a loop that passes the values 1 through 10 as arguments and displays the return value.

**CODE:**

## PROBLEM #1

# NAME: Tomoki Koike

# DUE: 3/31/2019

# DESCRIPTION: This program will calculate the distance an object falls in mid air using a function which accepts the falling time

# STAND: Class of 2020

##

# Function

def falling\_distance(time):

return 0.5 \* 9.8 \* time\*\*2 # Units in meters and seconds

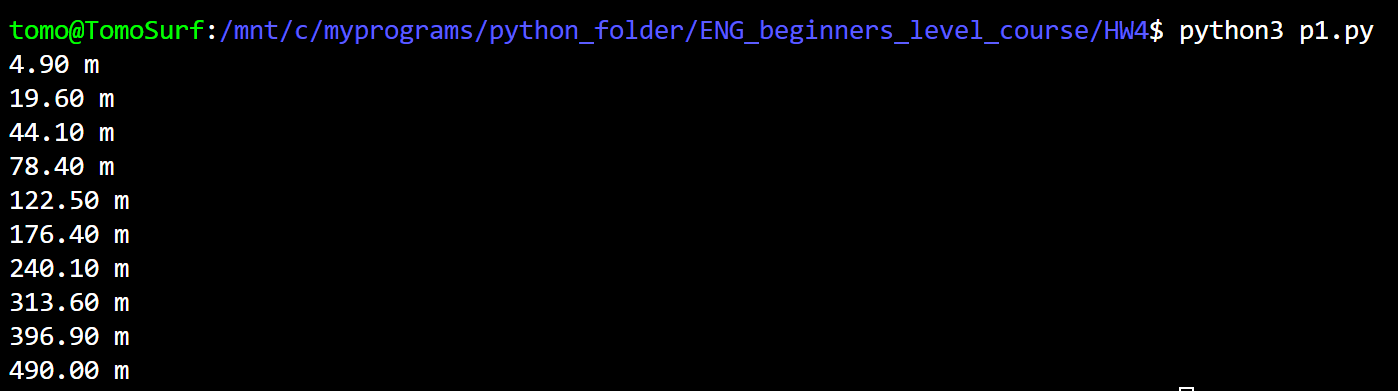
# Loop for executing the function for times 1 ~ 10

for x in range(1, 11): # Units in meters and seconds

# Printing output

print('{0:.2f} m'.format(falling\_distance(x)))

**SAMPLE EXECUTION:**



1. **(15 points, Math Quiz)** Write a program that gives simple math quizzes. The program should display two random numbers that are to be added, such as:

**247**

**+ 129**

The program should allow the student to enter the answer. If the answer is correct, a message of congratulations should be displayed. If the answer if incorrect, a message showing the correct answer should be displayed.

**(Test your program 3 times and take the screenshot for all the results.)**

**CODE:**

## PROBLEM #2

# NAME: Tomoki Koike

# DUE: 3/31/2019

# DESCRIPTION: This program will display a simple addition problem to the user to solve

# and if the user answers it correctly the program will congratulate the user; however,

# if the user answers incorrectly it will not.

# STAND: Class of 2020

##

# Importing random module

import random as rand

# Function that will display the problem

def rand\_addition(x = rand.randint(10, 10000)):

n1 = rand.randint(1, x)

n2 = rand.randint(1, x)

print(' {0:>4}'.format(n1))

print('+ {0:>4}'.format(n2))

print('--------')

answer = n1 + n2

return answer

# Function that will determine if the users answer is correct or not

def ans\_assess(user\_ans, correct\_ans):

if user\_ans == correct\_ans:

print('CONGRATULATIONS! Your answer is correct!')

else:

print('The correct answer is -> {0}'.format(correct\_ans))

return

# Generate random quiz

correct\_ans = rand\_addition()

