Learn Python

Basics

1. What is an expression?
   A expression is an instruction that combines values and operators and always evaluates down to a single value.
2. What is a syntax error?
   A syntax error is an error in the source code of a written program.
3. What is PEP8?
   PEP 8 is Python's style guide. It's a set of rules for how to format your Python code to maximize its readability.
4. What does a linter do?
   A lint or liner s a program that supports linting (Verifying code quality). Running a Lint program over your source code helps to ensure code is legible readable less polluted and easier to maintain.
5. What is the result of this expression: “*” * 10.
   Answer: - **********

Primitive Types

1. What is a variable?
   A variable is a reserved memory location to store values.
2. What are the primitive built-in types in Python?
   Primitive types are the most basic data structures. They are the building bloc for data manipulation and such contains pure simple values of data.

   Python has four primitive types:

   integers

   floats

   boolean

   string

3.When should we use “”” (tripe quotes) to define strings?

The triple quotation is a nice way to include other types of quotation within your string without having to use escape characters.

4.Assuming (name = “John Smith”), what does name[1] return?

Ans : o

5.What about name[-2]?

Ans : t

6.What about name[1:-1]?

Ans : ohn Smith

7.How to get the length of name?

Using len in-built method.

8.What are the escape sequences in Python?

In python string, the blackslash "\" is also called the "escape" character. It is used in representing certain whitespace characters. "\t" is a tab,"\n" is a newine, and "\r" is a carriage return. Conversely, prefixing a special character with "\" turns it into an ordinary character.

9.What is the result of f“{2+2}+{10%3}”?

4+1

10.Given (name = “john smith”), what will name.title() return?

john smith

11.What does name.strip() do?

john smith

Strip is an inbuilt function in python that returns a copy of the string with leading and trailing characters removed(based on the string argument passed)

12.What will name.find(“Smith”) return?

5. The strip() returns the lowest index of the substring if it is found in given string. If it is not found then it returns -1.

13.What will be the value of name after we call name.replace(“j”, “k”)?

kohn smith

14.How can we check to see if name contains “John”?

use the find() method. Name name.find("John"). Returns the lowest index of the substring if it is found in given string. If it's not found then it returns -1.

15.What are the 3 types of numbers in Python?

Int,float and complex.

Control Flow

1.What is the difference between 10 / 3 and 10 // 3?

10/3 == 3.33333 and 10//3 == 3.

Python has two operators, a single slash character for classic division and a double-slash for "floor divison". (rounds down to nearest whole number).

2.What is the result of 10 ** 3?

10000

3.Given (x = 1), what will be the value of after we run (x += 2)?

3

4.How can we round a number?

Using round() function. The round() method returns the floating point number rounded off to the given n digits after th;;e decimal point. If no n digits is provided, it rounds off the number to the nearest integer.

5.What is the result of float(1)?

1.0

6.What is the result of bool(“False”)?

True

7.What are the falsy values in Python?

Truthy values are values that evaluate to True in a boolean context. Falsy values are values that evaluate to False in a boolean context. Falsy values include empty sequences (lists, tuples, strings, dictionaries, sets), zero in every numeric type, None , and False.

None

False

0

0.0

0j

Decimal(0)

Fraction(0, 1)

[] - an empty list

{} - an empty dict

() - an empty tuple

'' - an empty str

b'' - an empty bytes

set() - an empty set

an empty range, like range(0)

objects for which

obj.__bool__() returns False

obj.__len__() returns 0

8.What is the result of 10 == “10”?

False

9.What is the result of “bag” > “apple”?

True. (ASCII of first character is being compared here)

10.What is the result of not(True or False)?

False. (True or False answer is True and not of True is False)

11.Under what circumstances does the expression 18 <= age < 65 evaluate to True?

Age must be inbetween 18 and 64. Including 18 and 64.

12.What does range(1, 10, 2) return?

13.Name 3 iterable objects in Python.

Lists,tuples,dictionaries.

Functions

1.What is the difference between a parameter and an argument?

A parameter is a variable in a method definition.

When a method is called, the arguments are the data you pass into the method's parameters.

Parameter is variable in the declaration of function.

Argument is the actual value of this variable that gets passed to function.

Parameters don’t change when the program is running

Arguments are probably going to be different every time the function is called

2.All functions in Python by default return?

Yes.In Python, every function returns something. If there are no return statements, then it returns None.

3.What are keyword arguments and when should we use them?

A keyword argument is where you provide a name to the variable as you pass it into the function.

we can use the name of the parameter irrespective of its position while calling the function to supply the values. All the keyword arguments must match one of the arguments accepted by the function.

4.How can we make a parameter of a function optional?

Just use the *args parameter, which allows you to pass as many arguments as you want after your a,b,c.

5.What happens when we prefix a parameter with an asterisk (*)?

*args allows for any number of optional positional arguments (parameters), which will be assigned to a tuple named args.

Asterisks for packing arguments given to function.

When defining a function, the * operator can be used to capture an unlimited number of positional arguments given to the function. These arguments are captured into a tuple.

Python's print and zip functions accept any number of positional arguments.

6.What about two asterisks (**)?

The **kwargs will give you all keyword arguments except for those corresponding to a formal parameter as a dictionary.

**kwargs allows for any number of optional keyword arguments (parameters), which will be in a dict named kwargs.

7.What is scope?

Scope. Variables can only reach the area in which they are defined, which is called scope. Think of it as the area of code where variables can be used. Python supports global variables (usable in the entire program) and local variables.

By default, all variables declared in a function are local variables.

8.What is the difference between local and global variables?

Global variables are declared outside any function, and they can be accessed (used) on any function in the program.

Local variables are declared inside a function, and can be used only inside that function. It is possible to have local variables with the same name in different functions.

9.Why is using the global statement a bad practice?

Global variables are dangerous because they can be simultaneously accessed from multiple sections of a program. This frequently results in bugs. Most bugs involving global variables arise from one function reading and acting on the value of a global variable before another function has the chance to set it to an appropriate value.

Coding Exercises

1.Write a function that returns the maximum of two numbers.

```
def findMaxOfTwo(a,b):
    global max
    if(a>b):
        max = a
    else:
        max = b
return max

max = findMaxOfTwo(5,5)
print("Maximun amount two : ", max)
```

2.Write a function called fizz_buzz that takes a number.

1. If the number is divisible by 3, it should return “Fizz”.
2. If it is divisible by 5, it should return “Buzz”.

3. If it is divisible by both 3 and 5, it should return “FizzBuzz”.
4. Otherwise, it should return the same number.

```
    def fizz_buzz(num):
        global value
        if(num%3 ==0 and num%5==0):
            value = "FizzBuzz"
        elif(num%5==0):
            value = "Buzz"
        elif(num%3==0):
            value = "Fizz"
        else:
            value = num
    return value

print(fizz_buzz(9))
```

3.Write a function for checking the speed of drivers. This function should have one parameter: speed.

1. If speed is less than 70, it should print “Ok”.

2.Otherwise, for every 5km above the speed limit (70), it should give the driver one demerit point and print the total number of demerit points. For example, if the speed is 80, it should print: “Points: 2”.

3.If the driver gets more than 12 points, the function should print: “License suspended”

```
    def speedTest(speed):
        if(speed < 70):
            print("Ok")
        elif(speed > 70):
            diff = speed-70
            dmt_points = diff//5
            if(dmt_points >= 12):
                print("License suspended")
            else:
                print("Demerited points",dmt_points)


    speedTest(45)
```

4.Write a function called showNumbers that takes a parameter called limit. It should print all the numbers between 0 and limit with a label to identify the even and odd numbers. For example, if the limit is 3, it should print:

- 0 EVEN
- 1 ODD
- 2 EVEN
- 3 ODD

```
def showNumbers(limit):
    for num in range(0,limit+1):
        if(num%2 == 0):
            print(num ," EVEN")
        else:
            print(num ," ODD")



showNumbers(3)
```

5.Write a function that returns the sum of multiples of 3 and 5 between 0 and limit (parameter). For example, if limit is 20, it should return the sum of 3, 5, 6, 9, 10, 12, 15, 18, 20.

```
def sumOfMul(limit):
    sum = 0
    for num in range(0,limit+1):
        if(num%3==0 or num%5==0):
            sum = sum + num
    return sum


print(sumOfMul(10))
```

6.Write a function called show_stars(rows). If rows is 5, it should print the following:

- *
- **
- ***
- ****
- *****

```
def show_stars(rows):
    for i in range(rows):
        for j in range(i+1):
            print("*",end="")
        print("\n")
show_stars(5)
```

