**Digital assignment 2(python)**

**Name:harsh kumar srivastava**

**Regno:22MCA0310**

**Question 1**

**Percentage of non-defective items**

A bag contains a total of 'T' nuts and bolts. Out of which there are 'N' nuts. In a quality check, x% of the nuts and y% of bolts in the bag were found to be defective. Design an algorithm and write a Python code to determine the percentage of non-defective items in the bag. For example if T is 200, N is 150, x is 50 and y is 20 then the percentage of non-defective items in bag is 57.5. Round the answer to two decimal places using format function.

Input for the Problem

'T' – total number of nuts and bolts in the bag

'N' – Number of nuts

'x' – Percentage of defective nuts in the bag

'y' – Percentage of defective bolts in the bag

Output of the Problem

Percentage of non-defective items in bag

Boundary Conditions

All inputs > 0

**Test Data**

**Input**

200

150

50

20

$END

300

100

40

15

**Output**

57.50

$END

76.67

CODE:

*# Taking inputs from user*

T = int(input("Enter the total number of nuts and bolts in the bag: "))

N = int(input("Enter the number of nuts in the bag: "))

x = int(input("Enter the percentage of defective nuts in the bag: "))

y = int(input("Enter the percentage of defective bolts in the bag: "))

*# Calculating the number of bolts*

B = T - N

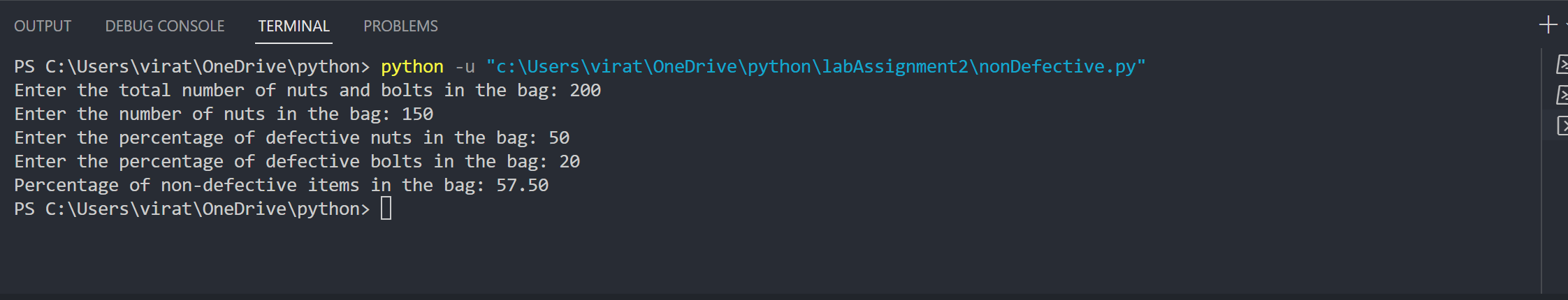
*# Calculating the percentage of non-defective items in the bag*

non\_defective\_percentage = ((100 - x) \* N + (100 - y) \* B) / T

*# Rounding off the answer to two decimal places using format function*

print("Percentage of non-defective items in the bag: {:.2f}".format(non\_defective\_percentage))

output:



**Question 2**

Calculate the number of birthdays celebrated by Mr.X. If Mr.X was born on 29th February of a leap year then he celebrates birthday only in leap years. Given the date of birth of Mr.X and the current year, design an algorithm and write the Python code to determine the number of birthdays celebrated by Mr.X. A year is a leap year if it is divisible by 4 and not divisible by 100 or when the year is divisible by 400. For example, year 1996 and 2000 are leap years whereas 1900 is not a leap year. Assume that the current day and month is greater than day and month of birthday.

