Question 1 to 150

Q1 = Write a Python program to print the following string in a specific format (see the output).Twinkle, twinkle, little star, How I wonder what you are! Up above the world so high, Like a diamond in the sky. Twinkle, twinkle, little star, How I wonder what you are"

```
print("""Twinkle, tiwinkle, little star,
          how I wonder what you are!
                    up above the world so high,
                    like a diamond in the sky.
Twinkle, twinkle,little star,
         How i woner that you are""")
```

```
Twinkle, tiwinkle, little star,
          how I wonder what you are!
                    up above the world so high,
                    like a diamond in the sky.
Twinkle, twinkle,little star,
         How i woner that you are
```

Q2 - Write a Python program to find out what version of Python you are using

```
sys.version
```

```
'3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27)
[MSC v.1929 64 bit (AMD64)]'
```

Q3 - Write a Python program to display the current date and time.

Sample Output : Current date and time : 2014-07-05 14:34:14

```
from datetime import datetime

a = datetime.now()

print(a)
```

```
2025-06-18 17:16:57.902628
```

Q4 - Write a Python program that calculates the area of a circle based on the radius entered by the user.

Sample Output : r = 1.1 Area = 3.8013271108436504

```
from math import pi

r = float(input("input the radius of circle : "))
```

```
input the radius of circle :  1.1
```

```
area = pi * r ** 2
```

```
print(f"r = {r} area = {area}")
```

```
r = 1.1 area = 3.8013271108436504
```

Q5 -Write a Python program that accepts the user's first and last name and prints them in reverse order with a space between them.

```
First_name = input("First_name ")
Last_name = input("Last_name")
print(f"{Last_name} {First_name}")
```

```
First_name  Manoj
Last_name Pandey

Pandey Manoj
```

Q6 - Write a Python program that accepts a sequence of comma-separated numbers from the user and generates a list and a tuple of those numbers.

Sample data : 3, 5, 7, 23 Output : List : ['3', ' 5', ' 7', ' 23'] Tuple : ('3', ' 5', ' 7', ' 23')

```
numbers = input("Enter comma-separated numbers: ")
num_list = numbers.split(',')
num_tuple = tuple(num_list)

print("List:", num_list)
print("Tuple:", num_tuple)
```

```
Enter comma-separated numbers:  3,5,7,23

List: ['3', '5', '7', '23']
Tuple: ('3', '5', '7', '23')
```

Q7 - Write a Python program that accepts a filename from the user and prints the extension of the file.

Sample filename : abc.java Output : java

```
filename = input("give me the file name ")
f_extent = filename.split(" . ")
print("the extension of the file is : " + repr(f_extent[-1]))
```

```
give me the file name  ab.java

the extension of the file is : 'ab.java'
```

Q8 -Write a Python program to display the first and last colors from the following list.

color_list = ["Red","Green","White" ,"Black"]

```
colours = input("Give me the colours list: ")
col = colours.split(',')  # Convert comma-separated string to list
print("Here are two colours:", repr(col[0]), "and", repr(col[2]))
```

```
Give me the colours list:  red , green ,white , black
```

```
Here are two colours: 'red ' and 'white '
```

```
# Create a list called 'color_list' containing color names
color_list = ["Red", "Green", "White", "Black"]
# Print the first and last elements of the 'color_list' using string formatting
# The '%s' placeholders are filled with the values of 'color_list[0]' (Red) and 'color_list[-1]' (Black)
print("%s %s" % (color_list[0], color_list[-1]))
```

```
Red Black
```

Q9- Write a Python program to display the examination schedule. (extract the date from exam_st_date).

exam_st_date = (11, 12, 2014) Sample Output : The examination will start from : 11 / 12 / 2014

```
DATE = (11, 12, 2014)
print("The examination will start from: %i / %i / %i" % (DATE))
```

```
The examination will start from: 11 / 12 / 2014
```

Q10 = Write a Python program that accepts an integer (n) and computes the value of n+nn+nnn.