7.Write a function that prints all the prime numbers between 0 and limit where limit is a parameter

```
def prime(limit):
    for i in range(2,limit):
        count = 0
        for j in range(2,i):
            if(i%j==0):
                count = count+1
                break
        if(count == 0):
            print(i)
prime(30)
```

8.What is the output of the following code?

```
age = 38
if (age >= 11):
print ("You are eligible to see the Football match.")
if (age <= 20 or age >= 60):
    print("Ticket price is $12")
else:
    print("Tic kit price is $20")
else:
    print ("You're not eligible to buy a ticket.")
```

Ans : You are eligible to see the Football match.Ticket price is $12

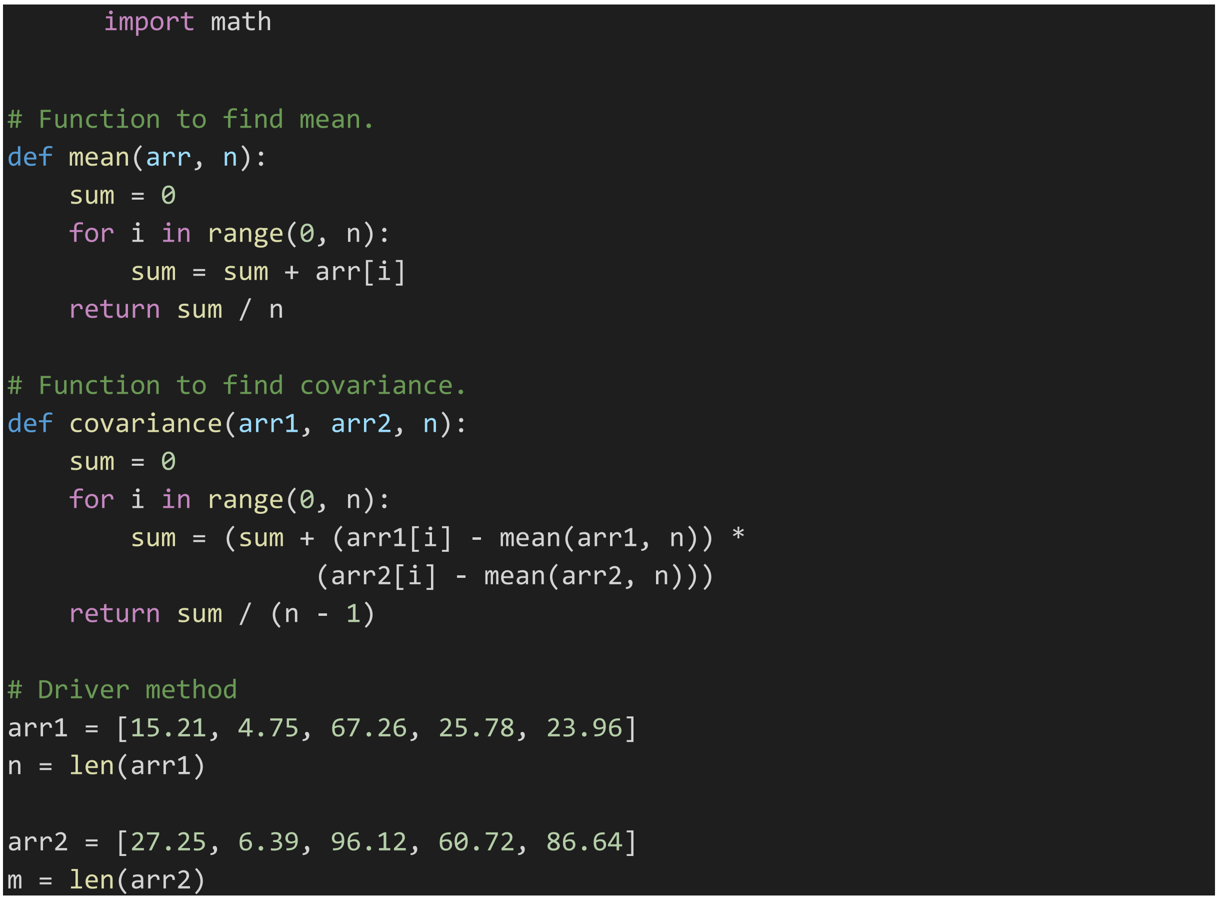
9.What is the output of the following piece of code?

```
n = 150

print(n)

#if n is greater than 500, n is multiplied by 7, otherwise n is divided by 7

result = n * 7 if n > 500 else n / 7

print(result)
```

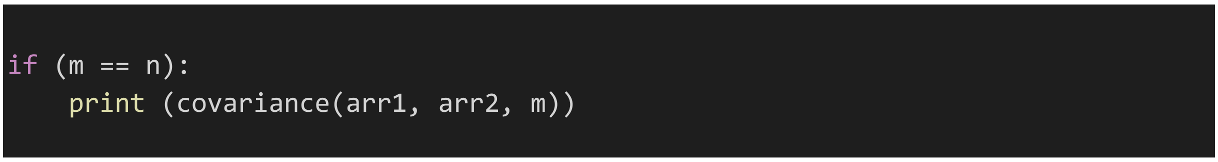
Ans: 150

    21

10.Write program to read 2X 2 matrix and find its covariance matrix. Eigen values and Eigen vectors of covariance matrix. Discuss what do you mean by eigen values and covariance matrix.

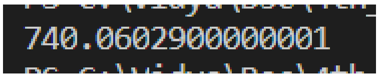


```
      import math


# Function to find mean.
def mean(arr, n):
    sum = 0
    for i in range(0, n):
        sum = sum + arr[i]
    return sum / n

# Function to find covariance.
def covariance(arr1, arr2, n):
    sum = 0
    for i in range(0, n):
        sum = (sum + (arr1[i] - mean(arr1, n)) *
                    (arr2[i] - mean(arr2, n)))
    return sum / (n - 1)

# Driver method
arr1 = [15.21, 4.75, 67.26, 25.78, 23.96]
n = len(arr1)

arr2 = [27.25, 6.39, 96.12, 60.72, 86.64]
m = len(arr2)
```



```
if (m == n):
    print (covariance(arr1, arr2, m))
```

Output:



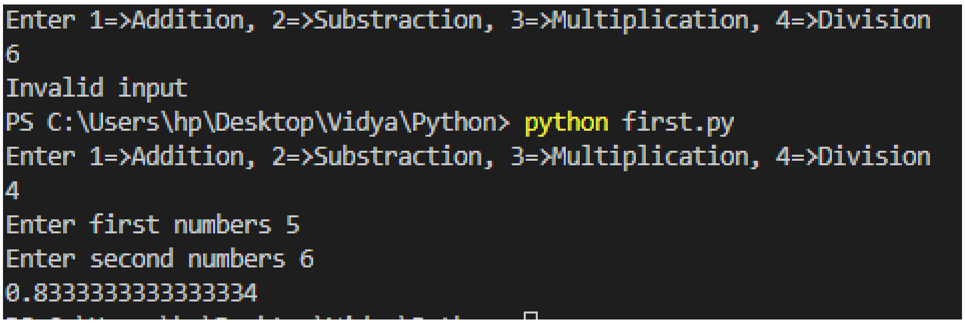
```
740.06029000000001
```

11Design a simple calculator to perform addition, subtraction, multiplication and division.

```
choice = int(input("Enter 1=>Addition, 2=>Substraction, 3=>Multiplication, 4=>Division\n"))

def Addition(a,b):
    return a+b

def Substraction(a,b):
    return a-b

def Multiplication(a,b):
    return a*b

def Division(a,b):
    return a/b

if(choice < 5):
    num1 = float(input("Enter first numbers "))
    num2 = float(input("Enter second numbers "))

    if(choice == 1):
        (Addition(num1,num2))
    elif(choice == 2):
        print(Substraction(num1,num2))
    elif(choice == 3):
        print(Multiplication(num1,num2))
    else:
        print(Division(num1,num2))
```

```
else:
    print("Invalid input")
```

Output :