**Input Format:**

day of birth

month of birth

year of birth

Current year

**Output Format:**

Number of birthdays celebrated by Mr. 'X'

**Boundary Condition:**

Current Year > Year of birth

**Input:**

29

2

1960

2010

$END

31

1

1975

2100

$END

3

2

1960

1950

**Output:**

12

$END

125

$END

Invalid input

Code:

*# Taking inputs from user*

day\_of\_birth = int(input("Enter the day of birth: "))

month\_of\_birth = int(input("Enter the month of birth: "))

year\_of\_birth = int(input("Enter the year of birth: "))

current\_year = int(input("Enter the current year: "))

*# Checking for invalid input*

if current\_year < year\_of\_birth:

    print("Invalid input")

else:

*# Calculating the number of birthdays celebrated by Mr. X*

    if month\_of\_birth == 2 and day\_of\_birth == 29:

*# Mr. X was born on 29th Feb*

        count = 0

        for year in range(year\_of\_birth, current\_year+1):

            if year % 400 == 0 or (year % 100 != 0 and year % 4 == 0):

*# Leap year*

                count += 1

    else:

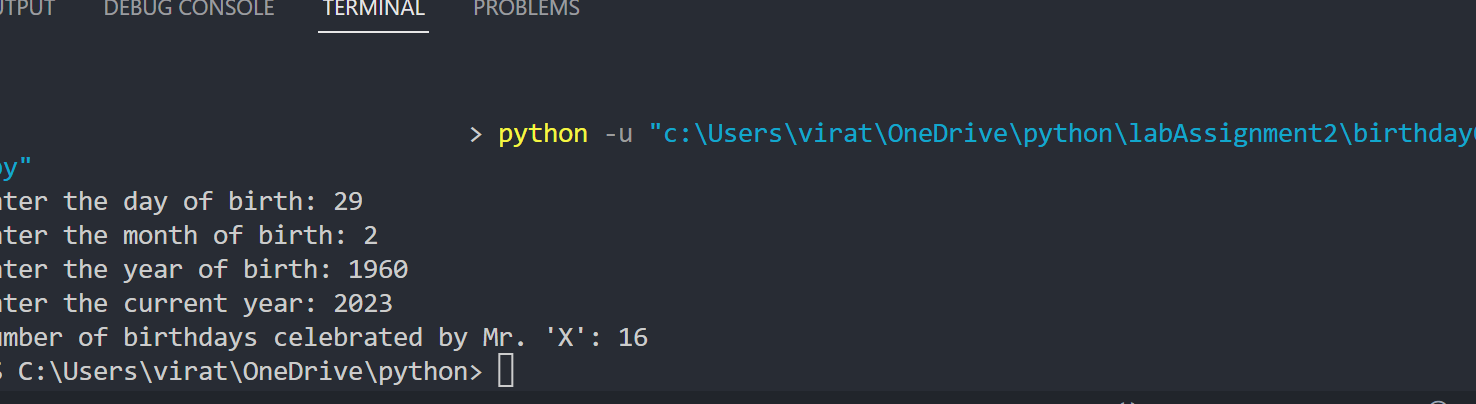
*# Mr. X was born on any other day than 29th Feb*

        count = current\_year - year\_of\_birth

*# Printing the output*

    print("Number of birthdays celebrated by Mr. 'X':", count)

output:



**Question 3**

A person 'X' has a farm which is rectangular in shape. 'X' wants to plant coconut trees in the farm and he has heard that every tree has to be separated by 2 feet. He needs to calculate the number of plants to be purchased. Write a program to calculate the number of plants when provided with the length and breadth of the farm (in feet). The program should display the number of rows and columns along with the total number of plants required for the farm. For example, if the length and breadth of the farm is 9 feet \* 4 feet then the farmer can plant trees in positions 0, 2, 4, 6, 8 along the length and 0,2,4 along the breadth therefore number of trees to be purchased is 5\*3 = 15.

**Input Format:**

The first input value read is the length of the farm (in feet).

The second input value read is the breadth of the farm (in feet).