Sample value of n is 5 Expected Result : 615

```
a = int(input("here is an interget"))
n1 = int("%s" % a )
n2 = int("%s%s" %(a,a ))
n3 = int("%s%s%s" % (a,a,a))
print(n1+n2+n3)
```

```
here is an interget 5
```

```
615
```

Q11 = Write a Python program to print the documents (syntax, description etc.) of Python built-in function(s).

Sample function : abs() Expected Result : *abs(number) -> number* Return the absolute value of the argument

```
print(abs.__doc__)

Return the absolute value of the argument.

print(len.__doc__)

Return the number of items in a container.

print(sorted.__doc__)

Return a new list containing all items from the iterable in ascending
order.

A custom key function can be supplied to customize the sort order, and
the
reverse flag can be set to request the result in descending order.

print(sum.__doc__)

Return the sum of a 'start' value (default: 0) plus an iterable of
numbers

When the iterable is empty, return the start value.
This function is intended specifically for use with numeric values and
may
reject non-numeric types.

print(map.__doc__)
print(filter.__doc__)

map(func, *iterables) --> map object

Make an iterator that computes the function using arguments from
each of the iterables.  Stops when the shortest iterable is exhausted.
filter(function or None, iterable) --> filter object

Return an iterator yielding those items of iterable for which
function(item)
is true. If function is None, return the items that are true.
```

Q12 =Write a Python program that prints the calendar for a given month and year.

- Note : Use 'calendar' module.

```
import calendar
y = int(input("print the year boss"))
m = int(input("print the month baby"))

print(calendar.month(y,m))

print the year boss 2025
print the month baby 7
```

```
     July 2025
Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31


calendar.prmonth(2034,5)

      May 2034
Mo Tu We Th Fr Sa Su
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

Q13 - Write a Python program to print the following 'here document'.

- Sample string :
- a string that you "don't" have to escape
- This
- is a ..... multi-line
- heredoc string --------> example

```
print("""
a string that you "dont have to escape
This
is a ......muti-line
hearodoc string --------> example
""")


a string that you "dont have to escape
This
is a ......muti-line
hearodoc string --------> example
```

Q14=Write a Python program to calculate the number of days between two dates.

- Sample dates : (2014, 7, 2), (2014, 7, 11)
- Expected output : 9 days

```
from datetime import date

a = date (2014, 7, 2)
b = date (2014, 7, 11)
```

```
hours = b-a           #if u try without .days u will get both days as
well time
print(hours.days)

9
```

Q15 - Write a Python program to get the volume of a sphere with radius six.

```
import math
radius = 6
volume = 4/3 * math.pi * radius **3

print('volume of the sphere with radius 6 is:', volume)

volume of the sphere with radius 6 is: 904.7786842338603
```

Q16 - Write a Python program to calculate the difference between a given number and 17. If the number is greater than 17, return twice the absolute difference.

```
#self
g=int(input("first number here"))
o=int(input(" seond number"))

b = g-o
c = b*2
print('the number difference is:',b)
print("twice the number is:",c)

first number here 20
 seond number 10

the number difference is: 10
twice the number is: 20

#2nd method

def difference(n):
    if n<= 17:
        return 17-n

    else:
        return(n-17)*2


num = int(input("enter a number dude"))

print((difference(num)))

enter a number dude 5

12
```

Q17 - Write a Python program to test whether a number is within 100 of 1000 or 2000.

```
def thousand(n):
    return ((abs(1000-n) <= 100) or (abs(2000 - n) <= 100))

print(thousand(500))
print(thousand(1500))
print(thousand(1000))
```

```
False
False
True
```

```
def test(n):
    if n <= 100:
        return ("100 ke andr h bhai")

    elif n <= 1000:
         return("Bigger then 100 but shorter then 1k")

    else:
         return("Bigger then 1k but under 2k")


difference = int(input(" WRITE THE NUMBMER DUDE"))

print(test(difference))
```

```
 WRITE THE NUMBMER DUDE 999
Bigger then 100 but shorter then 1k
```

Q18 = Write a Python program to calculate the sum of three given numbers. If the values are equal, return three times their sum.