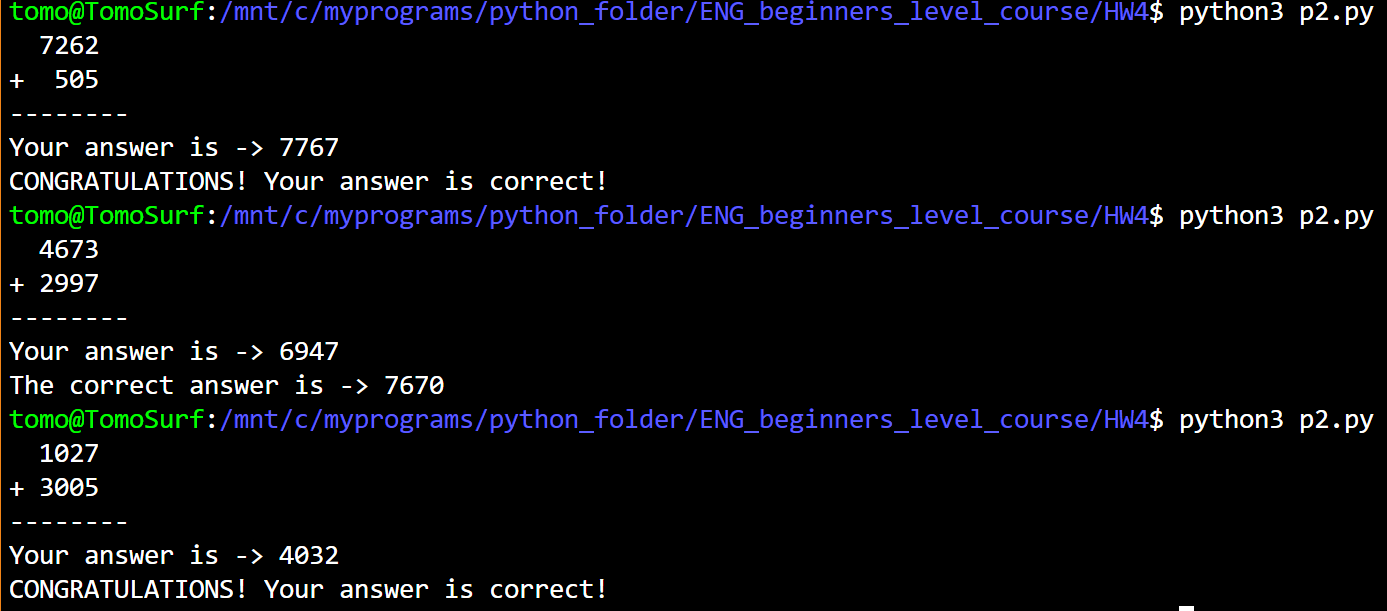
# Accept user input

user\_ans = int(input('Your answer is -> '))

# Judge if answer is correct

ans\_assess(user\_ans, correct\_ans)

**SAMPLE EXECUTION:**



1. **(20 points, Prime Numbers)**
   1. A prime number is a number that is only evenly divisible by itself and 1. For example, the number 5 is prime because it can only be evenly divided by 1 and 5. The number 6, however, is not prime because it can be divided evenly by 1, 2, 3, 6. Write a Boolean function named **is\_prime** which takes an integer as an argument and returns true if the argument is prime number, or false otherwise. Use the function in a program that prompt the user to enter a number then displays a message indicating whether the number is prime. (**Use 15, 2, 78, 3001 to test your program.**)
   2. Write another program that displays all the prime numbers from 1 to 100. The program should have a loop that calls the **is\_prime** function you create in a).

**(part a)**

**CODE:**

## PROBLEM #3

# NAME: Tomoki Koike

# DUE: 3/31/2019

# DESCRIPTION: This program will have a function that determines if an integer is # a prime number or not. Then generates some sample outputs with the function.

# After that there will be a second part where a loop will display all the prime

# numbers from 1 to 100 (with input validation)

# STAND: Class of 2020

##

# Function for input validation

def getValid(prompt):

while True:

try:

this = int(input(prompt))

except ValueError:

print('Sorry, could not understand. Please enter again.')

continue

if this > 0:

# Valid input

break

else:

# Invalid input

print('Error. Please enter a positive integer value.')

continue

return this

# Function for verfiying if the input number is a prime number or not

def is\_prime(int\_num):

counter = 0

for x in range(1, int\_num+1):

if int\_num % x == 0:

counter += 1

if counter <= 2:

judge = True

else:

judge = False

return judge

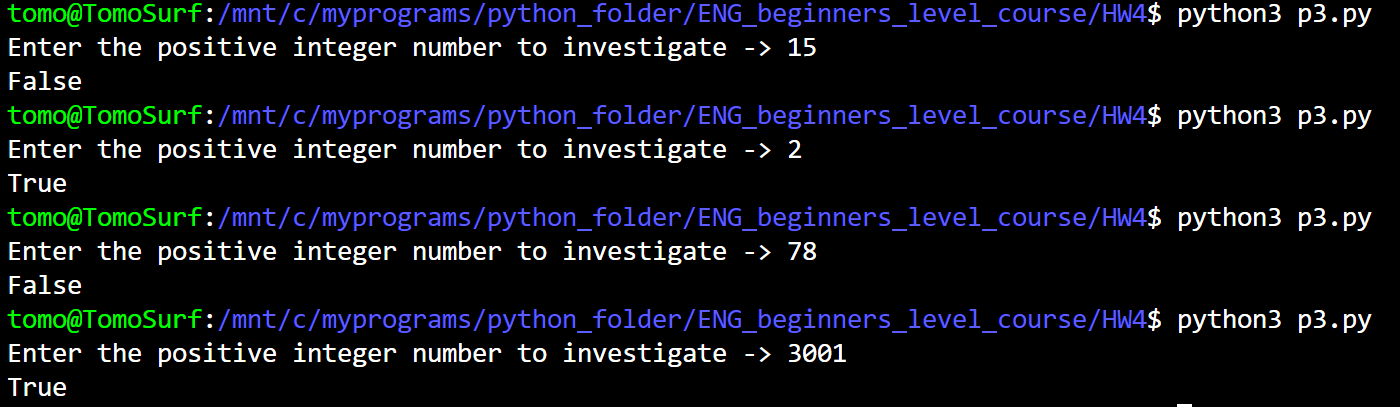
# (part a)

# Accepting the user input and executing the function to determine if a positive interger number

# prime number or not

is\_prime(getValid('Enter the positive integer number to investigate -> '))

**SAMPLE EXECUTION:**



**(part b)**

**CODE:**

# (part b)

# Printing out the prime numbers from 1 to 100

for x in range(1, 11):

for y in range(1, 11):

if is\_prime((x-1) \* 10 + y) == False:

print(' -', end = ' ')

else:

print('{0:>3}'.format((x-1) \* 10 + y), end = ' ')

print()

**SAMPLE EXECUTION:**