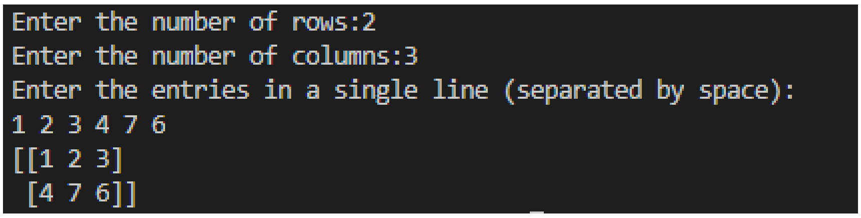
```
Enter 1=>Addition, 2=>Substraction, 3=>Multiplication, 4=>Division
6
Invalid input
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter 1=>Addition, 2=>Substraction, 3=>Multiplication, 4=>Division
4
Enter first numbers 5
Enter second numbers 6
0.8333333333333334
```

11.Perform following matrix operations:

a. Writing a data into given size matrix

```
import numpy as np

R = int(input("Enter the number of rows:"))
C = int(input("Enter the number of columns:"))


print("Enter the entries in a single line (separated by space): ")

# User input of entries in a
# single line separated by space
entries = list(map(int, input().split()))

# For printing the matrix
matrix = np.array(entries).reshape(R, C)
print(matrix)
```

Output :



```
Enter the number of rows:2
Enter the number of columns:3
Enter the entries in a single line (separated by space):
1 2 3 4 7 6
[[1 2 3]
 [4 7 6]]
```

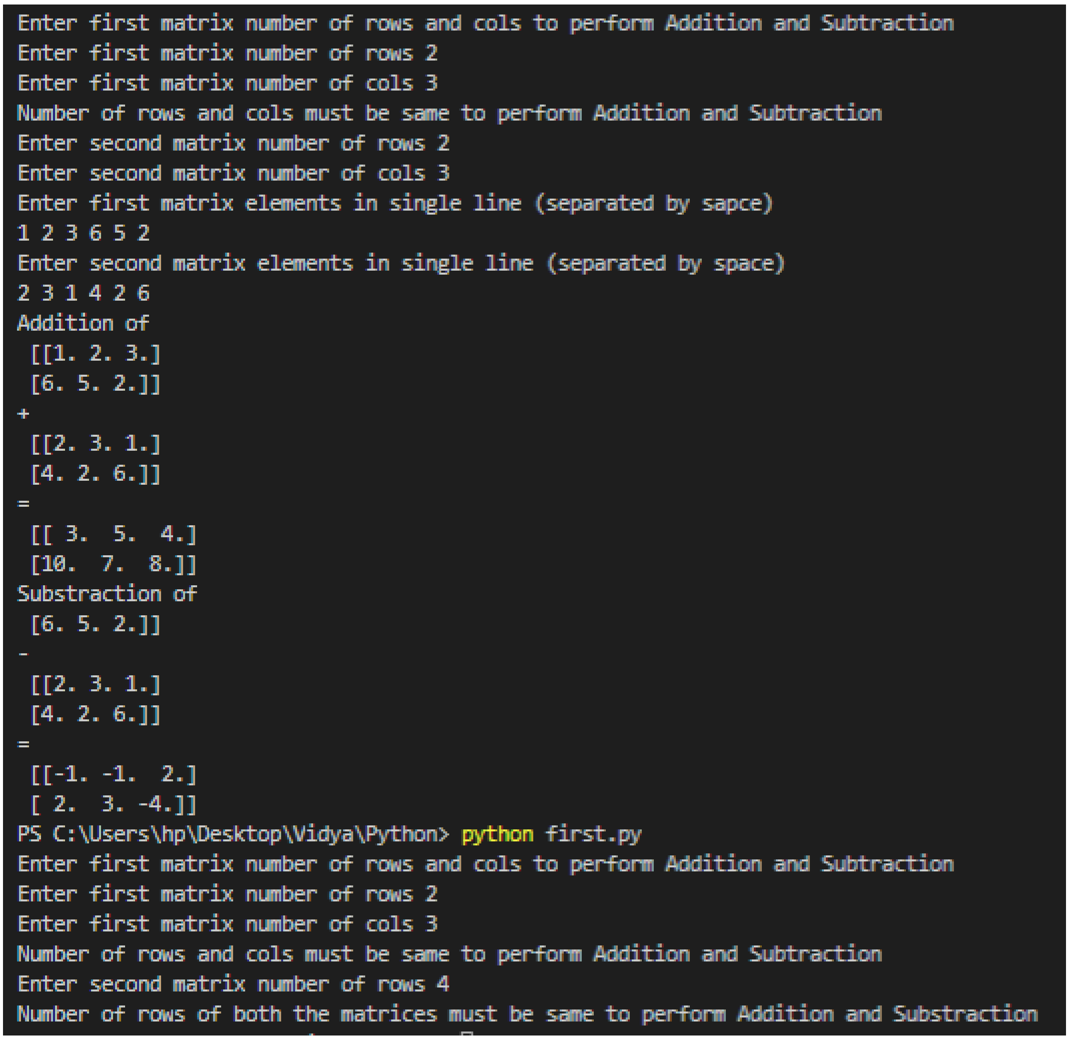
b. Reading a matrix to either row vector or column vector

c. Addition, subtraction, multiplication of matrices

Addition and Subtraction:

```
import numpy as np

print("Enter first matrix number of rows and cols to perform Addition and Subtraction")

row_1 = int(input("Enter first matrix number of rows "))
col_1 = int(input("Enter first matrix number of cols "))

print("Number of rows and cols must be same to perform Addition and Subtraction ")
row_2 = int(input("Enter second matrix number of rows "))

if(row_1 == row_2):
    col_2= int(input("Enter second matrix number of cols "))
    if(col_2 == col_1):
        print("Enter first matrix elements in single line (separated by sapce) ")
        fm_entries = list(map(float,input().split()))
        fm_matrix = np.array(fm_entries).reshape(row_1,col_1)

        print("Enter second matrix elements in single line (separated by space) ")
        sm_entries = list(map(float,input().split()))
        sm_matrix = np.array(sm_entries).reshape(row_2,col_2)

        #Add both the matrices
        print("Addition of\n",fm_matrix,"\n+\n",sm_matrix, "\n=\n",np.add(fm_matrix,sm_matrix))
        #Subtract both the matrices
        print("Substraction of\n",fm_matrix,"\n-
\n",sm_matrix, "\n=\n",np.subtract(fm_matrix,sm_matrix))

    else:
        print("Number of cols of both the matrices must be same to perform
 Addition and Substraction ")

else:
    print("Number of rows of both the matrices must be same to perform A
ddition and Substraction ")
```

Output:

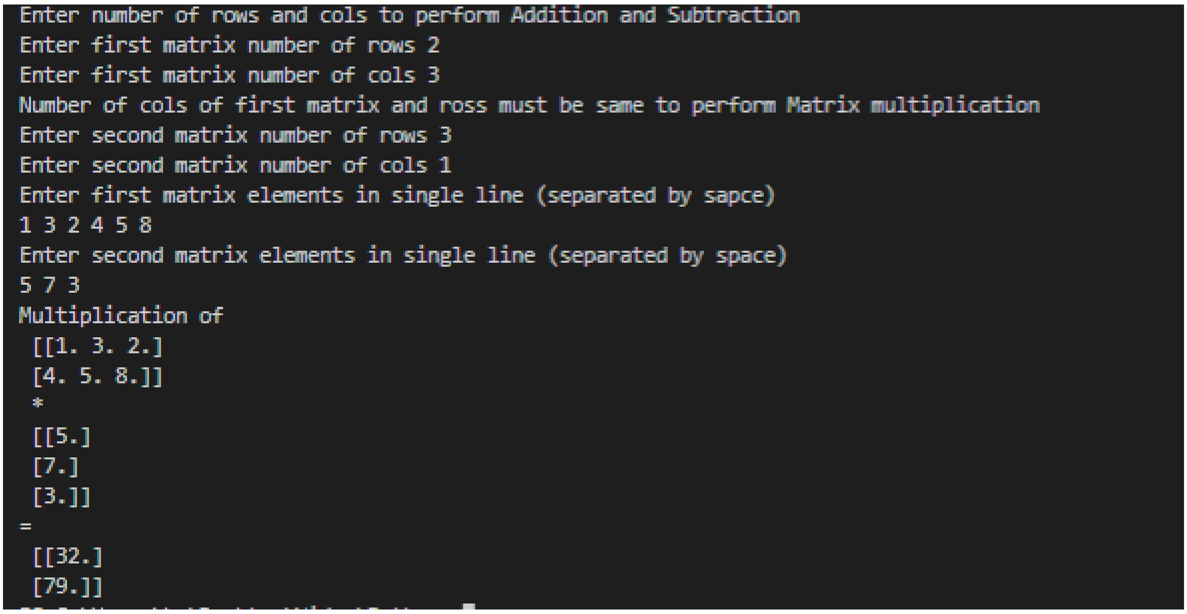


```
Enter first matrix number of rows and cols to perform Addition and Subtraction
Enter first matrix number of rows 2
Enter first matrix number of cols 3
Number of rows and cols must be same to perform Addition and Subtraction
Enter second matrix number of rows 2
Enter second matrix number of cols 3
Enter first matrix elements in single line (separated by sapce)
1 2 3 6 5 2
Enter second matrix elements in single line (separated by space)
2 3 1 4 2 6
Addition of
 [[1. 2. 3.]
 [6. 5. 2.]]
+
 [[2. 3. 1.]
 [4. 2. 6.]]
=
 [[ 3.  5.  4.]
 [10.  7.  8.]]
Substraction of
 [6. 5. 2.]]
-
 [[2. 3. 1.]
 [4. 2. 6.]]
=
 [[-1. -1.  2.]
 [ 2.  3. -4.]]
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter first matrix number of rows and cols to perform Addition and Subtraction
Enter first matrix number of rows 2
Enter first matrix number of cols 3
Number of rows and cols must be same to perform Addition and Subtraction
Enter second matrix number of rows 4
Number of rows of both the matrices must be same to perform Addition and Substraction
```

