PRACTICLE : 01

AIM : Install and configure the Python environment. Run basic Python commands to verify the Python environment.

PYTHON SETUP

Python is a widely used high-level programming language. It is one of the most popular and flexible server-side programming languages.

Windows does not have the Python programming language installed by default. However, you can install Python on Windows in just a few easy steps.

This guide provides step-by-step instructions to install and set up Python on Windows.

Python Installation on Windows

The installation requires downloading the official Python *.exe* installer and running it on your system. The sections below will explain several options and details during the installation process.

Step 1: Select Python Version

Deciding on a version depends on what you want to do in Python. The two major versions are Python 2 and Python 3. Choosing one over the other might be better depending on your project details. If there are no constraints, choose whichever one you prefer.

We recommend Python 3, as Python 2 reached its end of life in 2020. Download Python 2 only if you work with legacy scripts and older projects. Also, choose a stable release over the newest since the newest release may have bugs and issues.

Step 2: Download Python Executable Installer

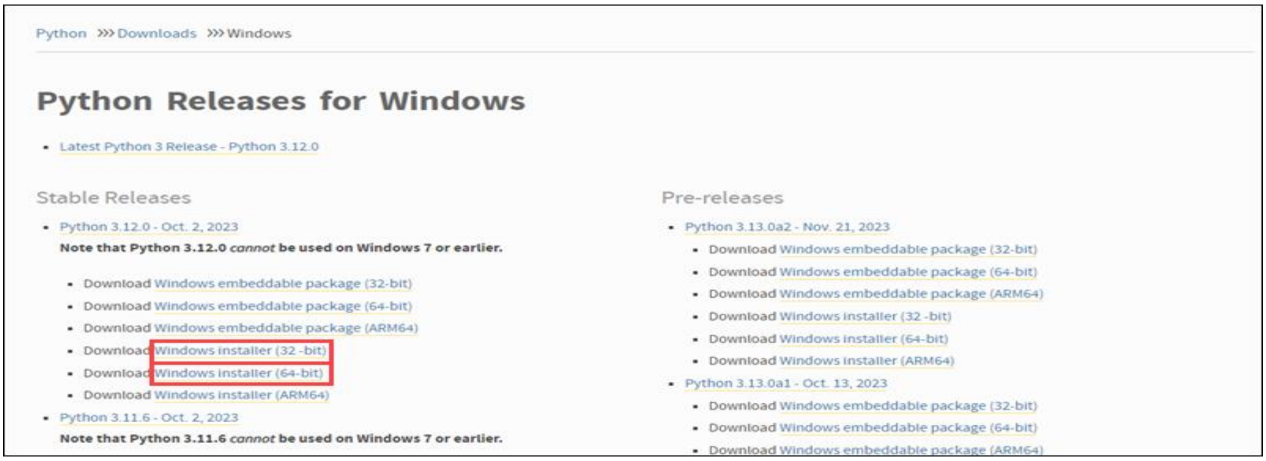
Start by downloading the Python executable installer for Windows:

1. Open a web browser and navigate to the Downloads for Windows section of the official Python website.

2. Locate the desired Python version.

3. Click the link to download the file. Choose either the Windows 32-bit or 64-bit installer.

The download is approximately 25MB.



Python >>> Downloads >>> Windows

Python Releases for Windows

- Latest Python 3 Release - Python 3.12.0

Stable Releases

- Python 3.12.0 - Oct. 2, 2023
  Note that Python 3.12.0 *cannot* be used on Windows 7 or earlier.
  - Download Windows embeddable package (32-bit)
  - Download Windows embeddable package (64-bit)
  - Download Windows embeddable package (ARM64)
  - Download Windows installer (32 -bit)
  - Download Windows installer (64-bit)
  - Download Windows installer (ARM64)
- Python 3.11.6 - Oct. 2, 2023
  Note that Python 3.11.6 *cannot* be used on Windows 7 or earlier.

Pre-releases

- Python 3.13.0a2 - Nov. 21, 2023
  - Download Windows embeddable package (32-bit)
  - Download Windows embeddable package (64-bit)
  - Download Windows embeddable package (ARM64)
  - Download Windows installer (32 -bit)
  - Download Windows installer (64-bit)
  - Download Windows installer (ARM64)
- Python 3.13.0a1 - Oct. 13, 2023
  - Download Windows embeddable package (32-bit)
  - Download Windows embeddable package (64-bit)
  - Download Windows embeddable package (ARM64)

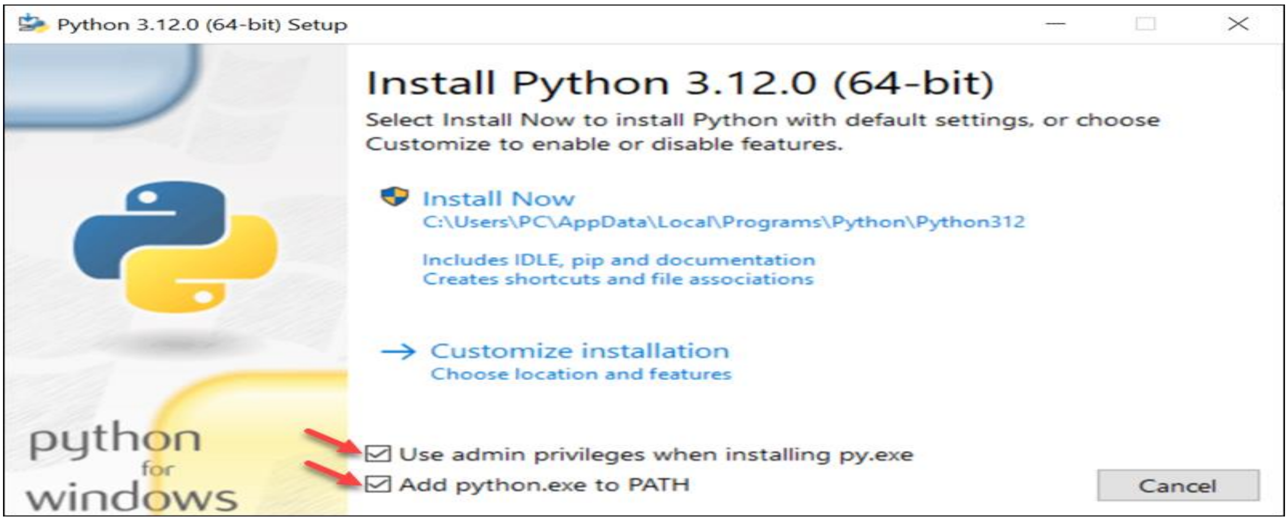
Step 3: Run Executable Installer

The steps below guide you through the installation process:

1. Run the downloaded **Python Installer**.

2. The installation window shows two checkboxes:

- **Admin privileges**. The parameter controls whether to install Python for the current or all system users. This option allows you to change the installation folder for Python.
- **Add Python to PATH**. The second option places the executable in the PATH variable after installation. You can also add Python to the PATH environment variable manually later.



Python 3.12.0 (64-bit) Setup

Install Python 3.12.0 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\PC\AppData\Local\Programs\Python\Python312

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Use admin privileges when installing py.exe
☑ Add python.exe to PATH

For the most straightforward installation, we recommend ticking both checkboxes.

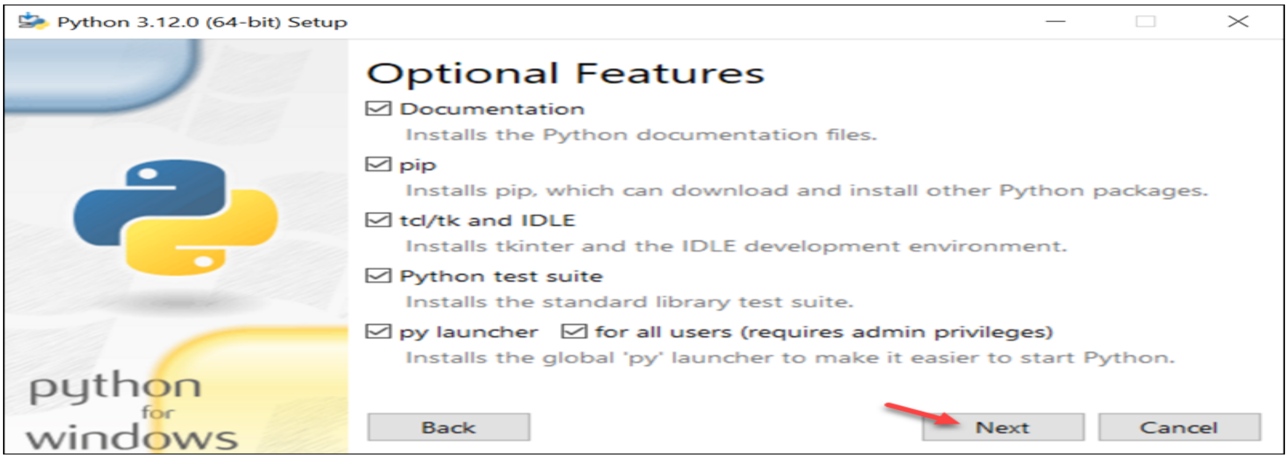
3. Select the **Install Now** option for the recommended installation (in that case, skip the next two steps).

To adjust the default installation options, choose **Customize installation** instead and proceed to the following step.

The default installation installs Python to *C:\Users\[user]\AppData\Local\Programs\Python\Python[version]* for the current user. It includes IDLE (the default Python editor), the PIP package manager, and additional documentation. The installer also creates necessary shortcuts and file associations.

Customizing the installation allows changing these installation options and parameters.

4. Choose the optional installation features. Python works without these features, but adding them improves the program's usability.



Python 3.12.0 (64-bit) Setup

Optional Features

☑ Documentation
Installs the Python documentation files.

☑ pip
Installs pip, which can download and install other Python packages.

☑ tcl/tk and IDLE
Installs tkinter and the IDLE development environment.

☑ Python test suite
Installs the standard library test suite.

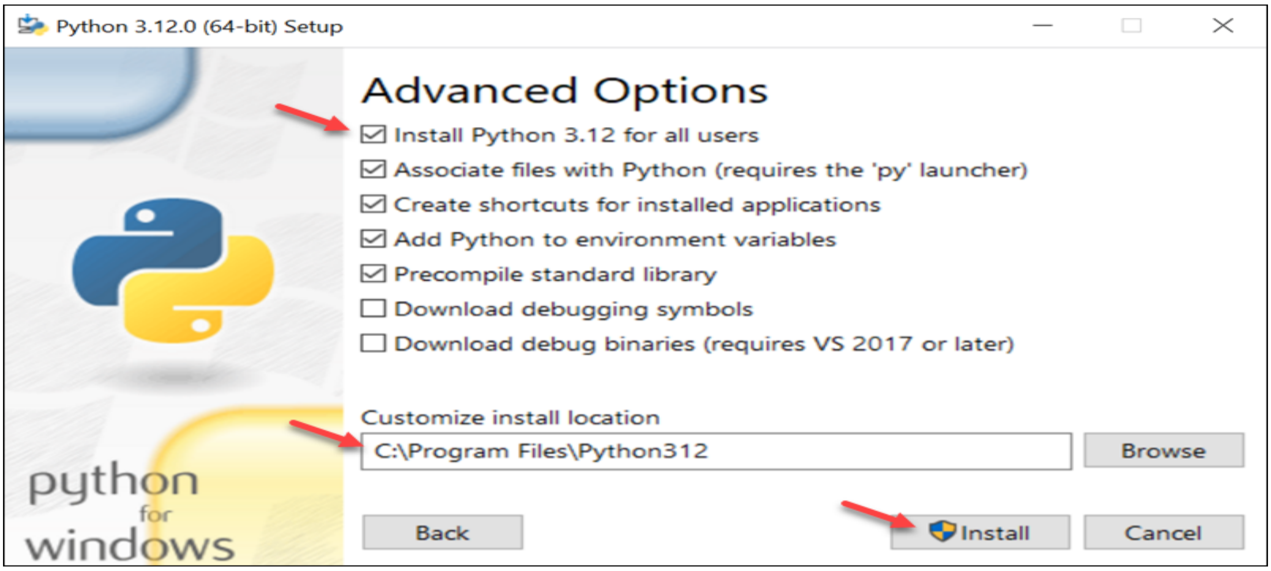
☑ py launcher ☑ for all users (requires admin privileges)
Installs the global 'py' launcher to make it easier to start Python.

Click **Next** to proceed to the Advanced Options screen.

5. The second part of customizing the installation includes advanced options.

Choose whether to install Python for all users. The option changes the install location to *C:\Program Files\Python[version]*. If selecting the location manually, a common choice is *C:\Python[version]* because it avoids spaces in the path, and all users can access it. Due to administrative rights, both paths may cause issues during package installation.

Other advanced options include creating shortcuts, file associations, and adding Python to path.



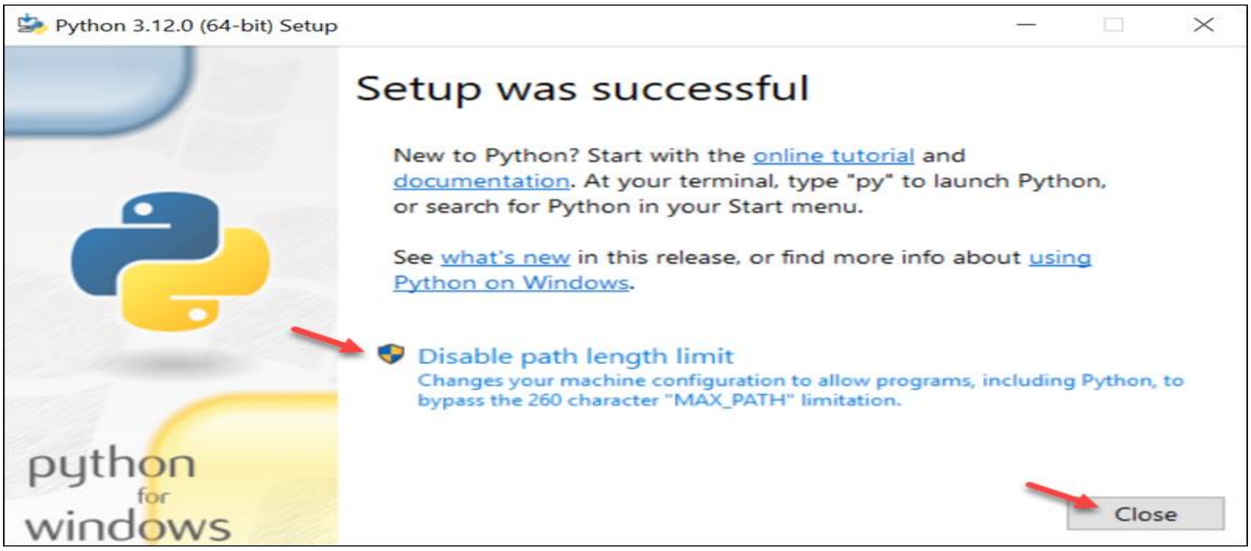
Advanced Options

☑ Install Python 3.12 for all users
☑ Associate files with Python (requires the 'py' launcher)
☑ Create shortcuts for installed applications
☑ Add Python to environment variables
☑ Precompile standard library
□ Download debugging symbols
□ Download debug binaries (requires VS 2017 or later)

Customize install location
C:\Program Files\Python312

After picking the appropriate options, click **Install** to start the installation.