```
def sum(a,b,c):
    sum = a +b +c

    if  a==b==c:
        sum = sum*3

    return sum

print(sum(1,1,1))
print(sum(2,4,1))
```

```
9
7
```

```
def sum(a, b, c):
    if a==b==c:
        return 3*(a + b + c)
    else:
        return a + b + c

print(sum(1, 2, 3))
```

```
6
```

Q19 = Write a Python program to get a newly-generated string from a given string where "Is" has been added to the front. Return the string unchanged if the given string already begins with "Is".

```
def string(s):
    if s.startswith("is"):
        return s
    else:
        return "is" + s

print(string("array"))
print(string("isEmpty"))
```

```
isarray
isEmpty
```

Q20 =Write a Python program that returns a string that is n (non-negative integer) copies of a given string.

```
def string(text , n):
    result = ""

    for i in range(n):
        result = result + text

    return result

print(string(" hello " , 3))
```

```
 hello  hello  hello
```

```
text = input(" enter a string ")
n = int(input(" enter how many times to repeat :"))

print(text  *  n)
```

```
 enter a string  manoj
 enter how many times to repeat : 4
```

```
manojmanojmanojmanoj
```

Q21 - Write a Python program that determines whether a given number (accepted from the user) is even or odd, and prints an appropriate message to the user

```
number = int(input( "please write the numner"))


if  num % 2 ==0 :
    print("this is odd number")
else:
    print("this is even number")
```

```
please write the numner 4
```

```
this is even number
```

Q22 - Write a Python program to count the number 4 in a given list.

```
numbers = [1, 4, 6, 4, 7, 4, 9, 4, 10]

count = numbers.count(4)

print("Number 4 appears", count, "times in the list.")
```

```
Number 4 appears 4 times in the list.
```

```
number = [ 1,4,6,4,5,3,4,5,6,8,9,7,6,]

n= int(input(" enter the number to count : " ))

count = number.count(n)
print(count)
```

```
 enter the number to count :  4
```

```
3
```

Q23 - Write a Python program to get n (non-negative integer) copies of the first 2 characters of a given string. Return n copies of the whole string if the length is less than 2.

```
string = input(" you can write your text here ")
n = int(input("here is how mmany times u need "))



if len(string) < 2:
    result = string * n

else:
    result = string[:2] *n

print("result"  , result)
```

```
 you can write your text here  manoj
here is how mmany times u need  4

result mamamama

string = input('enter the text here ')
n = int(input('how many times u wantthe entry' ))

if len(string) <2:
              result = string * n
else:
    result = string[:2] * n

print('result' , result)

enter the text here  karbinaro
how many times u wantthe entry 5

result kakakakaka
```

Q24 = Write a Python program to test whether a passed letter is a vowel or not.

```
vowel = input(" write the vowel here")

if  vowel in ('a' ,'e' , 'i' , 'o' , 'u'):
    result = ' yes it is passed this is vowel'
else:
    result = 'no this is not passed'

print(result)

 write the vowel here a

 yes it is passed this is vowel

#2nd


def vowel(check):
    all = 'aeiou'
    return check in all

print(vowel(input('write here')))

write here a

True
```

Q25 = Write a Python program that checks whether a specified value is contained within a group of values.

```
def check(value ,group):
    return value in group

group = [10,20,30,40]
value = int(input('enter a number to check'))

if check(value, group):
    print('yes , the value is in the group')

else:
    print('no, the value is not in the group')
```

```
enter a number to check 3

no, the value is not in the group
```

Q26 - Write a Python program to create a histogram from a given list of integers.

```
def hist(numbners):
    for num in numbers:
        print('*' * num)

data = [2,4,6,4,5,]

hist(data)
```

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

Q27 - Write a Python program that concatenates all elements in a list into a string and returns it.