Multiplication:

```
import numpy as np

print("Enter first matrix number of rows and cols to perform Addition and Subtraction")

row_1 = int(input("Enter first matrix number of rows "))
col_1 = int(input("Enter first matrix number of cols "))

print("Number of cols of first matrix and ross must be same to perform Matrix multiplication ")
row_2 = int(input("Enter second matrix number of rows "))

if(col_1 == row_2):
    col_2= int(input("Enter second matrix number of cols "))
    print("Enter first matrix elements in single line (separated by sapce) ")
    fm_entries = list(map(float,input().split()))
    fm_matrix = np.array(fm_entries).reshape(row_1,col_1)
    print("Enter second matrix elements in single line (separated by space ) ")
    sm_entries = list(map(float,input().split()))
    sm_matrix = np.array(sm_entries).reshape(row_2,col_2)

    #Multifly Add both the matrices
    print("Multiplication of\n",fm_matrix,"\n*s\n",sm_matrix, "\n=\n",fm_matrix.dot(sm_matrix))

else:
    print("Number of cols of first matrix and ross must be same to perform Matrix multiplication ")
```

Output :



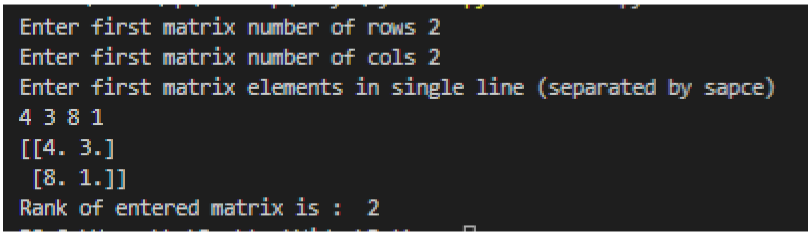
```
Enter number of rows and cols to perform Addition and Subtraction
Enter first matrix number of rows 2
Enter first matrix number of cols 3
Number of cols of first matrix and ross must be same to perform Matrix multiplication
Enter second matrix number of rows 3
Enter second matrix number of cols 1
Enter first matrix elements in single line (separated by sapce)
1 3 2 4 5 8
Enter second matrix elements in single line (separated by space)
5 7 3
Multiplication of
 [[1. 3. 2.]
 [4. 5. 8.]]
 *
 [[5.]
 [7.]
 [3.]]
=
 [[32.]
 [79.]]
```

d. Find the rank of a given matrix

```
import numpy as np
print("Enter first matrix number of rows and cols to perform Addition and Subtraction")
```

```
row_1 = int(input("Enter first matrix number of rows "))
col_1 = int(input("Enter first matrix number of cols "))

print("Enter first matrix elements in single line (separated by sapce) ")
fm_entries = list(map(float,input().split()))
fm_matrix = np.array(fm_entries).reshape(row_1,col_1)
print(fm_matrix)
print("Rank of entered matrix is : ",np.linalg.matrix_rank(fm_matrix))
```

Output :



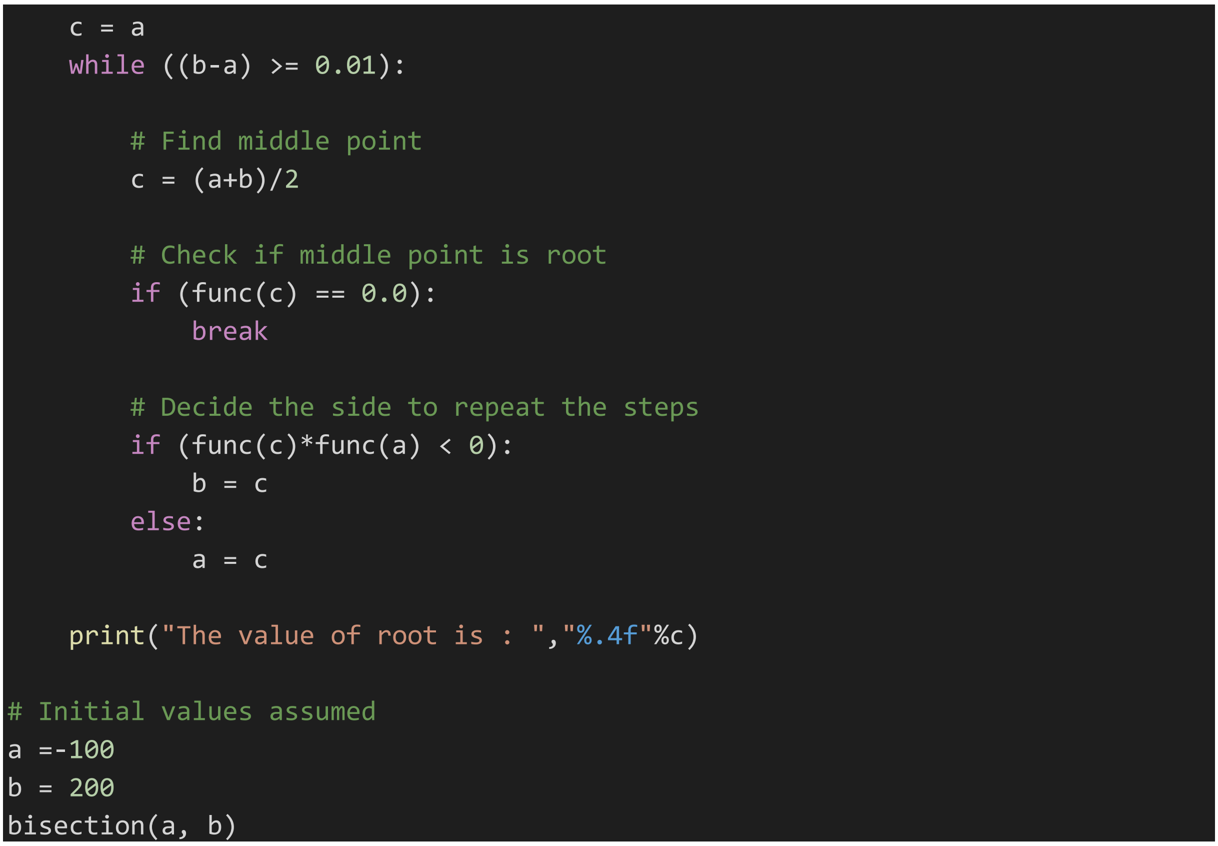
```
Enter first matrix number of rows 2
Enter first matrix number of cols 2
Enter first matrix elements in single line (separated by sapce)
4 3 8 1
[[4. 3.]
 [8. 1.]]
Rank of entered matrix is :  2
```

13.Perform data Analysis which includes:

- Read a data from a .xls file
- Write a date into a .xls file
- Plot a data from the file using different styles
- Compute Mean, median, standard deviation of a row and column separately

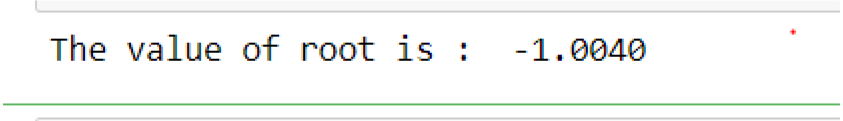
14.Use Bisection Method to find roots of a given function f(x)

```
      def func(x):

    return x*x*x - x*x + 2

# Prints root of func(x)
def bisection(a,b):

    if (func(a) * func(b) >= 0):
        print("You have not assumed right a and b\n")
        return
```