6. Select whether to disable the path length limit. Choosing this option will allow Python to bypass the 260-character **MAX_PATH** limit.



Setup was successful

New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.

See what's new in this release, or find more info about using Python on Windows.

Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

The option will not affect any other system settings, and disabling it resolves potential name-length issues. We recommend selecting the option and closing the setup.

Verify Python Was Installed on Windows

The first way to verify that Python was installed successfully is through the command line. Open the command prompt and run the following command:

```
python --version
```

Home » DevOps and Development » How to Install Python on Windows

Introduction

Python is a widely used high-level programming language. It is one of the most popular and flexible server-side programming languages.

Windows does not have the Python programming language installed by default. However, you can install Python on Windows in just a few easy steps.

This guide provides step-by-step instructions to install and set up Python on Windows.

Python Installation on Windows

The installation requires downloading the official Python *.exe* installer and running it on your system. The sections below will explain several options and details during the installation process.

Step 1: Select Python Version

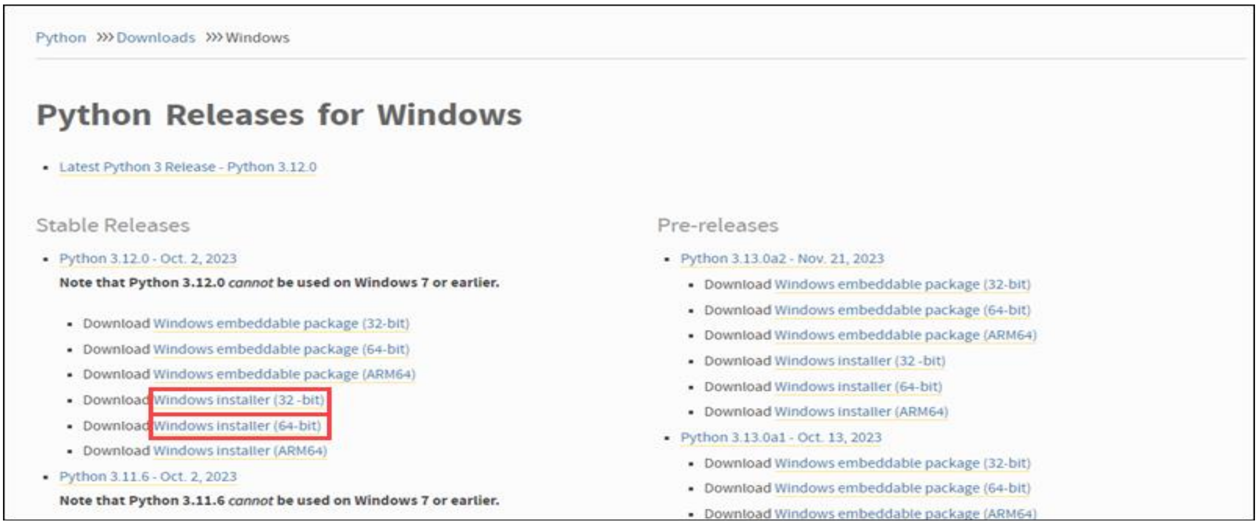
Deciding on a version depends on what you want to do in Python. The two major versions are Python 2 and Python 3. Choosing one over the other might be better depending on your project details. If there are no constraints, choose whichever one you prefer.

We recommend Python 3, as Python 2 reached its end of life in 2020. Download Python 2 only if you work with legacy scripts and older projects. Also, choose a stable release over the newest since the newest release may have bugs and issues.

Step 2: Download Python Executable Installer

Start by downloading the Python executable installer for Windows:

1. Open a web browser and navigate to the Downloads for Windows section of the official Python website.
2. Locate the desired Python version.



Python >>> Downloads >>> Windows

Python Releases for Windows

- Latest Python 3 Release - Python 3.12.0

Stable Releases

- Python 3.12.0 - Oct. 2, 2023
  Note that Python 3.12.0 *cannot* be used on Windows 7 or earlier.
  - Download Windows embeddable package (32-bit)
  - Download Windows embeddable package (64-bit)
  - Download Windows embeddable package (ARM64)
  - Download Windows installer (32 -bit)
  - Download Windows installer (64-bit)
  - Download Windows installer (ARM64)
- Python 3.11.6 - Oct. 2, 2023
  Note that Python 3.11.6 *cannot* be used on Windows 7 or earlier.

Pre-releases

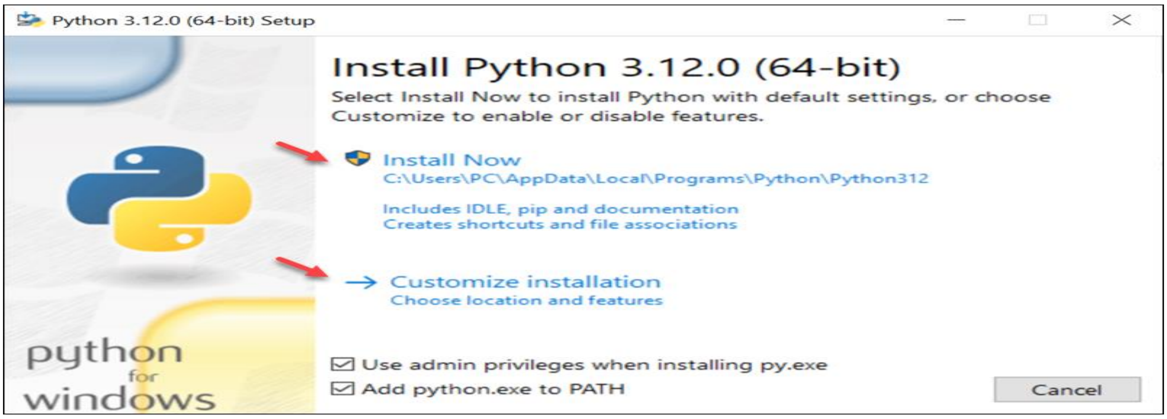
- Python 3.13.0a2 - Nov. 21, 2023
  - Download Windows embeddable package (32-bit)
  - Download Windows embeddable package (64-bit)
  - Download Windows embeddable package (ARM64)
  - Download Windows installer (32 -bit)
  - Download Windows installer (64-bit)
  - Download Windows installer (ARM64)
- Python 3.13.0a1 - Oct. 13, 2023
  - Download Windows embeddable package (32-bit)
  - Download Windows embeddable package (64-bit)
  - Download Windows embeddable package (ARM64)

3. Click the link to download the file. Choose either the Windows 32-bit or 64-bit installer. The download is approximately 25MB.

Step 3: Run Executable Installer

The steps below guide you through the installation process:
1. Run the downloaded **Python Installer**.
2. The installation window shows two checkboxes:

- **Admin privileges**. The parameter controls whether to install Python for the current or all system users. This option allows you to change the installation folder for Python.
- **Add Python to PATH**. The second option places the executable in the PATH variable after installation. You can also add Python to the PATH environment variable manually later.



Python 3.12.0 (64-bit) Setup

Install Python 3.12.0 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\PC\AppData\Local\Programs\Python\Python312

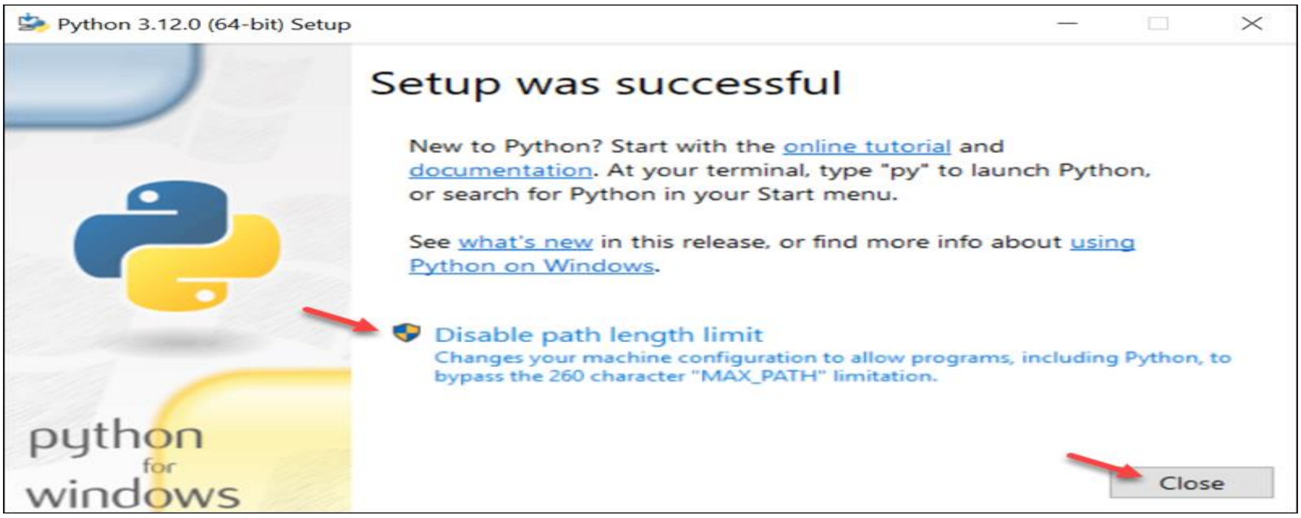
Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Use admin privileges when installing py.exe
☑ Add python.exe to PATH

For the most straightforward installation, we recommend ticking both checkboxes.
3. Select the **Install Now** option for the recommended installation (in that case, skip the next two steps).
To adjust the default installation options, choose **Customize installation** instead and proceed to the following step.

The default installation installs Python to *C:\Users\[user]\AppData\Local\Programs\Python\Python[version]* for the current user. It includes IDLE (the default Python editor), the PIP package manager, and additional documentation. The installer also creates necessary shortcuts and file associations.
After picking the appropriate options, click **Install** to start the installation.



Python 3.12.0 (64-bit) Setup

Setup was successful

New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.

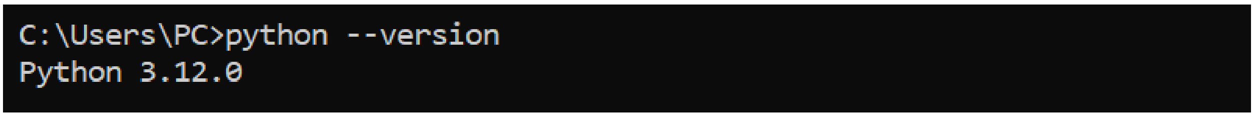
See what's new in this release, or find more info about using Python on Windows.

Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

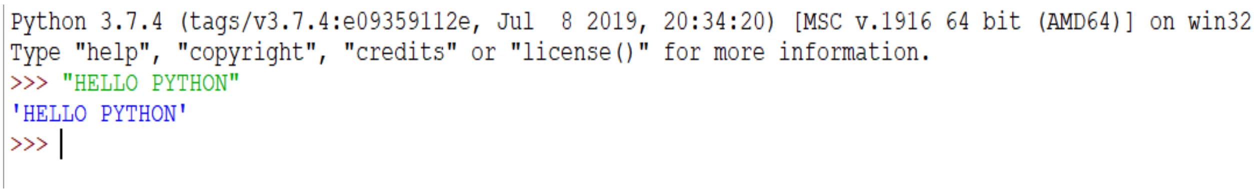
Step 5: Verify Python Was Installed on Windows

The first way to verify that Python was installed successfully is through the command line. Open the command prompt and run the following command:

```
python --version
```



```
C:\Users\PC>python --version
Python 3.12.0
```



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> "HELLO PYTHON"
'HELLO PYTHON'
>>> |
```

After installation is complete click on **Close**. Bingo..!! Python is installed. Now go to windows and type IDLE.

- This is Python Interpreter also called Python Shell. I printed Hello geeks, python is working smoothly.
- The three greater than >>> sign is called Python command prompt, where we write our program and with a single enter key, it will give result so instantly.

PRACTICAL 2

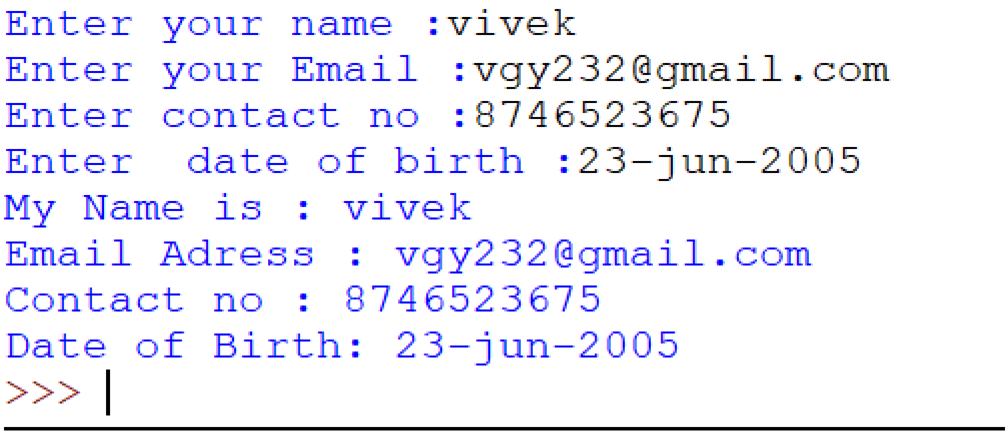
INPUT-OUTPUT

Aim:- Write a program to read your name, contact number, email, and birthdate and print those details on the screen.

INPUT:-