```
data = ['h' , 'e' , 'l' , 'l' ,'e']


def conct(elements):
    result = ''.join(elements)
    return result

print(conct(data))
```

```
helle
```

```
words = ['m' , 'a' , 'n' , 'o' , 'j']

def add(likho):
    result = ''.join(likho)
    return result

print(add(words))
```

```
manoj
```

Q28 - Write a Python program to print all even numbers from a given list of numbers in the same order and stop printing any after 237 in the sequence.

Sample numbers list:

numbers = [
386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978, 328, 615, 953, 345, 399, 162, 758, 219, 918, 237, 412, 566, 826, 248, 866, 950, 626, 949, 687, 217, 815, 67, 104, 58, 512, 24, 892, 894, 767, 553, 81, 379, 843, 831, 445, 742, 717, 958,743, 527

```
numbers = [386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978,
328, 615, 953, 345, 399, 162, 758, 219, 918, 237, 412, 566, 826, 248,
866, 950, 626, 949, 687, 217, 815, 67, 104, 58, 512, 24, 892, 894,
767, 553, 81, 379, 843, 831, 445, 742, 717, 958,743, 527]

for num in numbers:
    if num==237:
        break

    if num % 2 == 0:
        print(num)
```

```
386
462
418
344
236
566
978
328
162
758
918
```

```
#2nd method
numbers = [386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978,
328, 615, 953, 345, 399, 162, 758, 219, 918, 237, 412, 566, 826, 248,
866, 950, 626, 949, 687, 217, 815, 67, 104, 58, 512, 24, 892, 894,
767, 553, 81, 379, 843, 831, 445, 742, 717, 958,743, 527]
```

```
for x in numbers:
    if x == 237:
        break

    if x % 2 ==0:
        print(x)
```

```
386
462
418
344
236
566
978
328
162
758
918
```

Q 29 = Write a Python program that prints out all colors from color_list_1 that are not present in color_list_2.

Test Data : color_list_1 = set(["White", "Black", "Red"])

color_list_2 = set(["Red", "Green"])

Expected Output : {'Black', 'White'}

```
color1 = set(["White", "Black", "Red"])

color2 = set(["Red", "Green"])


result  = color1 - color2
print(result)
```

```
{'White', 'Black'}
```

Q30 = Write a Python program that will accept the base and height of a triangle and compute its area

```
b = int(input("aap yhaa base likhiyee "))
h = int(input('app yha height likhiye '))

a = 0.5 * b *h

print(f'this is area of traingle:' ,  a)
```

```
aap yhaa base likhiyee  4
app yha height likhiye  4

this is area of traingle: 8.0
```

Q31 - Write a Python program that computes the greatest common divisor (GCD) of two positive integers.

```
import math

a = int(input("enter the fisrt number"))
b = int(input(' enter second number'))


print('the gcd is:' , math.gcd(a,b))

enter the fisrt number 336
 enter second number 360

the gcd is: 24
```

Q32- Write a Python program to find the least common multiple (LCM) of two positive integers.

```
import math

a = int(input('enter the no .bro'))
b - int(input('fir se daal number'))

print('the lcm is:' , math.lcm(a,b))

enter the no .bro 336
fir se daal number 360

the lcm is: 5040
```

Q33 - Write a Python program to sum three given integers. However, if two values are equal, the sum will be zero.

```
a = int(input(' enter the first digit '))
b = int(input(' enter the second digit '))
c = int(input(' enter the third digit '))

def sum(a,b,c):
    if a==b or b==c or c==a :
        sum = 0
        return sum
    else:
        sum = a+b+c
        return sum
```

```
print(f'This is the value :',sum(a,b,c))
```

```
 enter the first digit  3
 enter the second digit  2
 enter the third digit  4

This is the value : 9
```

Q34 - Write a Python program to sum two given integers. However, if the sum is between 15 and 20 it will return 20.

```
a = int(input('enter the first digit '))
b = int(input('enter the two digit'))

def sum(a,b):
    sum = a+b
    if sum in  range(15,20):
        return (f'sum is bwtween 15 to 20:' , 20)

    else:
        return sum

print(sum(a,b))
```

```
enter the first digit  4
enter the two digit 11

('sum is bwtween 15 to 20:', 20)
```

Q35 - Write a Python program that returns true if the two given integer values are equal or their sum or difference is 5.