**Output Format:**

Print number of rows

Print number of columns

Print number of plants required

Code:

*# Taking inputs from user*

length = int(input("Enter the length of the farm (in feet): "))

breadth = int(input("Enter the breadth of the farm (in feet): "))

*# Calculating the number of rows and columns*

rows = (length-2) // 4 + 1

columns = (breadth-2) // 4 + 1

*# Calculating the total number of plants required*

total\_plants = rows \* columns

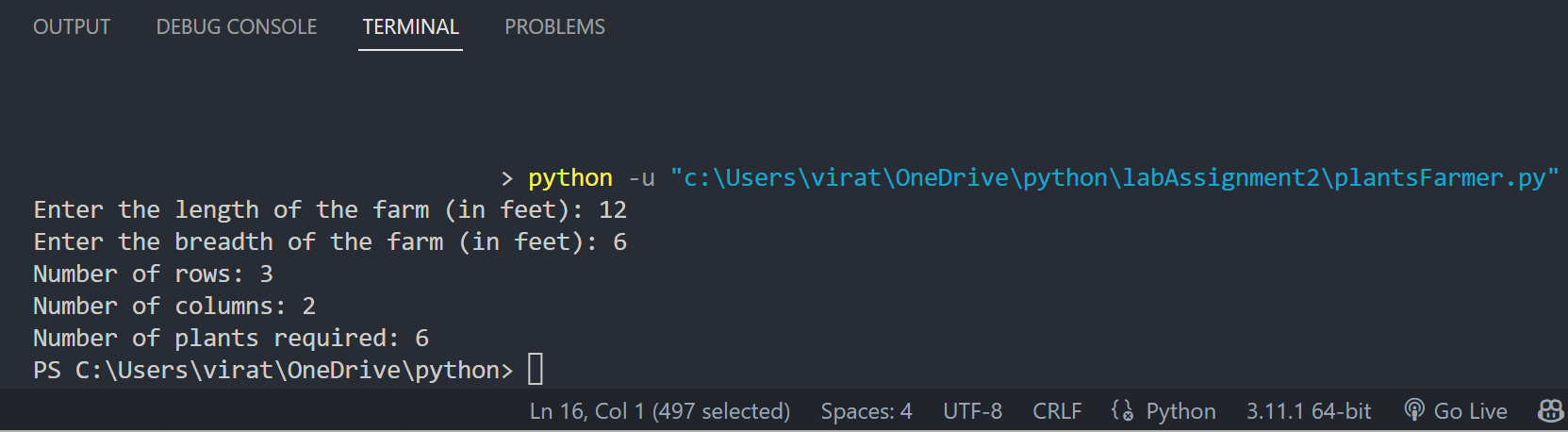
*# Printing the output*

print("Number of rows:", rows)

print("Number of columns:", columns)

print("Number of plants required:", total\_plants)

output:



**Question 4**

A finance consultant decides to create an application for the public who wants to decide on the type of investment they can make. He needs an application, which when given with the initial amount, no of years of investment ‘n’ and the expected amount after ‘n’ years, should tell the rate of interest based on which the customers will choose their investments. Assume that the application works with only simple interest calculation. Can you code for Raman? Use Rate of interest = ((expected amount-initial amount) / (initial amount \* no of years))\*100.

**Input Format:**

The first input value read is the initial amount the customer has.

The second input value read is the no of years he want to invest.

The third input value is the expected amount after ‘n’ years.

**Output Format:**

Print rate of interest

Code:

*# Taking inputs from user*

initial\_amount = float(input("Enter the initial amount: "))

no\_of\_years = int(input("Enter the number of years of investment: "))

expected\_amount = float(input("Enter the expected amount after {} years: ".format(no\_of\_years)))