```
name=input("Enter your name :")
email=input("Enter your Email :")
contact=int(input("Enter contact no :"))
dob=int(input("Enter  date of birth :"))

print('My Name is :',name)
print('Email Adress :',email)
print('Contact no :',contact)
print('Date of Birth:',dob)
```

OUTPUT:-



```
Enter your name :vivek
Enter your Email :vgy232@gmail.com
Enter contact no :8746523675
Enter  date of birth :23-jun-2005
My Name is : vivek
Email Adress : vgy232@gmail.com
Contact no : 8746523675
Date of Birth: 23-jun-2005
>>> 
```

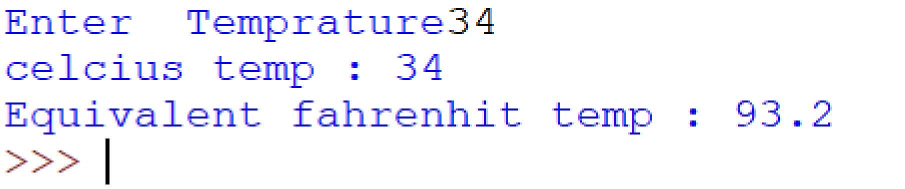
PRACTICLE : 3
Variables, operators, Expressions
PRACTICAL 3.1

AIM : Write a program to convert temperature from Celsius to Fahrenheit.

INPUT:-

```
c= int(input("Enter  Temprature"))
f=(c*9/5)+32
print("celcius temp :",c)
print("Equivalent fahrenhit temp :",f)
```

OUTPUT:-



```
Enter  Temprature34
celcius temp : 34
Equivalent fahrenhit temp : 93.2
>>> |
```

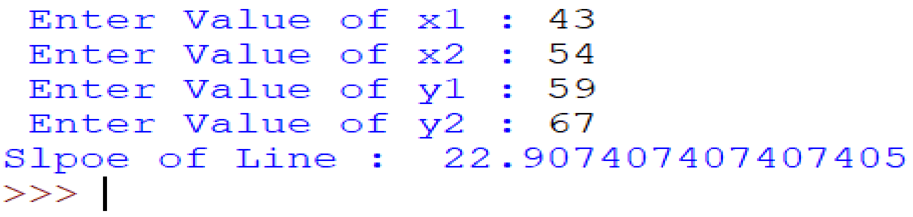
PRACTICAL 3.2

Aim:- Write a program to compute the slope of a line between two points (x1, y1) and (x2, y2)

INPUT:-

```
x1=int(input(" Enter Value of x1 : "))
x2=int(input(" Enter Value of x2 : "))
y1=int(input(" Enter Value of y1 : "))
y2=int(input(" Enter Value of y2 : "))
slope=(y2-y1/x2-x1)
print("Slpoe of Line : ",slope)
```

OUTPUT:-



```
 Enter Value of x1 : 43
 Enter Value of x2 : 54
 Enter Value of y1 : 59
 Enter Value of y2 : 67
Slpoe of Line :  22.907407407407405
>>> |
```

PRACTICLE : 3.3

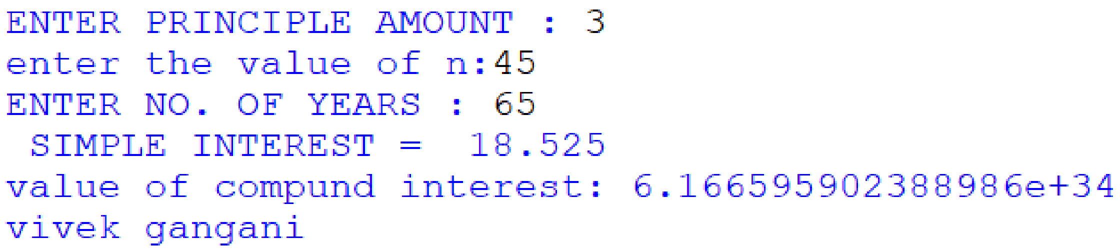
AIM : Write a program to calculate simple interest and compound interest.

$Simple\ Interest = P * R * T/100$

Input:

```
p=int(input("ENTER PRINCIPLE AMOUNT : "))
n=float(input("enter the value of n:"))
r=9.5
t=float(input("ENTER NO. OF YEARS : "))
si=p*r*t/100
print(" SIMPLE INTEREST = ",si)
ci=p*(1+r/100*n)**n*t
print("value of compund interest:",ci)
print(“vivek gangani”)
```

OUTPUT:-



```
ENTER PRINCIPLE AMOUNT : 3
enter the value of n:45
ENTER NO. OF YEARS : 65
 SIMPLE INTEREST =  18.525
value of compund interest: 6.166595902388986e+34
vivek gangani
```

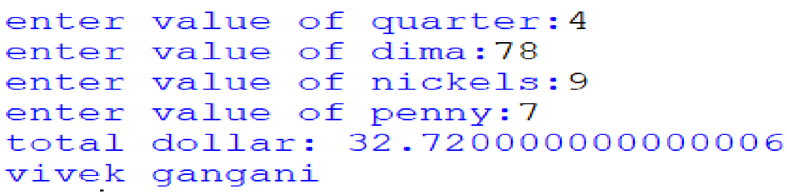
PRACTICLE : 3.4

AIM : Write a program to get change values in Quarter, Dime, Nickels and Pennies, and calculate the value of change in Dollars. Consider Quarter = 0.25 $, Dime = 0.10 $, Nickels = 0.05 $ and Penny = 0.01 $.

Input:

```
a=int(input("enter value of quarter:"))
b=int(input("enter value of dima:"))
c=int(input("enter value of nickels:"))
d=int(input("enter value of penny:"))
dollar=a*0.25+b*a*0.10+c*0.05+d*0.01
print("total dollar:",dollar)
print(“vivek gangani”)
```

Output:



```
enter value of quarter:4
enter value of dima:78
enter value of nickels:9
enter value of penny:7
total dollar: 32.720000000000006
vivek gangani
```

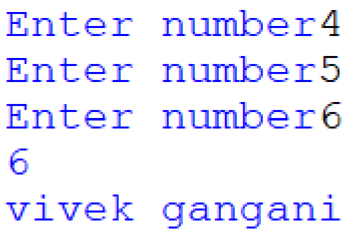
PRACTICLE : 3.5

AIM : Write a program to find a maximum of given three numbers (Use ternary operator).

Input:

```
a=int(input("Enter number"))
b=int(input("Enter number"))
c=int(input("Enter number"))
max=(a if a>b else b) if a>c else (b if b>c else c)
print(max)
print(“vivek gangani”)
```

Output:



```
Enter number4
Enter number5
Enter number6
6
vivek gangani
```

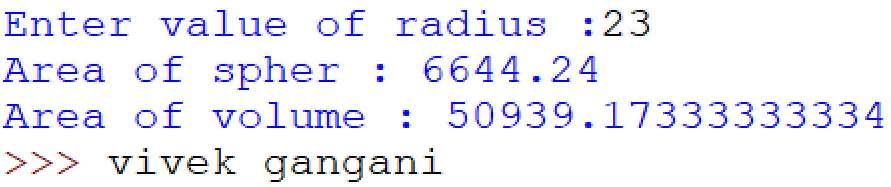
PRACTICLE : 3.6

AIM : Write a program to calculate area and volume of Sphere.

INPUT:-

```
r=float(input("Enter value of radius :"))

area=4*3.14*r*r
volume=(4/3)*3.14*r*r*r

print('Area of spher :',area)
print('Area of volume :',volume)
```

OUTPUT



```
Enter value of radius :23
Area of spher : 6644.24
Area of volume : 50939.17333333334
>>> vivek gangani
```

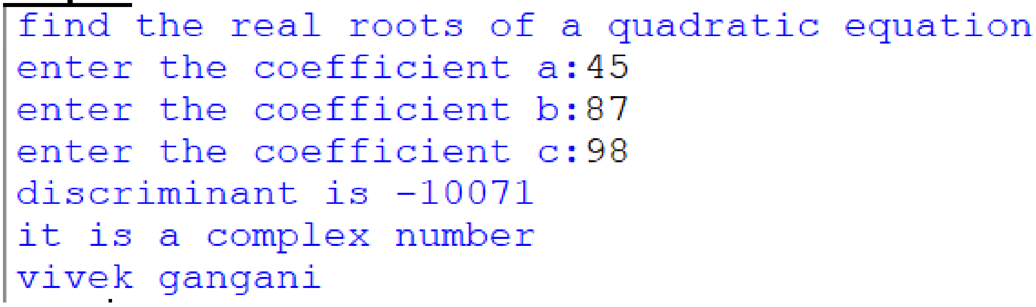
PRACTICLE : 3.7

AIM: :- Write a program that computes the real roots of a given quadratic equation (Use math library).

Input:

```
import math
print("find the real roots of a quadratic equation")
a=int(input("enter the coefficient a:"))
b=int(input("enter the coefficient b:"))
c=int(input("enter the coefficient c:"))
discriminant=b**2-4*a*c
print("discriminant is",discriminant)
if discriminant>0:
  root1= (-b+math.sqrt(discriminant))/2*a
  root2= (-b-math.sqrt(discriminant))/2*a
  print("root1",root1)
  print("root2",root2)
else:
  print("it is a complex number")
print("vivek gangani")
```

Output:



```
find the real roots of a quadratic equation
enter the coefficient a:45
enter the coefficient b:87
enter the coefficient c:98
discriminant is -10071
it is a complex number
vivek gangani
```

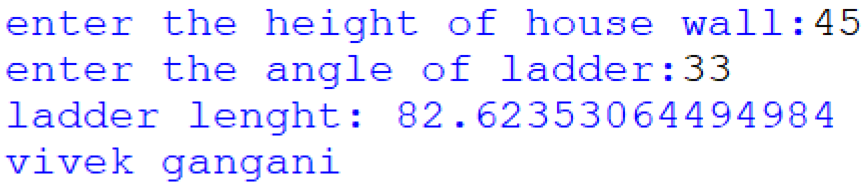
PRACTICLE : 3.8

AIM: Write a program to determine the length of ladder required to reach a given height when leaned against the house. The height and the angle of the ladder are given as inputs (Use math Library).

Input:

```
import math
height=int(input("enter the height of house wall:"))
angle=int(input("enter the angle of ladder:"))
ladder_length=height/math.sin(math.radians(angle))
print("ladder lenght:",ladder_length)
print("vivek gangani")
```

Output:



```
enter the height of house wall:45
enter the angle of ladder:33
ladder lenght: 82.62353064494984
vivek gangani
```

PRACTICLE : 4
Decision-Making Structures

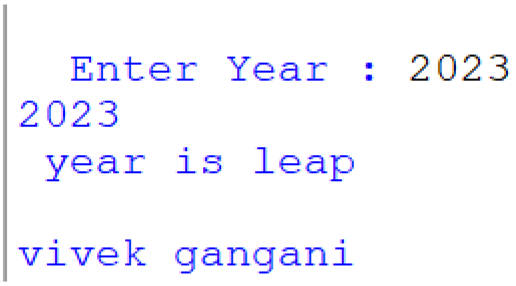
PRACTICLE : 4.1

AIM : Write A Program That Calculates Whether A Given Year Is A Leap Year Or Not

INPUT:-

```
year=int(input("\n  Enter Year : "))
if(year%4==0):
        if(year%100==0):
            if(year%400==0):
                print(year,"\n year is leap ")
            else:
                print(year,"\n year is not leap ")
        else:
            print(year,"\n year is leap ")
else:
            print(year,"\n year is leap \n")
          print("vivek gangani")
```

OUTPUT



```
  Enter Year : 2023
2023
 year is leap

vivek gangani
```

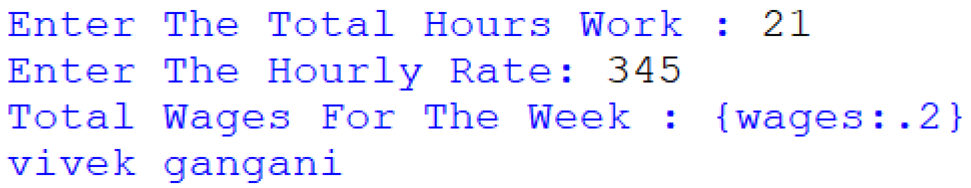
PRACTICLE : 4.2

AIM : Write a python program to input the number of hours worked and hourly rate and calculate the total wages for the week.

INPUT:-

```
work = float(input("Enter The Total Hours Work : "))
rate = float(input("Enter The Hourly Rate: "))

# Calculate total wages
wages = work * rate

# Display the result
print("Total Wages For The Week : {wages:.2}")

print("vivek gangani")
```

OUTPUT



```
Enter The Total Hours Work : 21
Enter The Hourly Rate: 345
Total Wages For The Week : {wages:.2}
vivek gangani
```

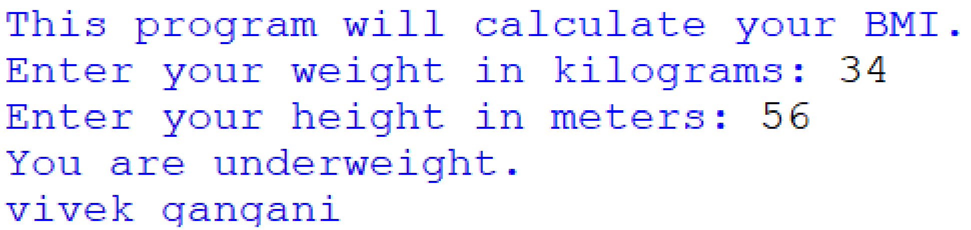
PRACTICLE : 4.3

AIM : Write a program to get a person's weight (kg) and height (cm) and display a message whether the person is healthy , underweight or overweight.

INPUT:-