```
###
a = int(input('enter the first digit '))
b = int(input('enter the two digit'))



def rule(a,b):
    return a==b or a+b ==5 or abs(a-b) == 5



print(rule(a,b))
```

```
enter the first digit  2
enter the two digit 4

False
```

```
a = int(input('enter the first digit '))
b = int(input('enter the two digit'))



def rule(a,b):
    return a==b or a+b ==5 or abs(a-b) == 5



print(rule(a,b))
```

```
enter the first digit  4
enter the two digit 4

True
```

Q36 -Write a Python program to add two objects if both objects are integers

```
a = input("Enter first value: ")
b = input("Enter second value: ")

# Try to convert both to integers
if a.isdigit() and b.isdigit():
    result = int(a) + int(b)
    print("Sum is:", result)
else:
    print("Both inputs must be integers.")
```

```
Enter first value:  1
Enter second value:  5

Sum is: 6
```

```
a = input("Enter first value: ")
b = input("Enter second value: ")

# Try to convert both to integers
if a.isdigit() and b.isdigit():
    result = int(a) + int(b)
    print("Sum is:", result)
else:
    print("Both inputs must be integers.")
```

```
Enter first value:  manoj
Enter second value:  oa

Both inputs must be integers.
```

Q37 - Write a Python program that displays your name, age, and address on three different lines.

```
A ='Name :    Manoj'
b ='Age :     19'
c ='Address : Banglore karnatake'

print(A + "\n" + b + "\n" + c)
```

```
Name :    Manoj
Age :     19
Address : Banglore karnatake
```

```
A = input('whats your first name ')
b = int(input('please write your age '))
c = input (' whats ur addres')

print(f' Name:' , A + "\n" + f'Age:' , str(b) + "\n" + f'address:', c)
```

```
whats your first name  majok
please write your age  33
 whats ur addres banglore

 Name: majok
Age: 33
address: banglore
```

Q38 - Write a Python program to solve (x + y) * (x + y).

```
a = int(input('x  please '))
b = int(input('y  please '))

def square(a,b):
    sum = a+b
    return sum ** 2



print(square(a,b))
```

```
x  please  4
y  please  3

49
```

Q39 - Write a Python program to compute the future value of a specified principal amount, rate of interest, and number of years.

```
amt = 10000
int= 3.5
years = 7
```

```
future_value = amt * (( 1 + (0.01 * int)) ** years)

print(round(future_value , 2))
```

```
12722.79
```

Q40 = rite a Python program to calculate the distance between the points (x1, y1) and (x2, y2).

```
import math

p1 = [3 ,5 ]
p2 = [4 ,10]

distance =math.sqrt(((p1[0] - p2[0]) ** 2) + ((p1[1] - p2[1]) ** 2))
print(distance)
```

```
5.0990195135927845
```

```
import math

x1 = float(input("Enter x1: "))
y1 = float(input("Enter y1: "))
x2 = float(input("Enter x2: "))
y2 = float(input("Enter y2: "))


distance = math.sqrt((x2 - x1) ** 2 + (y2 - y1)**2)
print("Distance between the points is:", round(distance, 2))
```

```
Enter x1:  4
Enter y1:  5
Enter x2:  3
Enter y2:  3

Distance between the points is: 2.24
```

Q41 - Write a Python program to check whether a file exists.

```
import os
file_path = 'mnoj.boss'

if os.path.exists(file_path):

    print("file exisist")
else:
    print('file not exist')
```

```
file not exist
```

```
import os
file_path = 'python practice.ipynb'

if os.path.exists(file_path):

    print("file exisist")
else:
    print('file not exist')
```

```
file exisist
```

Q42- Write a Python program to determine if a Python shell is executing in 32bit or 64bit mode on OS.