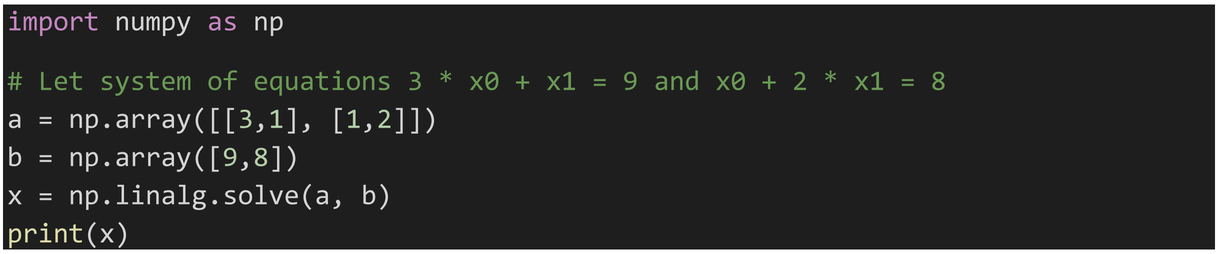
```
    c = a
    while ((b-a) >= 0.01):

        # Find middle point
        c = (a+b)/2

        # Check if middle point is root
        if (func(c) == 0.0):
            break

        # Decide the side to repeat the steps
        if (func(c)*func(a) < 0):
            b = c
        else:
            a = c

    print("The value of root is : ","%.4f"%c)

# Initial values assumed
a =-100
b = 200
bisection(a, b)
```

Output:



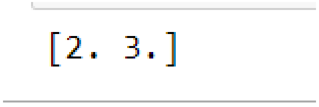
```
The value of root is :  -1.0040
```

15.The *Bisection Method* is a successive approximation *method* that narrows down an interval that contains a root of the function f(x).



```
import numpy as np

# Let system of equations 3 * x0 + x1 = 9 and x0 + 2 * x1 = 8
a = np.array([[3,1], [1,2]])
b = np.array([9,8])
x = np.linalg.solve(a, b)
print(x)
```

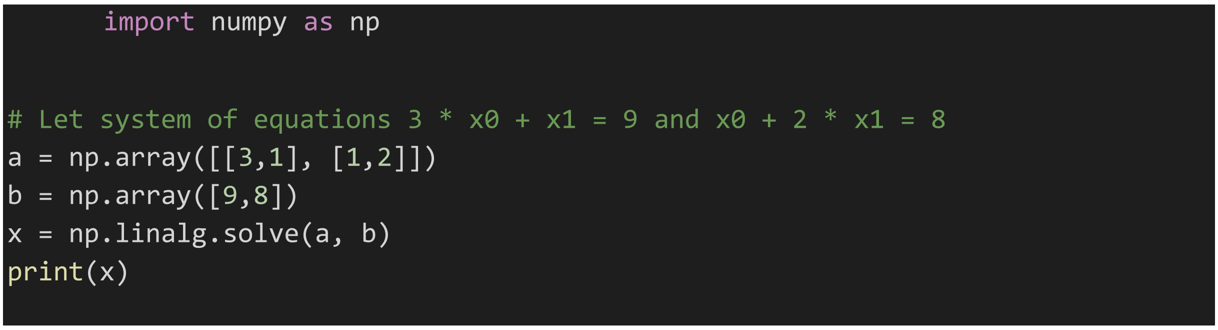
Output:



```
[2. 3.]
```

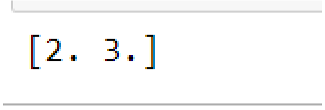
16.Use Cramer’s rule to solve minimum 3 linear equations

Cramer's rule is an explicit formula for the solution of a system of linear equations with as many equations as unknowns, valid whenever the system has a unique solution.



```
      import numpy as np


# Let system of equations 3 * x0 + x1 = 9 and x0 + 2 * x1 = 8
a = np.array([[3,1], [1,2]])
b = np.array([9,8])
x = np.linalg.solve(a, b)
print(x)
```

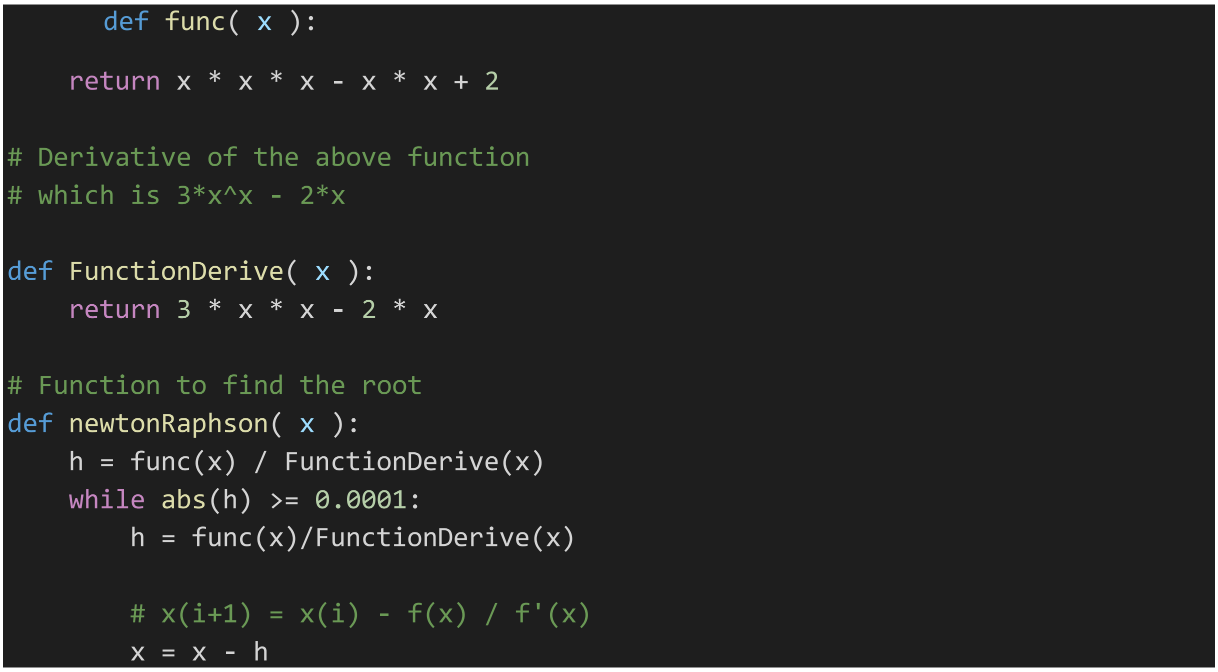
Output:



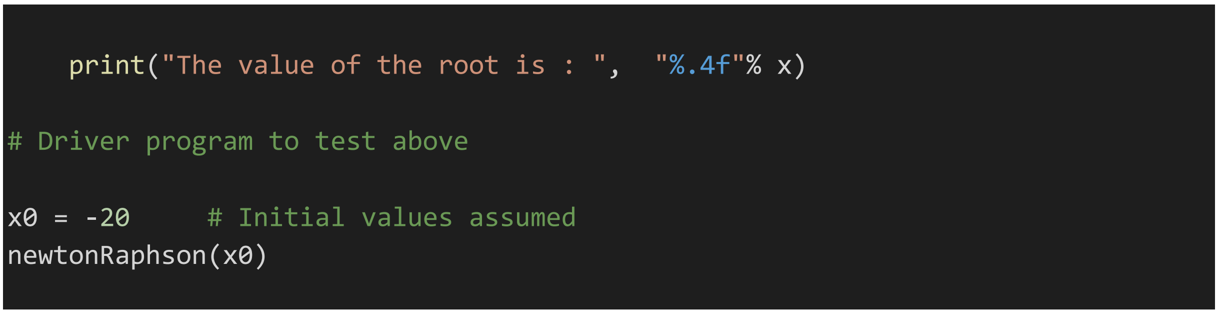
```
[2. 3.]
```

18.Use Newton-Raphson method to solve given equation

The Newton-Raphson method (also known as Newton's method) is a way to quickly find a good approximation for the root of a real-valued function. It uses the idea that a continuous and differentiable function can be approximated by a straight-line tangent to it.