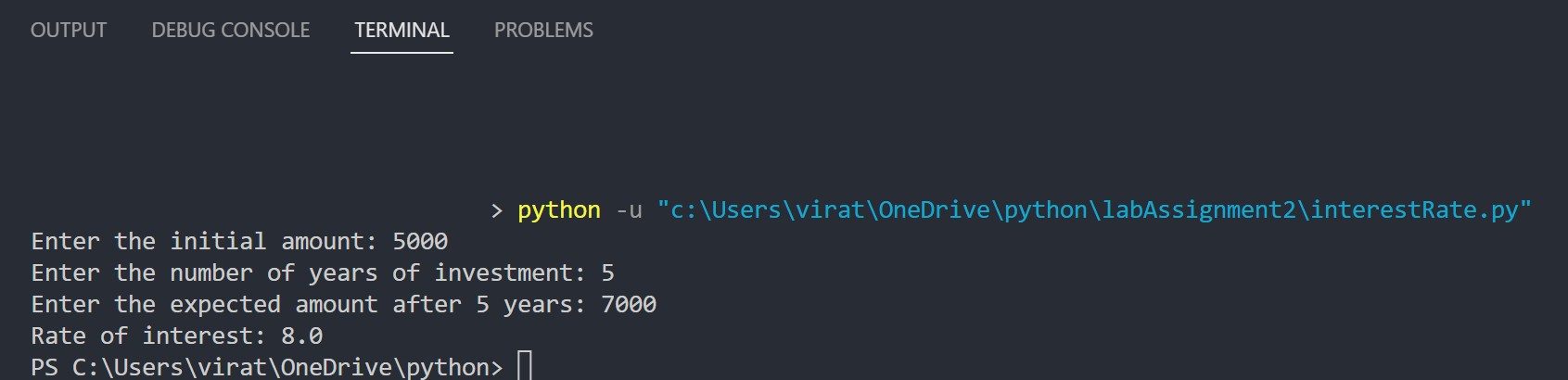
*# Calculating the rate of interest*

rate\_of\_interest = ((expected\_amount - initial\_amount) / (initial\_amount \* no\_of\_years)) \* 100

*# Printing the output*

print("Rate of interest:", rate\_of\_interest)

Output:



**Question 5**

Consider the thirsty crow story where a thirsty crow identifies a jug with little water. It puts pebbles into the water to raise the level of water and drinks it. Assume that the initial reading of the jug is 'm1' ml and the crow can drink water if the level of water has come to 'm2' ml. There are two categories of pebbles small and big in the field. Small pebble can raise the level of water by 'x' ml and big pebble can raise the level of water by 'y' ml. There are 'n' small pebbles. Crow prefers to place small pebbles in jug and then only takes big pebbles. Write an algorithm and the Python code to determine the number of pebbles required to raise the water to ‘m2’ level. For example, if value of 'm1', 'm2','x','y' and 'n' are 54, 300, 10, 20, 10 respectively then the number of pebbles required is 13.

**Input Format:**

Read the initial level of water in jug (in ml)

Read the level of water in jug required for drinking (in ml)

Read the height which small pebble will increase (in ml)

Read the height which big pebble will increase (in ml)

Read the number of small pebbles

**Output Format:**

Number of big pebbles required

**Code:**

*# Reading input values*

m1 = int(input("initial level of water in jug (in ml)"))

m2 = int(input("level of water in jug required for drinking (in ml)"))

x = int(input("height which small pebble will increase (in ml)"))

y = int(input("height which big pebble will increase (in ml)"))

n = int(input("number of small pebbles"))

*# Number of small pebbles required to increase water level to m2*

small\_pebbles = ((m2-m1)//x)

*# If the small pebbles cannot increase water level to m2, return -1*

if small\_pebbles < n:

    print("-1")

else:

*# Remaining height to be increased after using small pebbles54*

    remaining\_height = m2 - m1 - (n\*x)