```
print("This program will calculate your BMI.")
weight = float(input("Enter your weight in kilograms: "))
height = float(input("Enter your height in meters: "))
bmi = weight / (height ** 2)
if bmi < 19:
    print("You are underweight.")
elif bmi > 25:
    print("You are overweight.")
else:
    print("You are healthy.")
  print("vivek gangani")
```

OUTPUT



```
This program will calculate your BMI.
Enter your weight in kilograms: 34
Enter your height in meters: 56
You are underweight.
vivek gangani
```

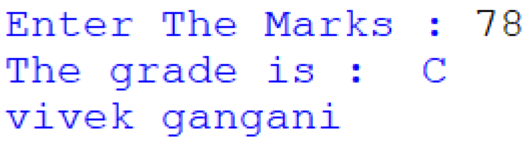
PRACTICLE : 4.4

AIM : Write a program to read the marks and assign a grade to a student.

INPUT:-

```
def grade(marks):
    if marks >= 90:
        return 'A'
    elif 80 <= marks <= 89:
        return 'B'
    elif 70 <= marks <= 79:
        return 'C'
    elif 60 <= marks <= 69:
        return 'D'
    elif 50 <= marks <= 59:
        return 'E'
    else:
        return 'F'
marks = float(input("Enter The Marks : "))
grade = grade(marks)
print("The grade is : ",grade)
print("vivek gangani")
```

OUTPUT



```
Enter The Marks : 78
The grade is :  C
vivek gangani
```

PRACTICLE : 5

Loops

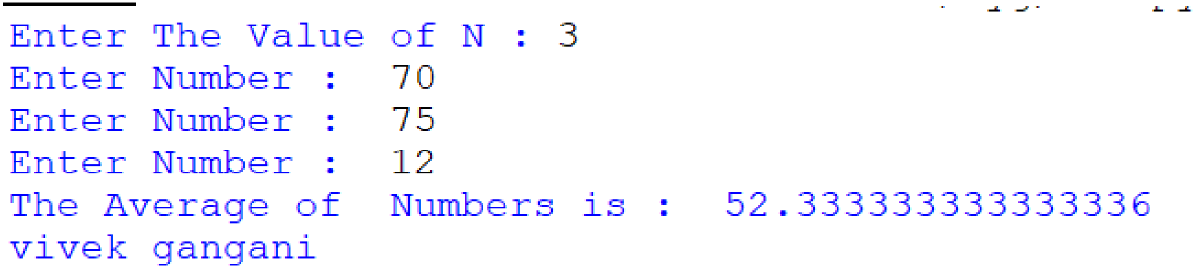
PRACTICLE : 5.1

AIM : Write a program to read n numbers from users and calculate the average of those n numbers.

INPUT:-

```
n = int(input("Enter The Value of N : "))
total = 0
for i in range(n):
    num = float(input("Enter Number :  "))
    total += num
# Calculate the average
avg = total / n
print("The Average of  Numbers is : ",avg)
print(“vivek gangani”)
```

OUTPUT



```
Enter The Value of N : 3
Enter Number :  70
Enter Number :  75
Enter Number :  12
The Average of  Numbers is :  52.333333333333336
vivek gangani
```

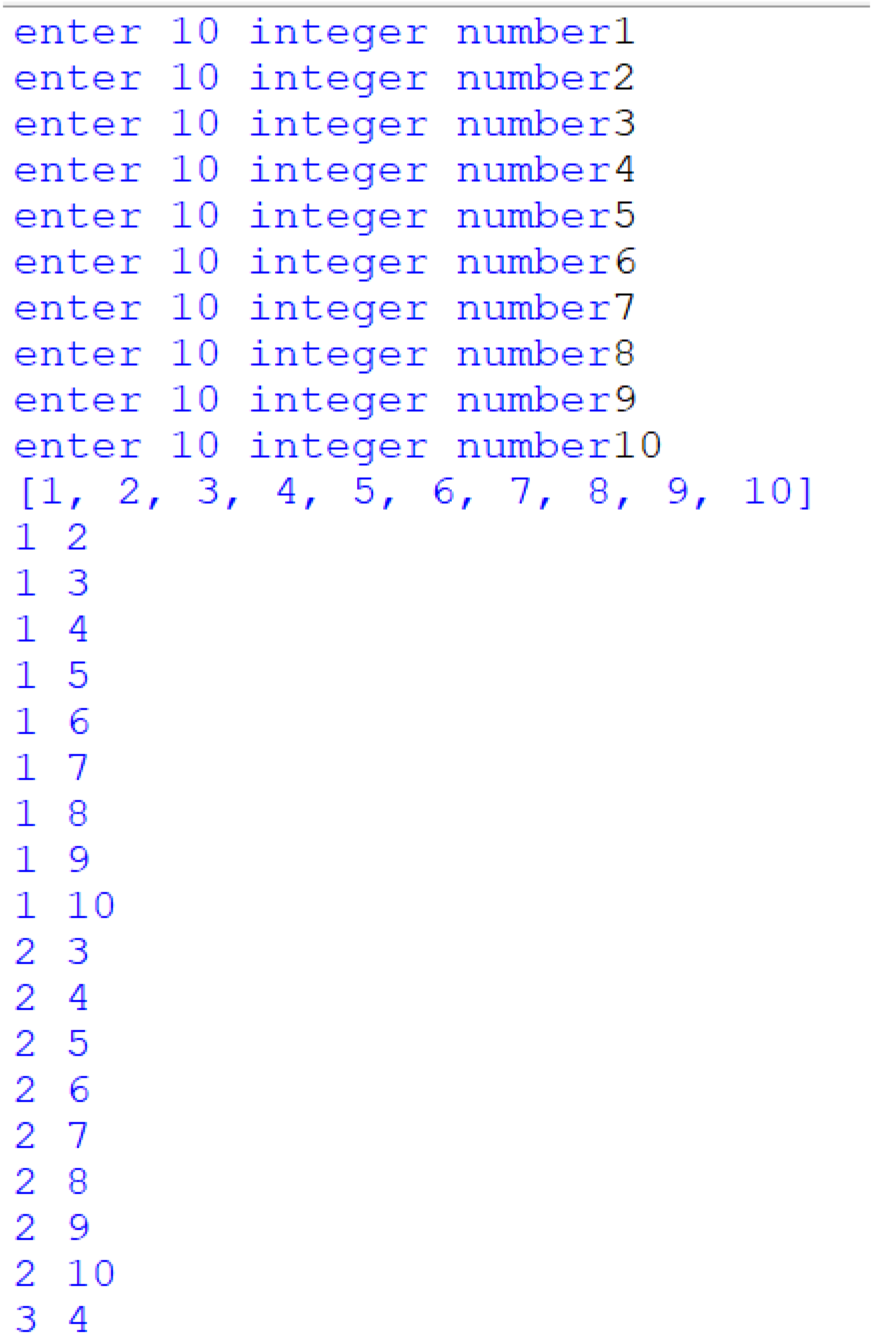
PRACTICLE : 5.2

AIM : Write a program that prompts the user to enter 10 integers and displays all the combinations of picking two numbers from the 10

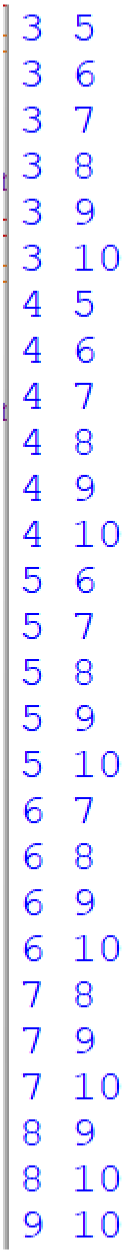
INPUT:-

```
                              numbers = []
#accept/prompt user to enter 10 integer numbes
for i in range(10):
   num=int(input("enter 10 integer number"))
   numbers.append(num)
print(numbers)
#display combination of two intger number
for i in range(10):
       for j in range(i+1,10):
            print(numbers[i],numbers[j])
```

OUTPUT



```
enter 10 integer number1
enter 10 integer number2
enter 10 integer number3
enter 10 integer number4
enter 10 integer number5
enter 10 integer number6
enter 10 integer number7
enter 10 integer number8
enter 10 integer number9
enter 10 integer number10
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
1 2
1 3
1 4
1 5
1 6
1 7
1 8
1 9
1 10
2 3
2 4
2 5
2 6
2 7
2 8
2 9
2 10
3 4
```



```
3 5
3 6
3 7
3 8
3 9
3 10
4 5
4 6
4 7
4 8
4 9
4 10
5 6
5 7
5 8
5 9
5 10
6 7
6 8
6 9
6 10
7 8
7 9
7 10
8 9
8 10
9 10
```

PRACTICLE : 5.3.1

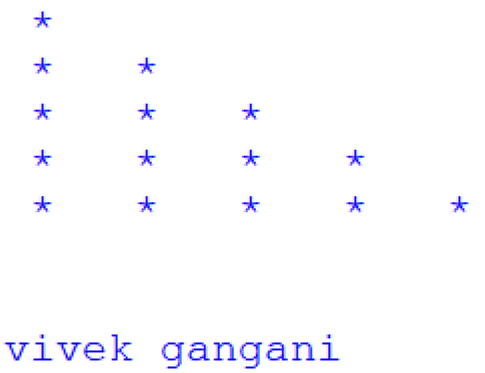
AIM : WRITE A PROGRAM TO PRINT THE FOLLOWING PATTERN

INPUT:-

```
*
*  *
*  *  *
*  *  *  *
*  *  *  *  *
```

```
i=1
while(i<=5):
        j=1
        while(j<=i):
                print(" * ",end=" ")
                j=j+1
        print(" ")
        i=i+1
print(" \n ")
print(“vivek gangani”)
```

<u>OUTPUT</u>



```
 *
 *    *
 *    *    *
 *    *    *    *
 *    *    *    *    *


vivek gangani
```

<u>PRACTICLE</u> : <u>5.3.2</u>

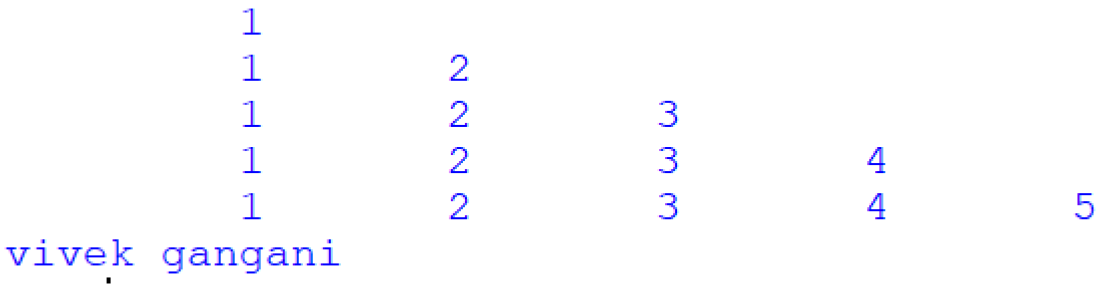
<u>AIM</u> : <u>WRITE A PROGRAM TO PRINT FOLLOWING PATTERN</u>

<u>INPUT:-</u>

1
1 2
1 2 3
1 2 3 4
1 2 3 4 5

```
rows = 5
# Outer loop to iterate through rows
for i in range(1, rows + 1):
   # Inner loop to iterate through numbers
   for j in range(1, i + 1):
      print('\t',j, end=" ")
     print()
   print(“vivek gangani”)
```

<u>OUTPUT</u>



```
        1
        1       2
        1       2       3
        1       2       3       4
        1       2       3       4       5
vivek gangani
```

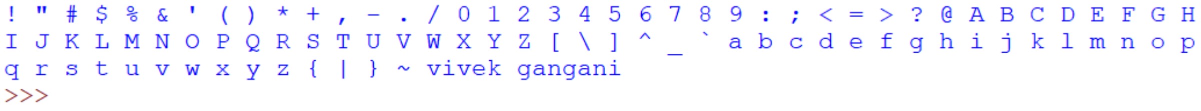
PRACTICLE : 5.4

AIM:Write a program that displays an ASCII character table from! to ~. Display the ASCII value of a character in decimal andhexadecimal. Display five characters per line

INPUT:-

```
for i in range(33,127):
     print(chr(i),end=" ")
   print(“vivek gangani”)
```

OUTPUT



```
! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G H
I J K L M N O P Q R S T U V W X Y Z [ \ ] ^ _ ` a b c d e f g h i j k l m n o p
q r s t u v w x y z { | } ~ vivek gangani
>>>
```

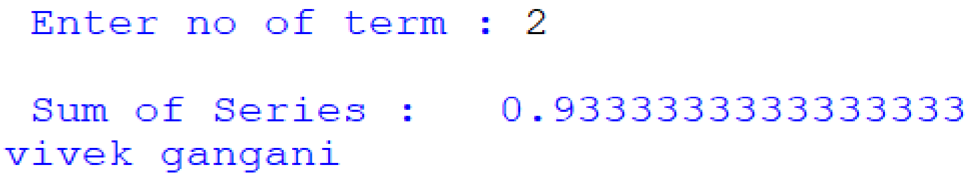
PRACTICLE : 5.5

AIM : Write A Program To Sum The Following Series 1/3 + 3/5 + 5/7 + 7/9 + 9/11 + 11/13 + ••••• + 95/97 + 97/99

INPUT:-

```
no_term=int(input("\n Enter no of term : "))
sum=0
numerator=1
denominator=3
for i in range(no_term):
            term=numerator/denominator
            sum=sum+term
            numerator=numerator+2
            denominator=denominator+2
print("\n Sum of Series :  ",sum)
print(“vivek gangani”)
```

OUTPUT



```
 Enter no of term : 2

 Sum of Series :   0.9333333333333333
vivek gangani
```

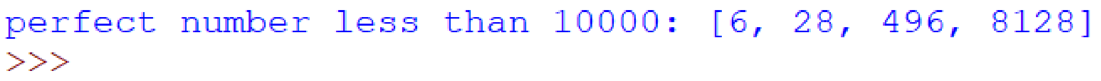
PRACTICLE : 5.6

AIM : if it is equal to the sum of all of its positive divisors, excluding itself. For example, 6 is the first perfect number, because 6 = 3 + 2 + 1, the next is 28 = 14 + 7 + 4 + 2 + 1. There are four perfect numbers that are less than 10,000. Write a program to find these four numbers.

Input:

```
perfect_num=[]
#loop from 2 to 10000
for num in range(2,10000):
    divisor_sum=1
    for divisor in range(2,num):
        if num%divisor==0:
            divisor_sum=divisor_sum+divisor
    if divisor_sum==num:
        perfect_num.append(num)
print("perfect number less than 10000:",perfect_num)
```

Output:



```
perfect number less than 10000: [6, 28, 496, 8128]
>>>
```

PRACTICLE:6
Lists, Tuples, Sets and Dictionaries

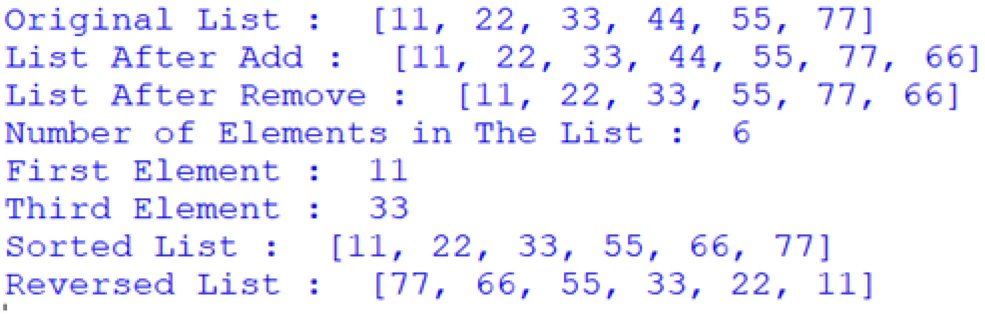
PRACTICLE : 6.1

AIM : Write a program to perform the below operations on the list.

INPUT:-

```
# Create a list
my_list = [11,22,33,44,55,77]
# Display the original list
print("Original List : ", my_list)
# Add an item to the list
my_list.append(66)
print("List After Add : ", my_list)
# Remove an item from the list
my_list.remove(44)
print("List After Remove : ", my_list)
# Get the number of elements in the list
num_elements = len(my_list)
print("Number of Elements in The List : ", num_elements)
# Access elements of the list using the index
line 1 = my_list[0]
Line 2 = my_list[2]
print("First Element : ", line 1)
print("Third Element : ", line 2)
# Sort the list
my_list.sort()
print("Sorted List : ", my_list)
# Reverse the list
my_list.reverse()
print("Reversed List : ", my_list)
```

OUTPUT



```
Original List :  [11, 22, 33, 44, 55, 77]
List After Add :  [11, 22, 33, 44, 55, 77, 66]
List After Remove :  [11, 22, 33, 55, 77, 66]
Number of Elements in The List :  6
First Element :  11
Third Element :  33
Sorted List :  [11, 22, 33, 55, 66, 77]
Reversed List :  [77, 66, 55, 33, 22, 11]
```

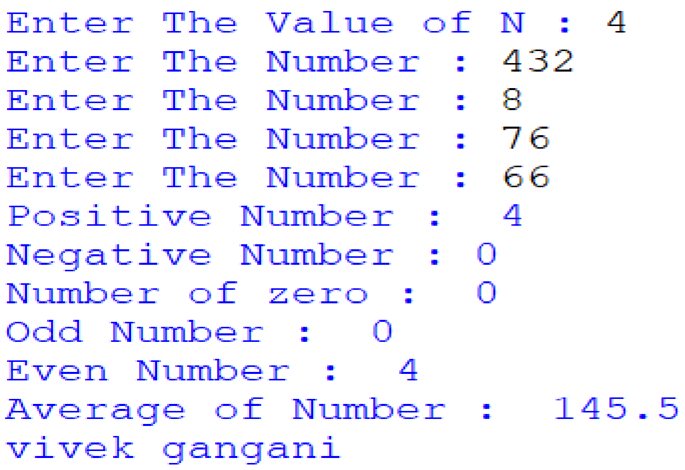
PRACTICLE : 6.2

AIM : Write a program to read n numbers from a user and print.

INPUT:-

```
n = int(input("Enter The Value of N : "))
positive_count = 0
negative_count = 0
zero_count = 0
odd_count = 0
even_count = 0
sum = 0

for i in range(n):
    num = float(input("Enter The Number : "))
    sum += num
    if num > 0:
        positive_count += 1
    elif num < 0:
        negative_count += 1
    else:
        zero_count += 1
    if num % 2 == 0:
        even_count += 1
    else:
        odd_count += 1
# Calculate the average
avg = sum / n
print("Positive Number : ",positive_count)
print("Negative Number :",negative_count)
print("Number of zero : ",zero_count)
print("Odd Number : ",odd_count)
print("Even Number : ",even_count)
print("Average of Number : ",avg)
print(“vivek gangani”)
```

OUTPUT



```
Enter The Value of N : 4
Enter The Number : 432
Enter The Number : 8
Enter The Number : 76
Enter The Number : 66
Positive Number :  4
Negative Number : 0
Number of zero :  0
Odd Number :  0
Even Number :  4
Average of Number :  145.5
vivek gangani
```

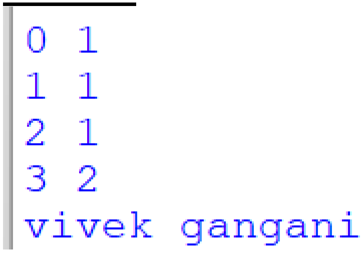
PRACTICLE : 6.3

AIM : Write a program that counts the occurrences of each digit in a string. The program counts how many times a digit appears in the string. For example, if the input is "12203AB3", then the output should output 0 (1 time), 1 (1 time), 2 (2 times), 3 (2 times).

Input:

```
string="1233mn0"
for i in range(4):
    count=string.count(str(i))
    print(i,count)
   print(“vivek gangani”)
```

OUTPUT



```
0 1
1 1
2 1
3 2
vivek gangani
```

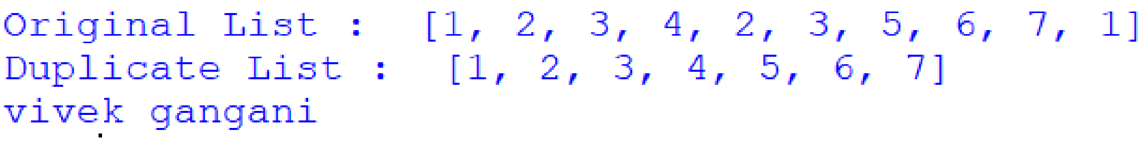
PRACTICLE : 6.4

AIM : Write a program to eliminate duplicate values in the list.

INPUT:-

```
# list with duplicate values
list1 = [1, 2, 3, 4, 2, 3, 5, 6, 7, 1]

# duplicates by converting the list
list2 = list(set(list1))

print("Original List : ",list1)
print("Duplicate List : ",list2)

print(“vivek gangani”)
```

OUTPUT



```
Original List :  [1, 2, 3, 4, 2, 3, 5, 6, 7, 1]
Duplicate List :  [1, 2, 3, 4, 5, 6, 7]
vivek gangani
```

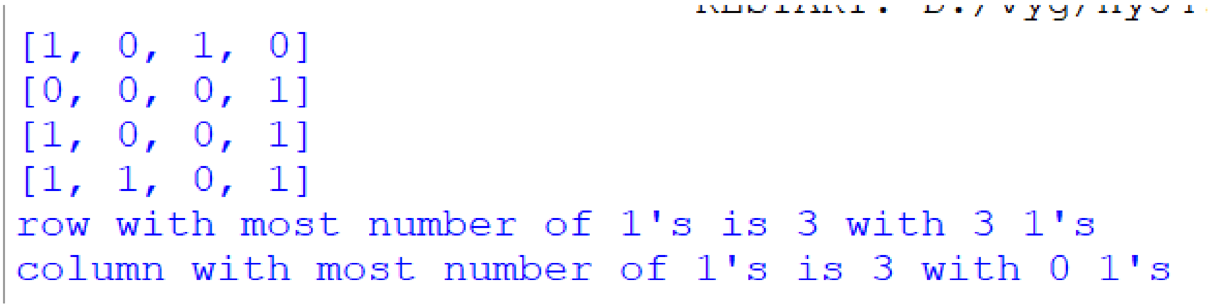
PRACTICLE : 6.5

AIM : Write a program to randomly fill in 0s and 1s into a 4x4 2-dimension list, print the list and find the rows and columns with the most number of 1s.

Input:

```
import random
max_row=0
max_col=0
row_no=0
col_no=0
matrix=[[random.choice([0,1]) for x in range (4)],
        [random.choice([0,1]) for x in range (4)],
        [random.choice([0,1]) for x in range (4)],
        [random.choice([0,1]) for x in range (4)]]

for row in matrix:
    print(row,end=" ")
    print()


for i in range(4):
    row_sum = sum(matrix[i])
    if row_sum>max_row:
         max_row=row_sum
         row_no=i


for j in range(4):
  col_sum=sum (matrix[i][j] for i in range(4))
  if col_sum>max_col:
      col_num = col_sum
      col_no = j

print("row with most number of 1's is",row_no,"with",max_row,"1's")
print("column with most number of 1's is",col_no,"with",max_col,"1's")
```

Output:



```
[1, 0, 1, 0]
[0, 0, 0, 1]
[1, 0, 0, 1]
[1, 1, 0, 1]
row with most number of 1's is 3 with 3 1's
column with most number of 1's is 3 with 0 1's
```

PRACTICLE : 7
Tuples, Sets and Dictionaries

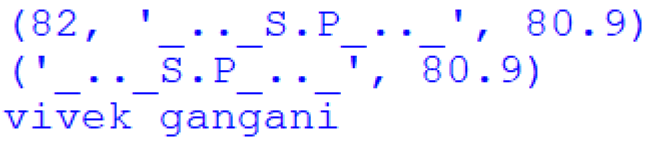
PRACTICLE : 7.1

AIM : Write a program to perform below operations on tuple.

INPUT:-

```
# create a tuple with differnt data types.
my_tuple=(82,'_.._S.P_.._',80.90)

# print tuple items
print(my_tuple)
# convert tuple into a list.
my_list=list(my_tuple)
# remove data items from a list.
my_list.remove(82)
# convert list into a tuple.
my_tuple=tuple(my_list)
# print tuple items.
print(my_tuple)
```

OUTPUT



```
(82, '_.._S.P_.._', 80.9)
('_.._S.P_.._', 80.9)
vivek gangani
```

PRACTICLE : 7.2

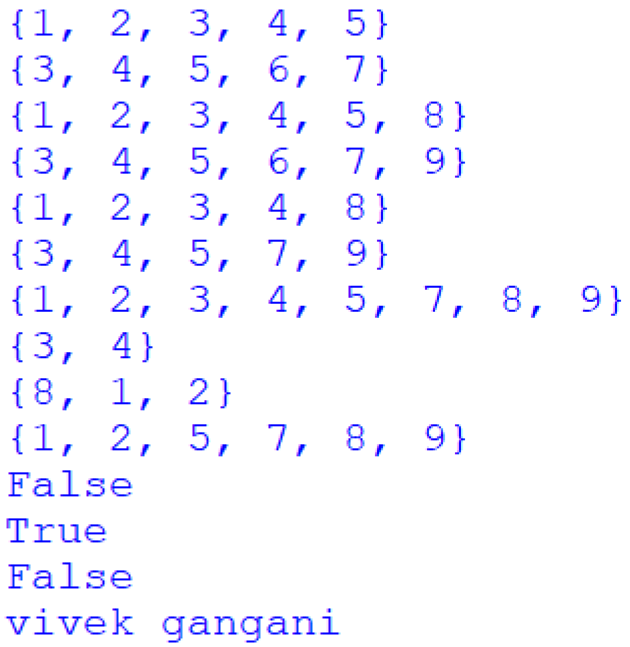
AIM : Write a program to perform below operations on set.

INPUT:-

```
#Create two different sets with the data
set1={1,2,3,4,5}
set2={3,4,5,6,7}

#Print set items
print(set1)
print(set2)

#Add/remove items frim  a set
set1.add(8)
```

```
set2.add(9)
print(set1)
print(set2)
set1.remove(5)
set2.remove(6)
print(set1)
print(set2)

#perform operations on sets:
#. union of set
print(set1|set2)
#. intersection on set
print(set1 & set2)
#. difference on set
print(set1-set2)
#. symmetric difference
print(set1^set2)

#Check subset of another set
#. Super set
set1={1,2,3,4,5}
set2={1,2,3,4}
print(set1 < set2)
#. Sub set
print(set1 > set2)
#. Check set1 & set2 is equivalent
print(set1==set2)
```

OUTPUT



```
{1, 2, 3, 4, 5}
{3, 4, 5, 6, 7}
{1, 2, 3, 4, 5, 8}
{3, 4, 5, 6, 7, 9}
{1, 2, 3, 4, 8}
{3, 4, 5, 7, 9}
{1, 2, 3, 4, 5, 7, 8, 9}
{3, 4}
{8, 1, 2}
{1, 2, 5, 7, 8, 9}
False
True
False
vivek gangani
```

PRACTICLE : 7.3

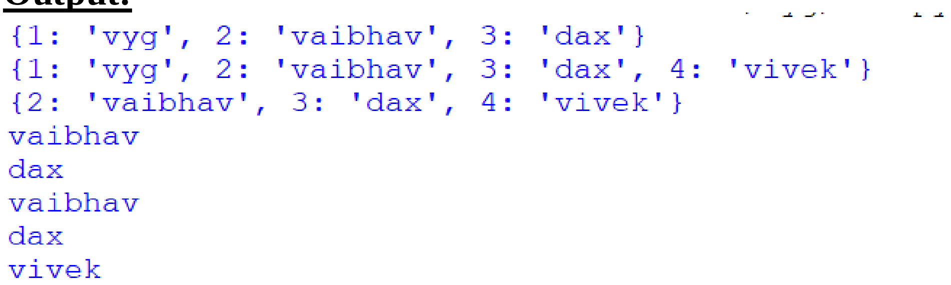
AIM : Write a program to perform below operations on dictionary.

- **Create a dictionary.**
- **Print dictionary items.**
- **Add/remove key-value pair in/from a dictionary.**
- **Check whether a key exists in a dictionary.**
- **Iterate through a dictionary.**
- **Concatenate multiple dictionaries.**

Input:

```
# Create a Dictionary
d={1:'vyg',2:'vaibhav',3:'dax'}
# Print dictionary item.
print(d)
# Add key-value pair in/form a Dictionary
d[4]='vivek'
print(d)
# Remove value from the Dictonary
del d[1]
print(d)
#Check weather a key exists in a Dictionary
print(d.get(2))
print(d.get(3))
#Iterate through a Dictionary
for x in d:
   print(d[x])
```

Output:



```
{1: 'vyg', 2: 'vaibhav', 3: 'dax'}
{1: 'vyg', 2: 'vaibhav', 3: 'dax', 4: 'vivek'}
{2: 'vaibhav', 3: 'dax', 4: 'vivek'}
vaibhav
dax
vaibhav
dax
vivek
```

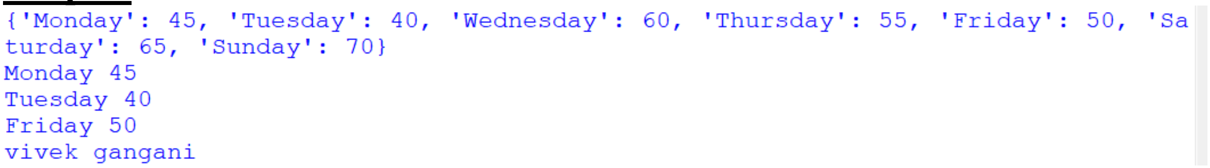
PRACTICLE : 7.4

AIM : Write a program that is given a dictionary containing the average daily temperature for each day of the week and which the average temperature was between 40 and 50 degrees.

Input:

```
# create dictionary
d1 = {'Monday': 45, 'Tuesday': 40, 'Wednesday': 60, 'Thursday': 55, 'Friday': 50,
        'Saturday': 65, 'Sunday': 70}
# print dictionary
print(d1)
# iterate over dictionary
for key, value in d1.items():
    if value >= 40 and value <= 50:
        print(key, value)
print("vivek gangani")
```

Output:



```
{'Monday': 45, 'Tuesday': 40, 'Wednesday': 60, 'Thursday': 55, 'Friday': 50, 'Sa
turday': 65, 'Sunday': 70}
Monday 45
Tuesday 40
Friday 50
vivek gangani
```

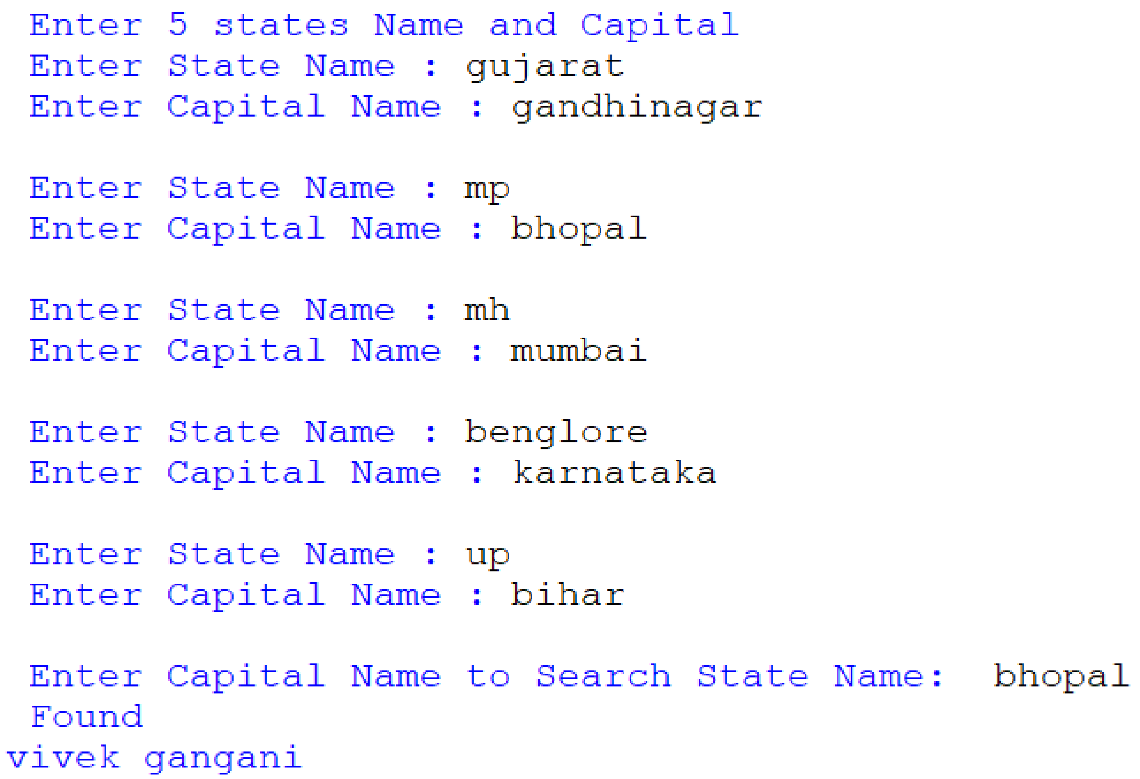
PRACTICLE : 7.5

AIM : Write a program to repeatedly prompt the user to enter the capital of a state.

Input:

```
d1=dict()
print(" Enter 5 states Name and Capital ")
for i in range(5):
        s=input(" Enter State Name : ")
        c=input(" Enter Capital Name : ")
        print()
        d1[s]=c
x=input(" Enter Capital Name to Search State Name:  ")
if x in d1.values():
        print(" Found")
else:
        print(" Not Found")
print("vivek gangani")
```

Output:



```
 Enter 5 states Name and Capital
 Enter State Name : gujarat
 Enter Capital Name : gandhinagar

 Enter State Name : mp
 Enter Capital Name : bhopal

 Enter State Name : mh
 Enter Capital Name : mumbai

 Enter State Name : benglore
 Enter Capital Name : karnataka

 Enter State Name : up
 Enter Capital Name : bihar

 Enter Capital Name to Search State Name:  bhopal
 Found
vivek gangani
```

PRACTICLE : 8.
Functions and Modules

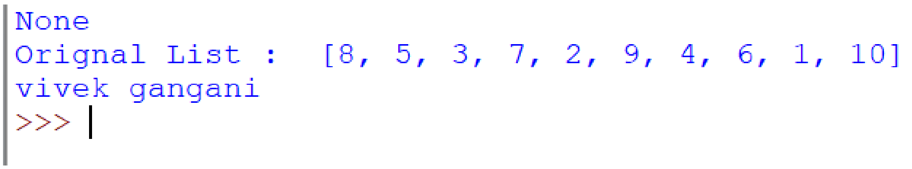
PRACTICLE : 8.1

AIM : Write a program that defines a function (shuffle) to scramble a list into a random order.

Input:

```
import random
numbers=[1,2,3,4,5,6,7,8,9,10]
print(random.shuffle(numbers))
print('Orignal List : ',numbers)
```

Output:



```
None
Orignal List :  [8, 5, 3, 7, 2, 9, 4, 6, 1, 10]
vivek gangani
>>> |
```

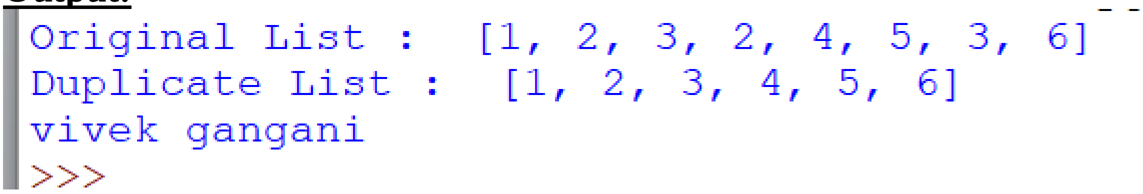
PRACTICLE : 8.2

AIM : Write a program that defines a function to return a new list by eliminating the duplicate values in the list.

Input:

```
def duplicates(list1):
    return list(set(list1))
l1 = [1, 2, 3, 2, 4, 5, 3, 6]
result_list = duplicates(l1)
print("Original List : ", l1)
print("Duplicate List : ", result_list)
```

Output:



```
Original List :  [1, 2, 3, 2, 4, 5, 3, 6]
Duplicate List :  [1, 2, 3, 4, 5, 6]
vivek gangani
>>>
```

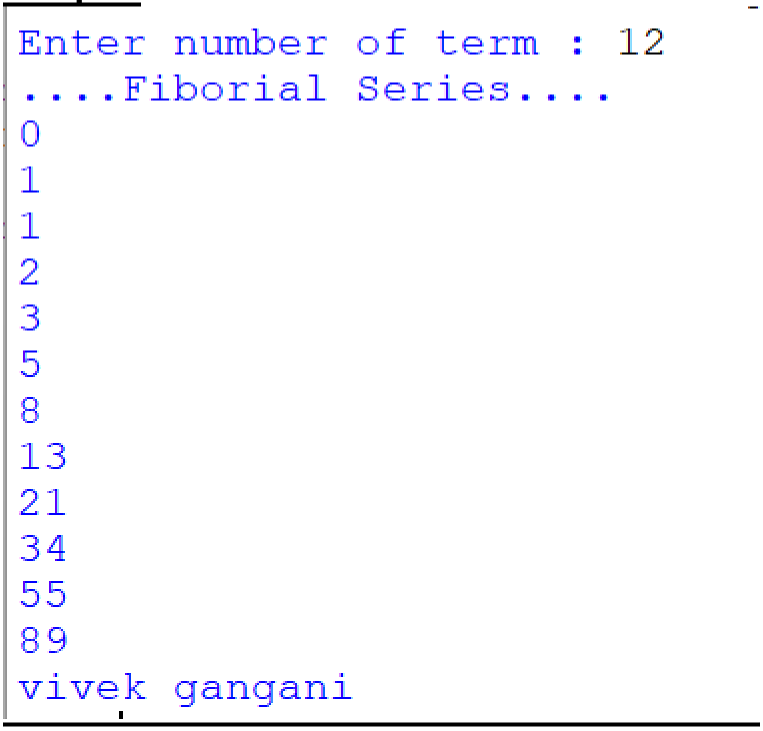
PRACTICLE : 8.3

AIM : Write a program to print Fibonacci sequence up to numbers using recursion.

Input:

```
n=int(input("Enter number of term : "))
def fibo(n):
    if(n<=1):
        return n
    else:
        return (fibo(n-1)+fibo(n-2))
print("....Fiborial Series....")
for i in range(n):
    print(fibo(i))
```

Output:



```
Enter number of term : 12
....Fiborial Series....
0
1
1
2
3
5
8
13
21
34
55
89
vivek gangani
```

PRACTICLE : 8.4

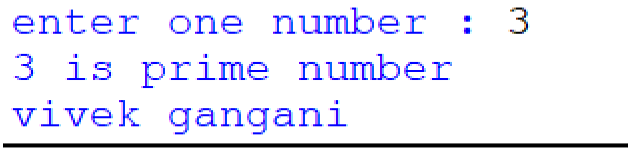
AIM : Write a program to check weather enter number is prine or not

Input:

```
# Acsept Number From User
n=int(input("enter one number : "))
flag=False
def prime(n):
    if n>1:
        for i in range(2,n):
            if(n%i==0):
                flag=True
                break
            else:
                flag=False
        if(flag==True):
            print(n,"is not prime number")
        else:
            print(n,"is prime number")
# function call
```

```
prime(n)
```

Output:



```
enter one number : 3
3 is prime number
vivek gangani
```

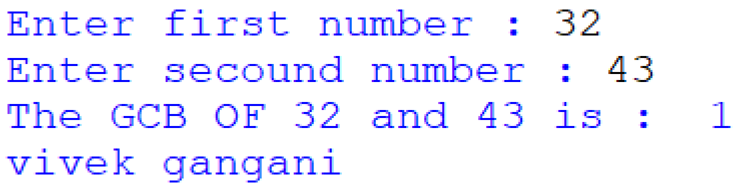
PRACTICLE : 8.5

AIM : Write a program that defines a function to find the GCD of two numbers.

Input:

```
a=int(input("Enter first number : "))
b=int(input("Enter secound number : "))

def gcb(a,b):
    if (b==0):
        return a
    else:
        return gcb(b,a%b)
print("The GCB OF",a,"and",b,"is : ",gcb(a,b))
```

Output:



```
Enter first number : 32
Enter secound number : 43
The GCB OF 32 and 43 is :  1
vivek gangani
```

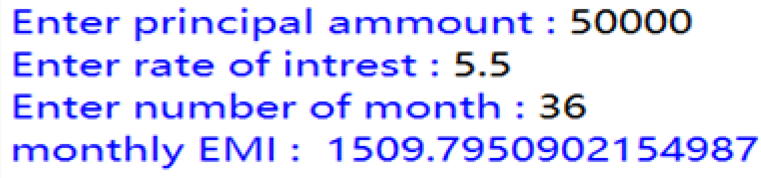
PRACTICLE : 8.6

AIM : Write a program calculate monthly EMI and display the amortization schedule for the loan.

Input:

```
p=int(input("Enter principal ammount : "))
r=float(input("Enter rate of intrest : "))
n=int(input("Enter number of month : "))

def emi_cal(p,r,n):
# calculate intrest for month
    r=r/(12*100)

# calculate EMI instalation per month
    emi=p*r*((1+r)**n/((1+r)**n-1)
# print EMI
    print("monthly EMI : ",emi)
# call function
emi_cal(p,r,n)
```

Output:



```
Enter principal ammount : 50000
Enter rate of intrest : 5.5
Enter number of month : 36
monthly EMI :  1509.7950902154987
```

PRACTICLE : 9
Modules

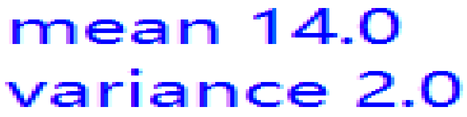
PRACTICLE : 9.1

AIM : Write a program that defines functions that computes mean and standard deviation of given numbers.

Input:

```
list=[12,13,14,15,16]
def variance(list):
    n=len(list)
    mean=sum(list)/n
    print("mean",mean)
    deviation=[(i-mean)**2 for i in list]
    variance=sum(deviation)/n
    return variance
print("variance",variance(list))
```

Output:



```
mean 14.0
variance 2.0
```

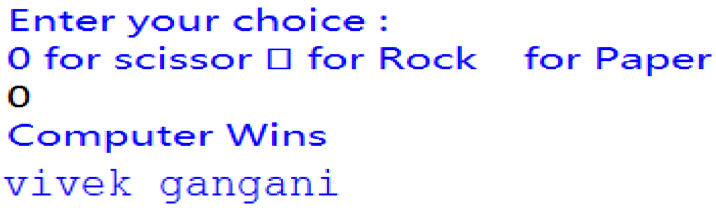
PRACTICLE : 9.2

AIM : Write a program that plays the popular scissor-rock-paper game and displays a message indicating whether the user or the computer wins, loses, or draws.

Input:

```
import random
def game():
    user_choice=int(input("Enter your choice : \n0 for scissor \1 for Rock \2 for Paper \n"))
    computer_choice=random.randint(0,2)
    if user_choice==computer_choice:
        return "Draw"
    elif user_choice == 0: # scissor
        if computer_choice== 1: # ROck
            return "Computer Wins"
        elif computer_choice == 2: #paper
            return "User Wins"
    elif user_choice == 1: # Rock
        if computer_choice == 2: # paper
            return "computer Wins"
        elif computer_choice == 0: # scissor
            return "User Wins"
        elif user_choice == 2: paper
            if computer_wins == 0: 3 scissor
                return "computer Wins"
            elif computer_choice == 1: # Rock
                return "User Wins"
    return "Invalid Input"
print(game())
```

Output:



```
Enter your choice :
0 for scissor □ for Rock    for Paper
0
Computer Wins
vivek gangani
```

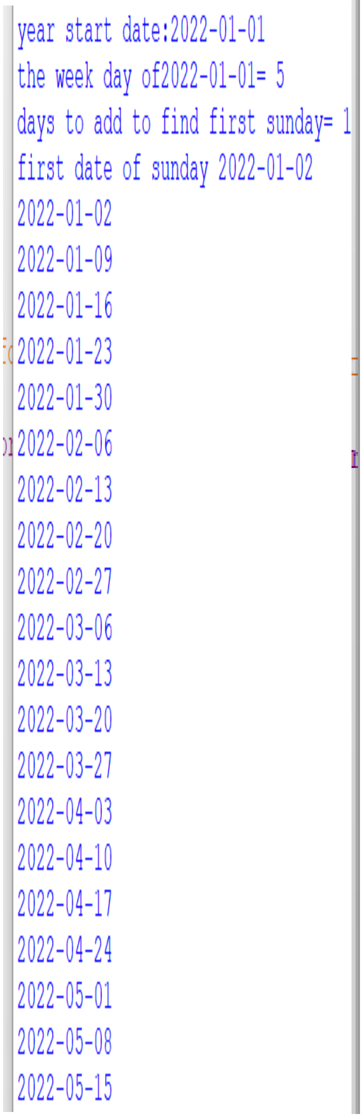
PRACTICLE : 9.3

AIM : Write a program to print the dates of all the Sundays in a given year.

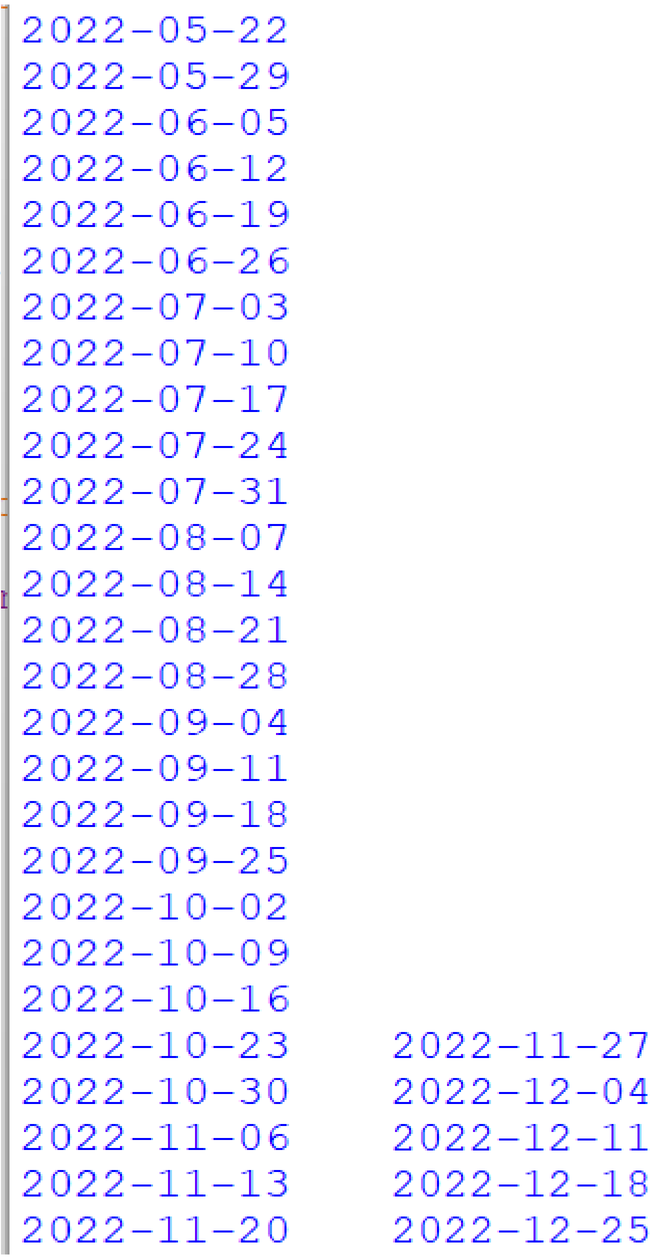
Input:

```
from datetime import date,timedelta
def all_sundays(year):
    dt=date(year,1,1)
    print(f"year start date:{dt}")
    print(f"the week day of{dt}=",dt.weekday())
    days=6-dt.weekday()
    print("days to add to find first sunday=",days)
    dt=dt+timedelta(days)
    print("first date of sunday",dt)
    while dt.year==year:
        yield dt
        dt=dt+timedelta(days=7)
for s in all_sundays(2022):
    print(s)
```

Output:



```
year start date:2022-01-01
the week day of2022-01-01= 5
days to add to find first sunday= 1
first date of sunday 2022-01-02
2022-01-02
2022-01-09
2022-01-16
2022-01-23
2022-01-30
2022-02-06
2022-02-13
2022-02-20
2022-02-27
2022-03-06
2022-03-13
2022-03-20
2022-03-27
2022-04-03
2022-04-10
2022-04-17
2022-04-24
2022-05-01
2022-05-08
2022-05-15
```



```
2022-05-22
2022-05-29
2022-06-05
2022-06-12
2022-06-19
2022-06-26
2022-07-03
2022-07-10
2022-07-17
2022-07-24
2022-07-31
2022-08-07
2022-08-14
2022-08-21
2022-08-28
2022-09-04
2022-09-11
2022-09-18
2022-09-25
2022-10-02
2022-10-09
2022-10-16
2022-10-23
2022-10-30
2022-11-06
2022-11-13
2022-11-20
2022-11-27
2022-12-04
2022-12-11
2022-12-18
2022-12-25
```

PRACTICLE : 9.4

AIM : write a program to display a graph for ReLU (Rectified Linear Unit) function. ReLU function is defined as below: $y = max\ (0, x)$

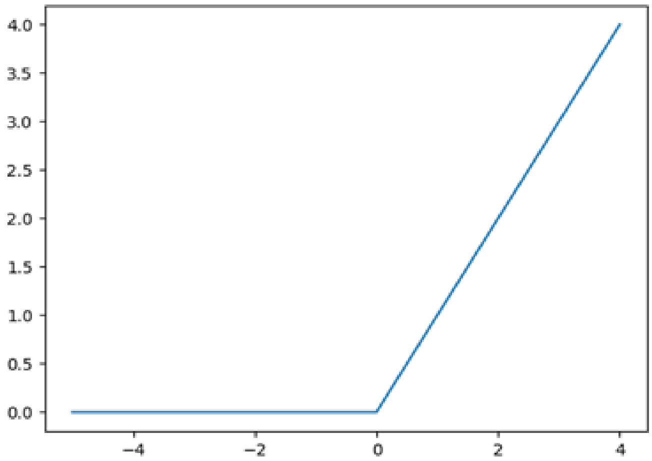
Input:

```
import matplotlib.pyplot as plt
def relu(x):
    return max(0, x)
    print("relu value", relu_value)
    return relu_value
x = list(range(-5, 5))
```

```
print("x=",x)
y = [relu(i) for i in x]
print("y=",y)
plt.plot(x, y)
plt.show()
```

Output:

```
x= [-5, -4, -3, -2, -1, 0, 1, 2, 3, 4]
y= [0, 0, 0, 0, 0, 0, 1, 2, 3, 4]
```



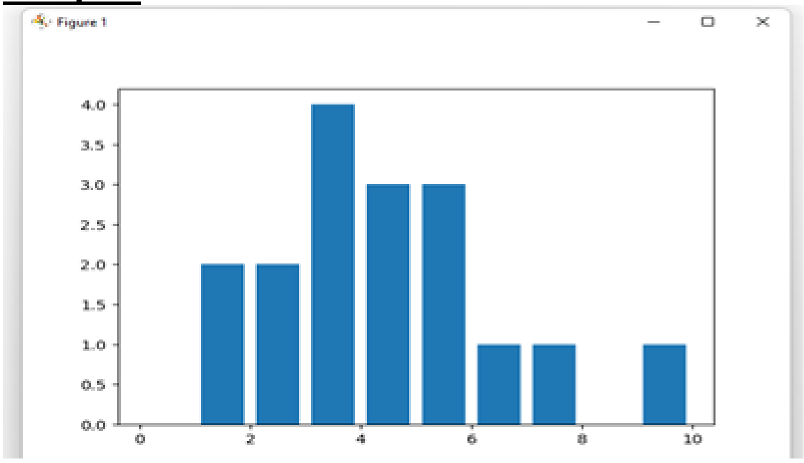
PRACTICLE : 9.5

AIM : Write a program to create a list representing the results of 100 students in a test and display a histogram for the result.

Input:

```
import matplotlib.pyplot as plt
testscores=[5,3,5,4,2,1,3,5,6,7,4,3,2,44,5,54,4,4,44,3,2,1,10,11,23,11]
bins=[0,1,2,3,4,5,6,7,8,9,10]
plt.hist(testscores,bins,histtype='bar',rwidth=0.8)
plt.show()
```

Output:



PRACTICLE : 9.6

AIM : Write a program to import the module and access functions defined in the module.

- **Mymath.py File**

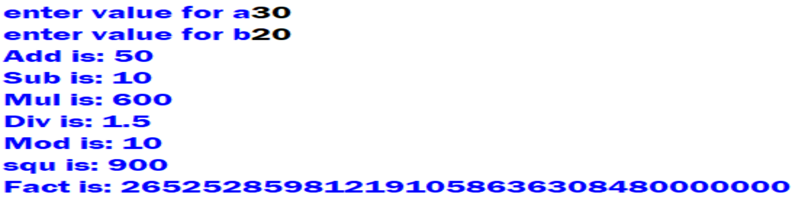
Input:

```
# Addition
def add(a,b):
   return a+b
# Subtraction
def sub(a,b):
   return a-b
# Multiplication
def mul(a,b):
   return a*b
# Division
def div(a,b):
   return a/b
# Modulo division
def mod(a,b):
   return a%a
#Squr root
def mod(a):
   return a*a
# Fectorial
def fact(a):
   if a==0:
      return 1
   else:
      return a * fact (a-1)
```

- **app.pyModule File**

```
from calc import add,sub,mul,div,mod,sqr,fact
a=calc.add(10,15)
b=calc.sub(10,5)
c=calc.mul(6,5)
d=calc.div(10,5)
e=calc.mod(10,5)
f=calc.sqr(5)
g=calc.fact(10)

print("\n Addition : ",a)
print("\n Subtraction : ",b)
print("\n Multiplication : ",c)
print("\n Divition : ",d)
print("\n Modulo Divition : ",e)
print("\n Squre Root : ",f)
print("\n Factorial : ",g)
```

Output:



```
enter value for a30
enter value for b20
Add is: 50
Sub is: 10
Mul is: 600
Div is: 1.5
Mod is: 10
squ is: 900
Fact is: 265252859812191058636308480000000
```

PRACTICLE : 10
String Processing

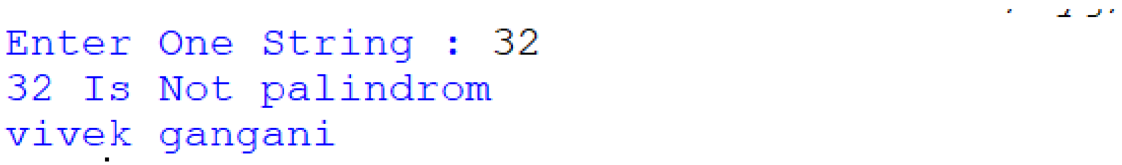
PRACTICLE : 10.1

AIM : Write a program to check whether a given string is palindrome or not.

Input:

```
input_str=input("Enter One String : ")
# string for less comparision
input_str=input_str.casefold()

# reverse string
rev_str=reversed(input_str)

# if statement
if list(input_str)== list(rev_str):
   print(input_str,"Is palindrom")
else:
   print(input_str,"Is Not palindrom")
```

Output:



```
Enter One String : 32
32 Is Not palindrom
vivek gangani
```

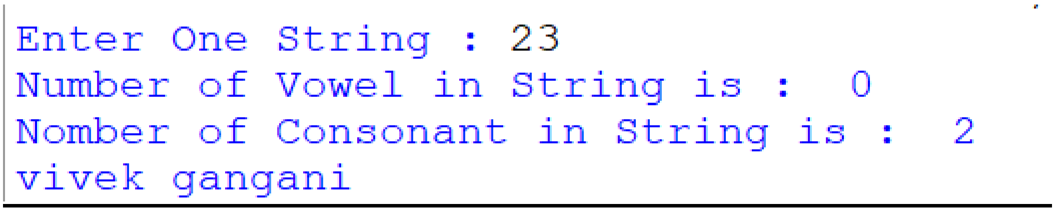
PRACTICLE : 10.2

AIM : Write a program to read a string containing the number of vowels and consonants in the string.

Input:

```
str=input("Enter One String : ")
vowel=0
const=0

for i in str:
   if(i=='a' or i=='e' or i=='i' or i=='o' or i=='u'):
      vowel=vowel+1
   else:
      const=const+1

print("Number of Vowel in String is : ",vowel)
print("Nomber of Consonant in String is : ",const)
```

Output:



```
Enter One String : 23
Number of Vowel in String is :  0
Nomber of Consonant in String is :  2
vivek gangani
```

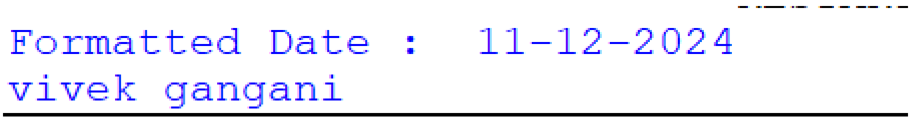
PRACTICLE : 10.3

AIM : Write a program to read a date in the format DD/MM/YYYY and print the same date in MM-DD-YYYY format.

Input:

```
from datetime import datetime
# Current Date
now=datetime.now()
# Converd date into string
new_format=now.strftime("%m-%d-%Y")
# Display formated date
print("Formatted Date : ",new_format)
```

Output:



```
Formatted Date :  11-12-2024
vivek gangani
```

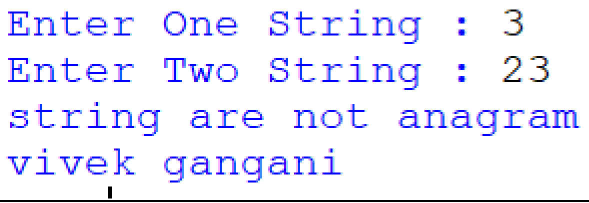
PRACTICLE : 10.4

AIM : Write a program that checks whether two words are anagrams.

Input:

```
s1=input("Enter One String : ")
s2=input("Enter Two String : ")

sort1=sorted(s1)
sort2=sorted(s2)

if (sort1 == sort2):
   print(s1,"and",s2,"are anagram string")
else:
    print("string are not anagram ")
```

Output:



```
Enter One String : 3
Enter Two String : 23
string are not anagram
vivek gangani
```

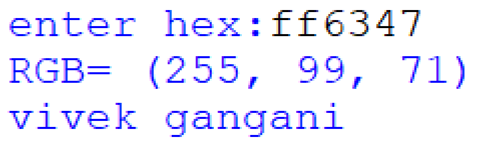
PRACTICLE : 10.5

AIM : Write a program that allows users to enter six-digit RGB color codes and converts them into base 10. In this format, the first two hexadecimal digits represent the amount of red, the second two the amount of green, and the last two the amount of blue. For example: If a user enters FF6347, then the output should be Red (255), Green (99) and Blue (71).

Input:

```
h=input('enter hex:').lstrip('#')
print('RGB=',tuple(int(h[i:i+2],16) for i in (0,2,4)))
print(“vivek gangani”)
```

Output:



```
enter hex:ff6347
RGB= (255, 99, 71)
vivek gangani
```

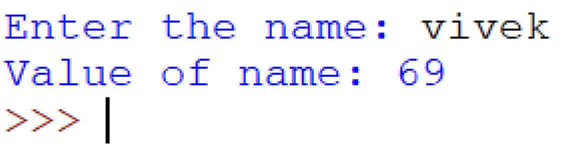
PRACTICLE : 10.6

AIM : . Numerologists claim to be able to determine a person's character traits based on the "numeric value" of a name. The value of a name is determined by summing up the values of the letters of the name, where "a" is 1 "b" is 2 "c" is 3 and so on up to "z" being 26. For example, the name "Python" would have the value 16 + 25 + 20 + 8 + 15 + 14 = 98. Write a program that calculates the numeric value of a name provided as input.

Input:

```
def name_value(name):
    name = name.lower()
    value = 0
    for char in name:
        value += ord(char) - 96
    return value
name = input("Enter the name: ")
print("Value of name:", name_value(name))
```

Output:



```
Enter the name: vivek
Value of name: 69
>>> |
```

Practical: 11

Practical: 11.1

Aim: Write a program to perform the below operations on files:

• Create a text file and write a string to it.

• Read an entire text file.

• Read a text file line by line.

• Write a string to a file.

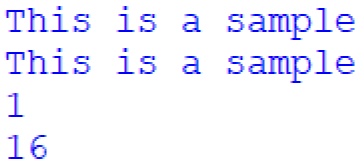
• Write a list of strings to a file.

• Count the number of lines, words in a file.

Input:

```
# create text file and write a string to it
f = open("test.txt", "w")
f.write("This is a sample")
# read the file
f = open("test.txt", "r")
print(f.read())
# read the file line by line
f = open("test.txt", "r")
print(f.readline())
# write a string to the file
f = open("test.txt", "a")
f.write("This is a test file")
# write a list of strings to the file
f = open("test.txt", "a")
f.writelines(["This is a test file", "This is a test file"])
# count the number of lines in the file
f = open("test.txt", "r")
print(len(f.readlines()))
# count the number of words in the file
f = open("test.txt", "r")
print(len(f.read().split()))
f.close()
```

Output:



```
This is a sample
This is a sample
1
16
```

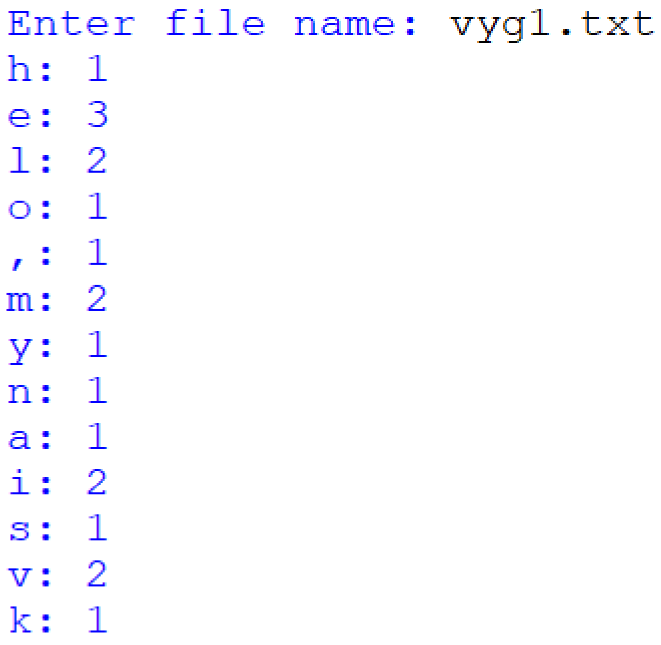
Practical: 11.2

Aim: Write a program that reads a text file and counts the occurrences of each alphabet in the file. The program should prompt the user to enter the filename.

Input:

```
filename = input("Enter file name: ")
# open file in read mode
f = open(filename, "r")
# read the file
data = f.read()
# count the occurrences of characters
count = {}
for char in data:
 if char != " ":
    if char in count:
        count[char] += 1
    else:
        count[char] = 1
# print the result
for key, value in count.items():
 print(f"{key}: {value}")
```

Output:



```
Enter file name: vygl.txt
h: 1
e: 3
l: 2
o: 1
,: 1
m: 2
y: 1
n: 1
a: 1
i: 2
s: 1
v: 2
k: 1
```

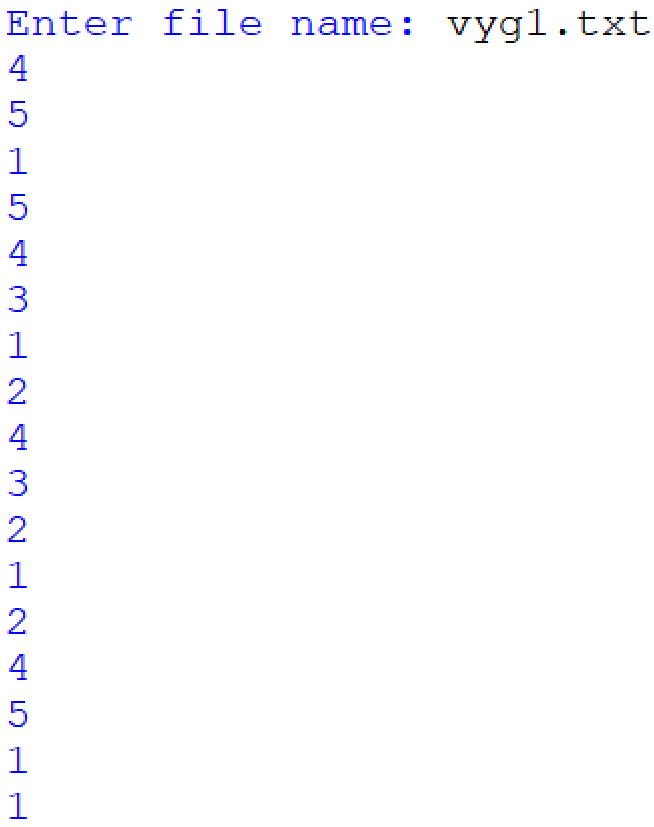
Practical: 11.3

Aim: Write a program that reads a text file and displays all the numbers found in the file.

Input:

```
filename = input("Enter file name: ")
# open file in read mode
f = open(filename, "r")
# read the file
data = f.read()
# display all the numbers in the file
for char in data:
 if char.isdigit():
     print(char)
f.close()
```

Output:



```
Enter file name: vygl.txt
4
5
1
5
4
3
1
2
4
3
2
1
2
4
5
1
1
```

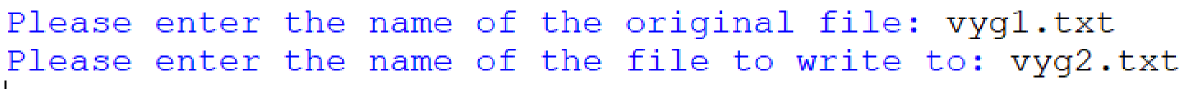
Practical: 11.4

Aim: Write an automated censor program that reads the text from a file and creates a new file where all of the four-letter words have been replaced by "**". You can ignore punctuation, and you may assume that no words in the file are split across multiple lines.**

Input:

```
import string
def main ():
 # Get input file
 file = input("Please enter the name of the original file: ")
 infile = open(file, 'r')
 # get output file
 otherFile = input("Please enter the name of the file to write to: ")
 outfile = open(otherFile, 'w')
 # Change file
 for line in infile:
     words = line.split( )
 for word in words:
     counter = 0
 for letter in word:
     if not letter in string.punctuation:
         counter += 1
         if counter == 4:
             if "." in word:
                 word = "****."
             elif "," in word:
                 word = "****,"
             elif "?" in word:
                 word = "****?"
             elif "!" in word:
                 word = "****!"
             else:
                 word = "****"
 print (word + " ", file = outfile, end = "")
 # close files
 infile.close()
 outfile.close()
if __name__=='__main__':
 main ()
```

Output:



```
Please enter the name of the original file: vyg1.txt
Please enter the name of the file to write to: vyg2.txt
```

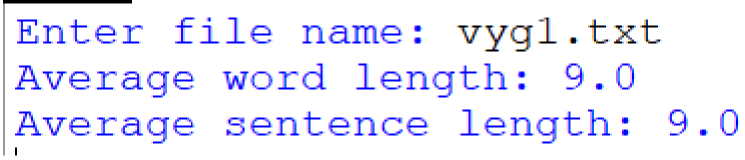
Practical: 11.5

Aim: Write a program that reads a text file and calculates the average word length and sentence length in that file.

Input:

```
filename = input("Enter file name: ")
# open file in read mode
f = open(filename, "r")
# read the file
data = f.read()
# calculate average word length and sentence length
words = data.split()
sentences = data.split(".")
total_words = 0
total_sentences = 0
for word in words:
 total_words += len(word)
for sentence in sentences:
 total_sentences += len(sentence)
average_word_length = total_words / len(words)
average_sentence_length = total_sentences / len(sentences)
print(f"Average word length: {average_word_length}")
print(f"Average sentence length: {average_sentence_length}")
```

Output:



```
Enter file name: vygl.txt
Average word length: 9.0
Average sentence length: 9.0
```

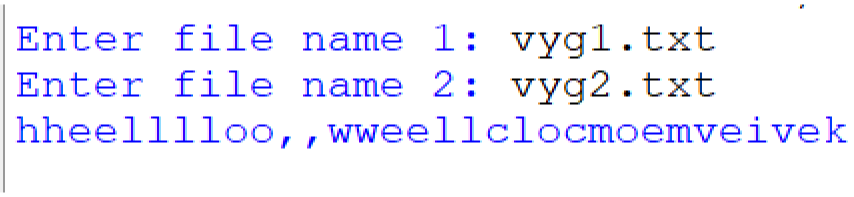
Practical: 11.6

Aim: Write a program that reads two strings stored in two different text files and prints a string containing the characters of each string interleaved. Remove white spaces from both strings before string interleaving. For example, Two strings “Hello World” and “Sky is the Limit” should generate output “HSeklyliosWtohrelLdimit”.

Input:

```
filename1 = input("Enter file name 1: ")
filename2 = input("Enter file name 2: ")
# open file in read mode
f1 = open(filename1, "r")
f2 = open(filename2, "r")
# read the file
data1 = f1.read()
data2 = f2.read()
# remove spaces from the string
data1 = data1.replace(" ", "")
data2 = data2.replace(" ", "")
# get the length of the string
length1 = len(data1)
length2 = len(data2)
max_length = max(length1, length2)
# print interleaved
for i in range(max_length):
 if i < length1:
     print(data1[i], end="")
 if i < length2:
     print(data2[i], end="")
print()
```

Output:



```
Enter file name 1: vyg1.txt
Enter file name 2: vyg2.txt
hheelllloo,,wweellclocmoemveivek
```