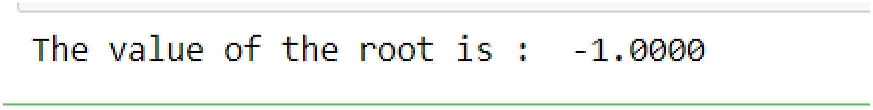


```
      def func( x ):

    return x * x * x - x * x + 2

# Derivative of the above function
# which is 3*x^x - 2*x

def FunctionDerive( x ):
    return 3 * x * x - 2 * x

# Function to find the root
def newtonRaphson( x ):
    h = func(x) / FunctionDerive(x)
    while abs(h) >= 0.0001:
        h = func(x)/FunctionDerive(x)

        # x(i+1) = x(i) - f(x) / f'(x)
        x = x - h
```



```
    print("The value of the root is : ",  "%.4f"% x)

# Driver program to test above

x0 = -20     # Initial values assumed
newtonRaphson(x0)
```

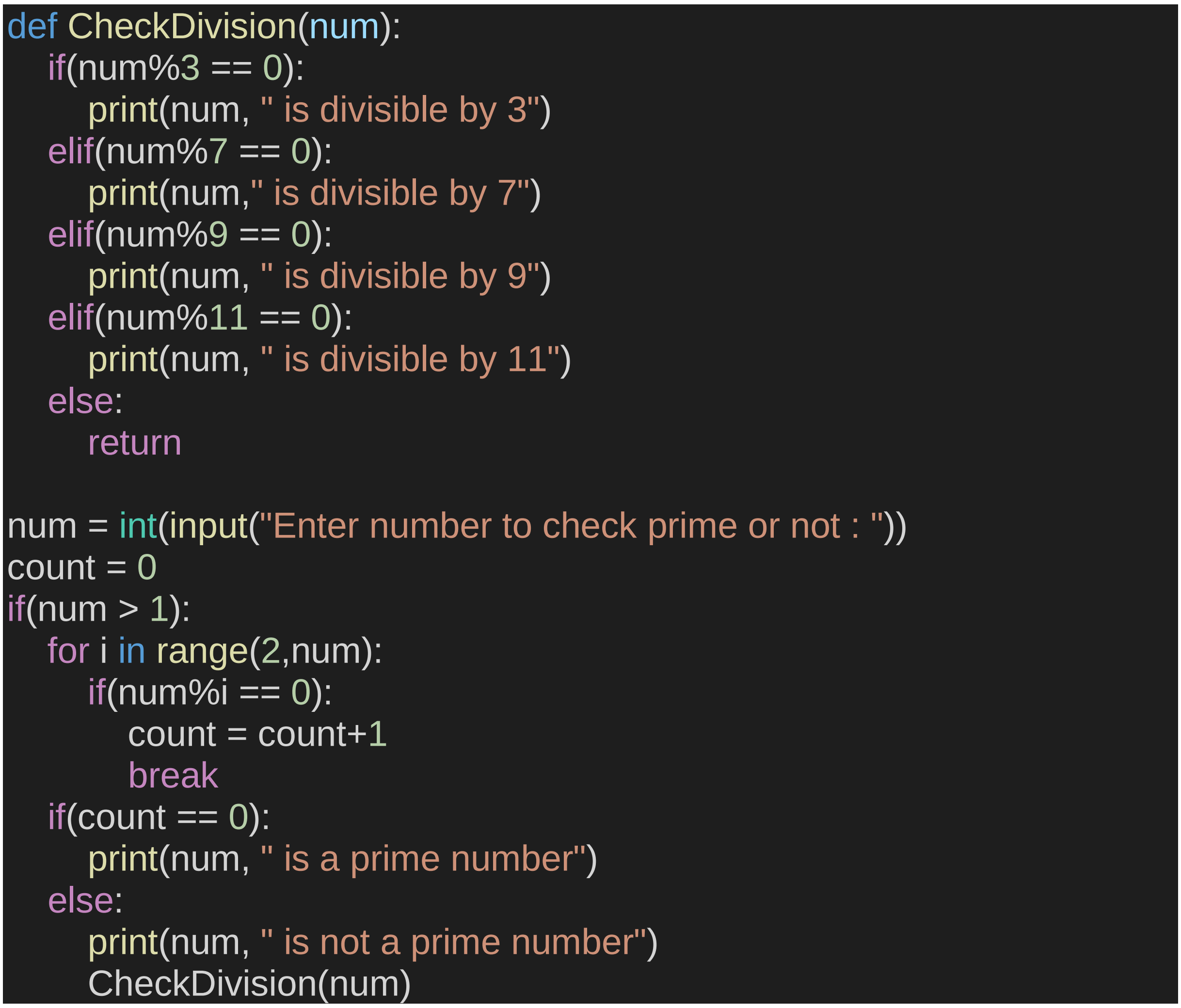
Output:



```
The value of the root is :  -1.0000
```

20.Write a code to check whether a given number is prime or not,

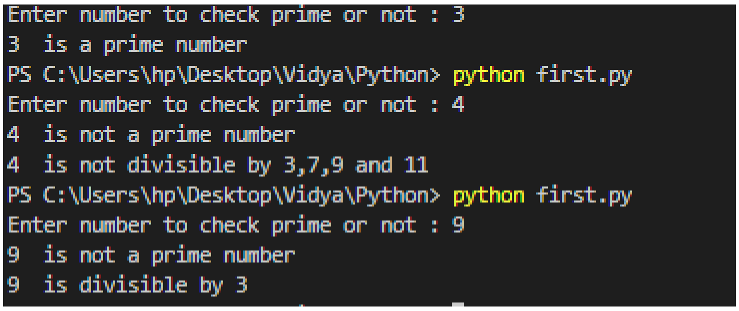
If not prime, whether it is divisible by 3,7,9,11?



```
def CheckDivision(num):
    if(num%3 == 0):
        print(num, " is divisible by 3")
    elif(num%7 == 0):
        print(num," is divisible by 7")
    elif(num%9 == 0):
        print(num, " is divisible by 9")
    elif(num%11 == 0):
        print(num, " is divisible by 11")
    else:
        return

num = int(input("Enter number to check prime or not : "))
count = 0
if(num > 1):
    for i in range(2,num):
        if(num%i == 0):
            count = count+1
            break
    if(count == 0):
        print(num, " is a prime number")
    else:
        print(num, " is not a prime number")
        CheckDivision(num)
```

```
else:
    print("1 or negative numbers are not fall into prime number category ")
```

Output:

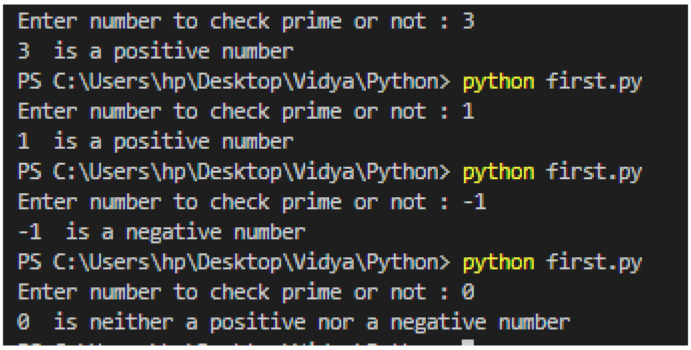


```
Enter number to check prime or not : 3
3  is a prime number
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter number to check prime or not : 4
4  is not a prime number
4  is not divisible by 3,7,9 and 11
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter number to check prime or not : 9
9  is not a prime number
9  is divisible by 3
```

21. Write a Python program to check if a number is positive or negative.

```
num = int(input("Enter number to check prime or not : "))

if(num > 0):
    print(num, " is a positive number")
elif(num < 0):
    print(num, " is a negative number")
else:
    print(num, " is neither a positive nor a negative number")
```

Output:

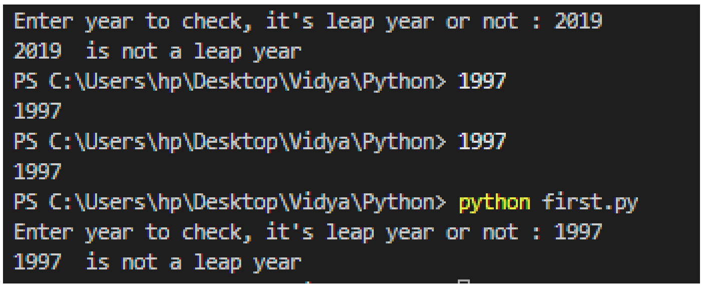


```
Enter number to check prime or not : 3
3  is a positive number
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter number to check prime or not : 1
1  is a positive number
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter number to check prime or not : -1
-1  is a negative number
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter number to check prime or not : 0
0  is neither a positive nor a negative number
```