*# Number of big pebbles required to increase remaining height*

    big\_pebbles = remaining\_height // y

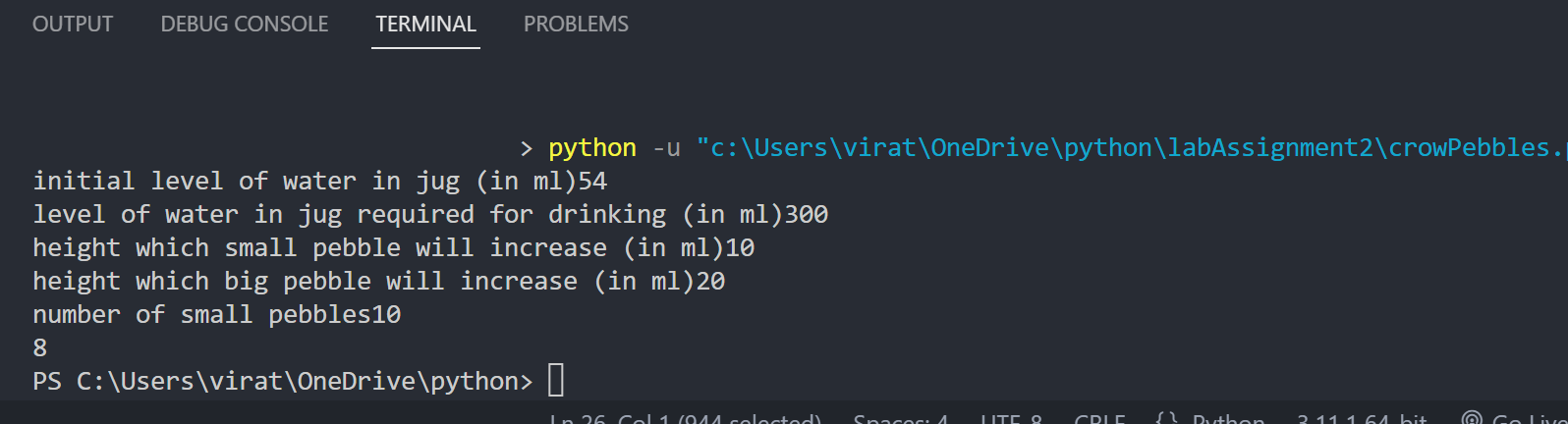
*# If big pebbles cannot increase remaining height, add one more big pebble*

    if remaining\_height % y != 0:

        big\_pebbles += 1

    print(big\_pebbles)

**Output:**

****

**Question 6**

Given 'n' points, design a flowchart or an algorithm and write the Python code to determine the pair of points that are far from each other. Distance between two points (x1, y1) and (x2, y2) is determined using the formula. Write a function to determine distance between the two points. Consider only two decimal places of distance for comparison.

**Code:**

import math

*# function to calculate distance between two points*

def distance(x1, y1, x2, y2):

    return math.sqrt((x1 - x2) \*\* 2 + (y1 - y2) \*\* 2)

*# read the number of points*

n = int(input("Enter the number of points: "))

*# read the coordinates of the points*

points = []

for i in range(n):

    x, y = map(float, input(f"Enter coordinates of point {i+1}: ").split())

    points.append((x, y))

*# initialize variables to store farthest points and distance*

max\_distance = 0

farthest\_points = (0, 0)

*# iterate through all possible pairs of points*

for i in range(n):

    for j in range(i+1, n):

*# calculate distance between current pair of points*

        d = distance(points[i][0], points[i][1], points[j][0], points[j][1])

*# update farthest points and distance if necessary*

        if d > max\_distance:

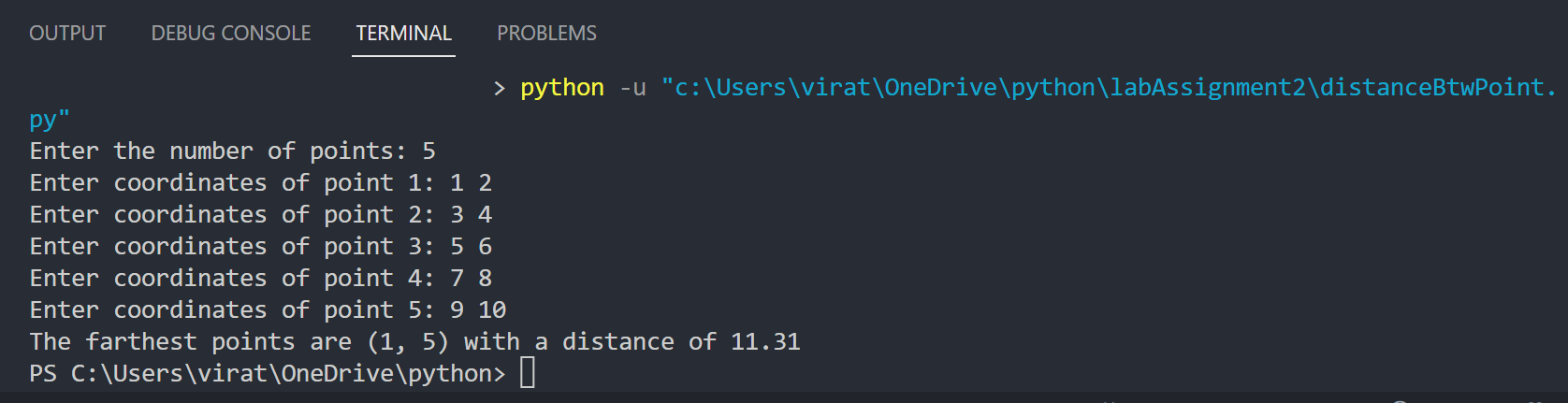
            max\_distance = d

            farthest\_points = (i+1, j+1)

*# print the indices of farthest points and distance between them*

print(f"The farthest points are {farthest\_points} with a distance of {max\_distance:.2f}")

**Output:**

****

**Question 7**

A person wants to know his Body Mass Index (BMI). He knows his weight in pounds and height in inches. The evaluator knows the formula for calculating BMI using the formula,

BMI = (weight in kilograms) / (height in m \* height in m)

Help the person in finding his BMI by writing a program for him. (Use the conversion formulae: 1 pound =0.4536 kilograms, 1 inch = 2.54 cms)

**Input Format:**

Weight of person in pounds.

Height of the person in inches.

**Output Format:**

BMI of the person calculated using the formula,

(weight in kilograms) / (height in m \* height in m)

**Code:**

weight\_pounds = float(input("Enter weight in pounds: "))

height\_inches = float(input("Enter height in inches: "))

*# Convert weight from pounds to kilograms*

weight\_kg = weight\_pounds \* 0.4536

*# Convert height from inches to meters*

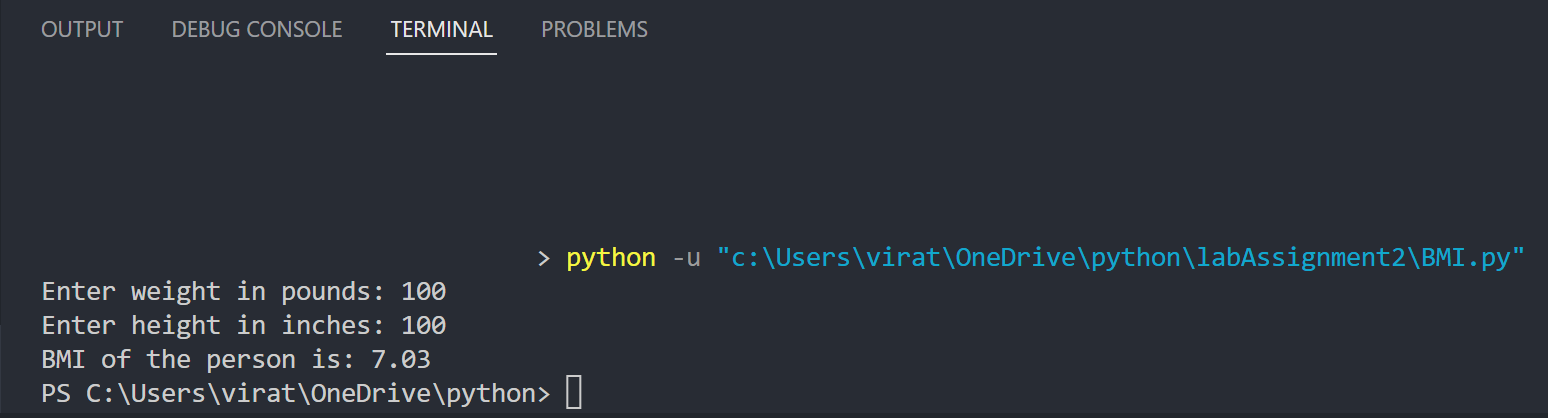
height\_m = height\_inches \* 0.0254

*# Calculate BMI using the formula*

bmi = weight\_kg / (height\_m \* height\_m)

print("BMI of the person is:", round(bmi, 2))

**Output:**

****