22. Write a python program to check leap year

```
year = int(input("Enter year to check, it's leap year or not : "))
if(year%4 == 0):
    if(year%100 == 0):
        if(year%400 == 0):
            print(year, " is a leap year ")
        else:
            print(year, " is not a leap year ")
    else:
        print(year, " is a leap year ")
else:
    print(year, " is not a leap year ")
```

Output :

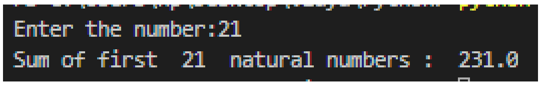


```
Enter year to check, it's leap year or not : 2019
2019  is not a leap year
PS C:\Users\hp\Desktop\Vidya\Python> 1997
1997
PS C:\Users\hp\Desktop\Vidya\Python> 1997
1997
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter year to check, it's leap year or not : 1997
1997  is not a leap year
```

23. Write a Python Program to Read a Number n And Print the Series "1+2+…..+n= "

```
num = int(input("Enter the number:"))
sum = 0
sum = num * (num+1)/2
print("Sum of first ",num," natural numbers : ",sum)
```

Output:



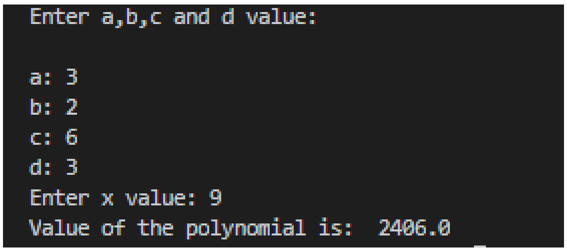
```
Enter the number:21
Sum of first  21  natural numbers :  231.0
```

24. Write a Program to Compute a Polynomial Equation given that the Coefficients of the Polynomial are stored in a List

```
print("Enter the coefficients of the form ax^3 + bx^2 + cx + d: ")
list = []
```

```
coefficients = ['a: ','b: ','c: ','d: ']
print("Enter a,b,c and d value:\n ")
for i in range(4):
    value = float(input(coefficients[i]))
    #Store coefficients in a list
    list.append(value)

#Value for x
x = float(input("Enter x value: "))

sum = 0
exp = 4
#Evaluate the equation
for i in range(0,4):
    exp = exp - 1
    sum = sum + (x**exp) * list[i]
print("Value of the polynomial is: ",sum)
```

Output:

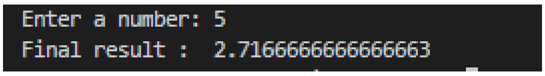


```
Enter a,b,c and d value:

a: 3
b: 2
c: 6
d: 3
Enter x value: 9
Value of the polynomial is:  2406.0
```

25. Write a Python Program to Compute the Value of Euler's Number e. Use the Formula: e = 1 + 1/1! + 1/2! + …… 1/n!

```
def Factorial(n):
    fact = 1
    for i in range(n,0,-1):
        fact = fact * i
    return fact

num = int(input("Enter a number: "))
sum = 1
for n in range(1,num+1):
    sum = sum + (1/Factorial(n))
print(sum)
```

Output:



```
Enter a number: 5
Final result :  2.7166666666666663
```

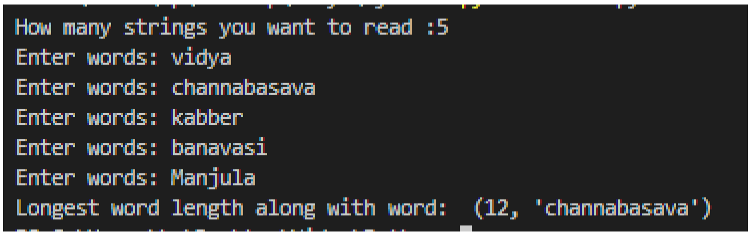
26. Write a Python Program to Convert Binary to Gray Code

```
num = int(input("Enter a binary number: "))
def storeEachDigit(num):
    count = 0
    list = []
    while(num):
        num = int(num)
        rem = num%10
        #Store each digit of entered array
        list.append(rem)
        num = num/10
        num = int(num)
        #Reverse the list while returning
    return list[::-1]

def BinaryToGray():
    #Calling storeEachDigt method
    stored_elmts = storeEachDigit(num)
    result = []

    for i in range(0, len(stored_elmts)):
        print(stored_elmts[i])

        if(i==0):
            result.append(stored_elmts[i])
        else:
            result.append(stored_elmts[i-1] ^ stored_elmts[i])
    return result
result= BinaryToGray()

#Display gray code
print("Gray code of binary value ",num," is: ")
for i in result:
    print(i,end="")
```

27.Write a Python Program to Read a List of Words and Return the Length of the Longest One.

```
no_of_strings = int(input("How many strings you want to read :"))
Words = []
for i in range(no_of_strings):
    name = input("Enter words: ")
    Words.append(name)
#print(Words)
def LogestLenWord():
    max_len = 0
    for word in Words:
        if(len(word) > max_len):
            max_len = len(word)
            max_len_word = word
    return max_len,max_len_word

print("Longest word length along with word: ",LogestLenWord())
```

Output:

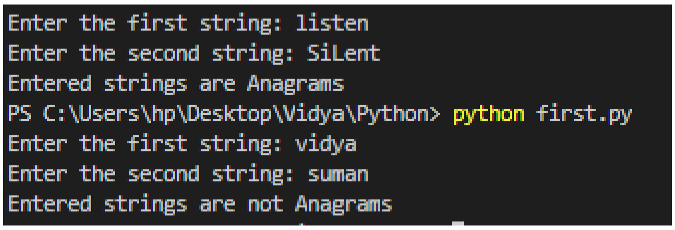


```
How many strings you want to read :5
Enter words: vidya
Enter words: channabasava
Enter words: kabber
Enter words: banavasi
Enter words: Manjula
Longest word length along with word:  (12, 'channabasava')
```

28.Write a Python Program to detect if Two Strings are Anagrams

```
#Write a Python Program to detect if Two Strings are Anagrams
#Anagrams => An anagram of a string is another string that contains the same characters.Ex : listen and silent
string1 = input("Enter the first string: ")
string2 = input("Enter the second string: ")

def checkAnagrams(string1,string2):
    mainCounter = 0
    for fchar in string1:
        charCounter = 0
        for schar in string2:
            if(fchar.upper() == schar.upper()):
                #if char found in second string no need to do further verification
                charCounter = charCounter+1
                break
        if(charCounter != 1):
            break
        else:
            mainCounter = mainCounter +charCounter
    return mainCounter

if(len(string1) == len(string2)):
    result = checkAnagrams(string1,string2)
    if(result == len(string1)):
        print("Entered strings are Anagrams")
    else:
         print("Entered strings are Anagrams")
else:
    print("Entered strings are not Anagrams")
```

Output :

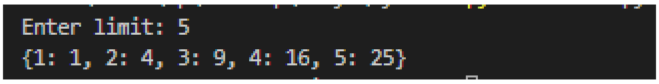


```
Enter the first string: listen
Enter the second string: SiLent
Entered strings are Anagrams
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter the first string: vidya
Enter the second string: suman
Entered strings are not Anagrams
```

29.Write a Python Program to Generate a Dictionary that Contains Numbers (between 1 and n) in the Form (x,x*x).

```
#Write a Python Program to Generate a Dictionary that Contains Numbers (between 1 and n) in the Form (x,x*x).
num = int(input("Enter limit: "))
dict = {}
for i in range(1,num+1):
    key_value = {i : i*i}
    dict.update(key_value)
print(dict)
```

Output:

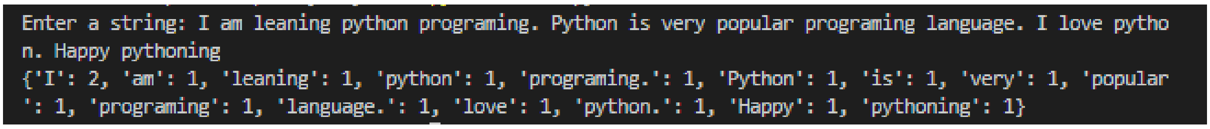


```
Enter limit: 5
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

30. Write a Python Program to Count the Frequency of Words Appearing in a String Using a Dictionary.

```
#Write a Python Program to Count the Frequency of Words Appearing in a String Using a Dictionary
test_string = input("Enter a string: ")
list = []
list = test_string.split()
wordfreq = [list.count(p) for p in list]
print(dict(zip(list,wordfreq)))
```

Output:

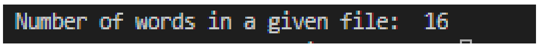


```
Enter a string: I am leaning python programing. Python is very popular programing language. I love pytho
n. Happy pythoning
{'I': 2, 'am': 1, 'leaning': 1, 'python': 1, 'programing.': 1, 'Python': 1, 'is': 1, 'very': 1, 'popular
': 1, 'programing': 1, 'language.': 1, 'love': 1, 'python.': 1, 'Happy': 1, 'pythoning': 1}
```

31. Write a Python Program to Count the Number of Words in a Text File.

```
#Write a Python Program to Count the Number of Words in a Text File
try:
    word_count = 0
    f = open("test.txt","r")
    for line in f:
        words = line.split()
        word_count += len(words)
    print("Number of words in a given file: ",word_count)
except IOError:
    print("File not accessible")
finally:
    f.close()
```

Output:

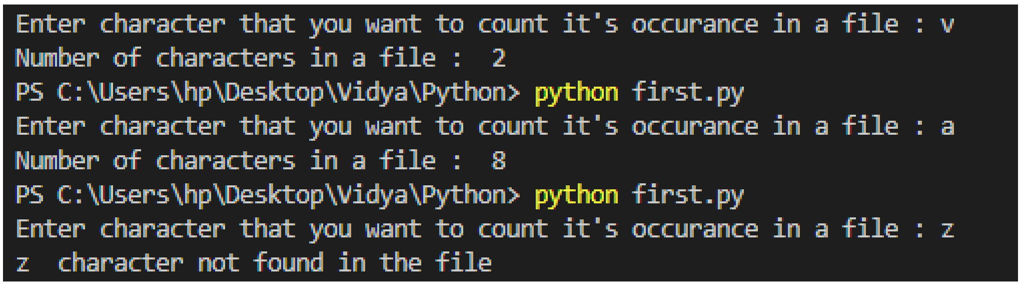


```
Number of words in a given file:  16
```

32. Write a Python Program that Reads a Text File and Counts the Number of Times a Certain Letter Appears in the Text File.

```
#Write a Python Program to Count the Number of Words in a Text File
try:
    char_count = 0
    find_char = input("Enter character that you want to count it's occurance in a file : ")
    f = open("test.txt","r")
    while 1:
        char = f.read(1)
        if not char:
            break
        else:
            if(find_char == char):
                char_count = char_count + 1
    if(char_count > 0):
        print("Number of characters in a file : ",char_count)
    else:
        print(find_char," character not found in the file")
except IOError:
    print("File not accessible")
finally:
    f.close()
```

Output:



```
Enter character that you want to count it's occurance in a file : v
Number of characters in a file :  2
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter character that you want to count it's occurance in a file : a
Number of characters in a file :  8
PS C:\Users\hp\Desktop\Vidya\Python> python first.py
Enter character that you want to count it's occurance in a file : z
z  character not found in the file
```

33. Write a Python Program to Implement Stack using One Queue

    (Out of syllabus)

34. Write a Python Program to Check String is Palindrome using Stack

    (Out of syllabus)

35. Write Notes on the following points

- **What Is a Python NumPy Array?**
  NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.

- **NumPy Arrays v/s List**

  Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays. A numpy array is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the *rank* of the array; the *shape* of an array is a tuple of integers giving the size of the array along each dimension.
  The Python core library provided

  Lists. A list is the Python equivalent of an array, but is resizeable and can contain elements of different types.
  A common beginner question is what is the real difference here. The answer is performance.

  Numpy data structures perform better in:

  - **Size** - Numpy data structures take up less space
  - **Performance** - they have a need for speed and are faster than lists
  - **Functionality** - SciPy and NumPy have optimized functions such as linear algebra operations built in.

  Memory
  The main benefits of using NumPy arrays should be smaller memory consumption and better runtime behavior.

For Python Lists - We can conclude from this that for every new element, we need another eight bytes for the reference to the new object.

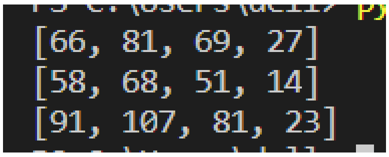
- **NumPy Operations**
  Arithmetic Operations. Input arrays for performing arithmetic operations such as add(), subtract(), multiply(), and divide()

- **NumPy Special Functions**
  The main feature of the scipy. special package is the definition of numerous special functions of mathematical physics. Available functions include airy, elliptic, bessel, gamma, beta, hypergeometric, parabolic cylinder, mathieu, spheroidal wave, struve, and kelvin. There are also some low-level stats functions that are not intended for general use as an easier interface to these functions is provided by the stats module. Most of these functions can take array arguments and return array results following the same broadcasting rules as other math functions in Numerical Python. Many of these functions also accept complex numbers as input.

36. Write a python program for Matrix Multiplication Using Nested List Comprehension and using Using Nested Loop

```
X = [[12,7,3],
   [4 ,5,6],
   [7 ,8,9]]
# 3x4 matrix
Y = [[5,8,1,2],
   [6,7,3,0],
   [4,5,9,1]]
# result is 3x4
result = [[0,0,0,0],
        [0,0,0,0],
        [0,0,0,0]]
# iterate through rows of X
for i in range(len(X)):
  # iterate through columns of Y
  for j in range(len(Y[0])):
      # iterate through rows of Y
      for k in range(len(Y)):
          result[i][j] += X[i][k] * Y[k][j]

#output is
for r in result:
  print(r)
```

Output:

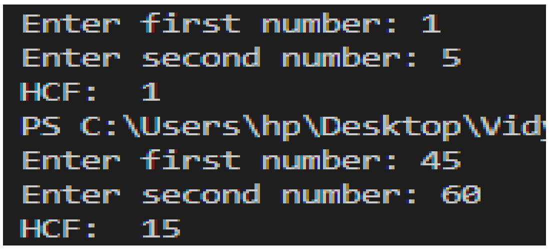


```
[66, 81, 69, 27]
[58, 68, 51, 14]
[91, 107, 81, 23]
```

37. Write a python program to find the H.C.F of two input number using loops.

```
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))
divider = 1
while 1:
    if(num1%divider == 0 and num2%divider==0):
        hcf = divider
    if(divider == num1 or divider == num2):
        break
    divider = divider +1
print("HCF: ",hcf)
```

Output:



```
Enter first number: 1
Enter second number: 5
HCF:  1
PS C:\Users\hp\Desktop\Vid
Enter first number: 45
Enter second number: 60
HCF:  15
```

38. Answer the following

- What is SCipy?
  SciPyis a free and open-source Python library used for scientific computing and technical computing.
- Difference between Scipy and Numpy?

NumPy	SciPy
NumPy is a pretty generic library for fast matrix computations.	SciPy is a set of libraries for scientific computing.
NumPy deals with array manipulation and basic linear algebra, Fourier transform, and random number capabilities	Mainly SciPy is used to perform complex linear algebra, ODE, Calculus

NumPy:

- Numpy is written in C and use for mathematical or numeric calculation.
- It is faster than other Python Libraries
- Numpy is the most useful library for Data Science to perform basic calculations.
- Numpy contains nothing but array data type which performs the most basic operation like sorting, shaping, indexing, etc.

SciPy:

- SciPy is built in top of the NumPy
- SciPy is a fully-featured version of Linear Algebra while Numpy contains only a few features.
- Most new Data Science features are available in Scipy rather than Numpy.

- Why use SciPy
  - SciPy contains varieties of sub packages which help to solve the most common issue related to Scientific Computation.
  - SciPy is the most used Scientific library only second to GNU Scientific Library for C/C++ or Matlab's.
  - Easy to use and understand as well as fast computational power.
  - It can operate on an array of NumPy library.

39. Write a python program to read a image and flip it using scipy library.

(out of syllabus)

40. Perform single integration using Scipy li Brary

$$\int_a^b f(x)dx$$

```
import scipy.integrate

from numpy import exp

f= lambda x:exp(-x**2)

i = scipy.integrate.quad(f, 0, 1)

print(i)
```

Output:

```
(0.7468241328124271, 8.291413475940725e-15)
```