```
import struct
# packing and unpacking data in c style binary format we use struct

print(struct.calcsize("P") * 8)
```

```
64
```

```
import platform
bit_mode = platform.architecture()[0]
print(bit_mode)
```

```
64bit
```

Q43 -Write a Python program to get OS name, platform and release information.

```
import os
import platform

print('os name ', os.name)
print('platform', platform.system())
print('release' , platform.release())
print('version' , platform.version())
```

```
os name  nt
platform Windows
release 11
version 10.0.26100
```

Q44 - Write a Python program to locate Python site packages.

```
import site

print(site.getsitepackages())
```

```
['C:\\Users\\ASUS\\anaconda', 'C:\\Users\\ASUS\\anaconda\\Lib\\site-packages']
```

Q45 - Write a Python program that calls an external command.

```
# Import the 'os' module to work with the operating system.
import os

# Use 'os.system(command)' to execute the 'ls -l' command in the
system's shell.
# This command lists the files and directories in the current
directory and prints the result.
print(os.system('ls -l'))
```

```
1
```

Q 46 -Write a Python program to retrieve the path and name of the file currently being executed.

```
import os
import sys

# Get the current notebook path
notebook_path = os.path.abspath(sys.argv[0])

print("Notebook Path:", notebook_path)
```

```
Notebook Path: C:\Users\ASUS\anaconda\Lib\site-packages\
ipykernel_launcher.py
```

Q47 -Write a Python program to find out the number of CPUs used.

```
import os

# Get the number of CPUs
num_cpus = os.cpu_count()

print("Number of CPUs:", num_cpus)
```

```
Number of CPUs: 8
```

```
# Import the 'multiprocessing' module to work with multi-processing
features.
import multiprocessing

# Use 'multiprocessing.cpu_count()' to determine the number of
available CPU cores.
cpu_count = multiprocessing.cpu_count()

# Print the number of CPU cores available on the system.
print(cpu_count)
```

```
8
```

Q48 = Write a Python program to parse a string to float or integer.

```
a = "143.00"
b = float(a)
c = int(b)

print(a)
print(b)
print(c)
```

```
143.00
143.0
143
```

Q49 =Write a Python program to list all files in a directory.

```
import os

# Get current working directory
current_path = os.getcwd()

# List everything in current directory
items = os.listdir(current_path)

# Print only directories
for item in items:
    if os.path.isdir(os.path.join(current_path, item)):
        print(item)
```

```
.ipynb_checkpoints
```

Q50 = Write a Python program to print without a newline or space.

```
print("hello" , end='')
print("world" , end='')
```

```
helloworld
```

```
a = "manoj"
b = "pandey"

print(a, b, sep='', end='')
```

```
manojpandey
```

Q51 =Write a Python program to determine the profiling of Python programs.

```
import cProfile
def sum():
    print(1+2)

cProfile.run('sum()')
```

```
3
         486 function calls (470 primitive calls) in 0.003 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall
filename:lineno(function)
        3    0.000    0.000    0.000    0.000 <frozen
abc>:121(__subclasscheck__)
        5    0.000    0.000    0.000    0.000 <frozen
importlib._bootstrap>:1390(_handle_fromlist)
        1    0.000    0.000    0.000    0.000
_weakrefset.py:75(__contains__)
        1    0.000    0.000    0.000    0.000 asyncio.py:210(call_at)
        2    0.000    0.000    0.000    0.000
asyncio.py:225(add_callback)
        1    0.000    0.000    0.000    0.000
asyncio.py:539(_start_select)
        5    0.000    0.000    0.000    0.000
attrsettr.py:42(__getattr__)
        5    0.000    0.000    0.000    0.000
attrsettr.py:65(_get_attr_opt)
      2/0    0.000    0.000    0.000
base_events.py:1909(_run_once)
        5    0.000    0.000    0.000    0.000
base_events.py:2004(get_debug)
        3    0.000    0.000    0.000    0.000
base_events.py:539(_check_closed)
        4    0.000    0.000    0.000    0.000 base_events.py:734(time)
        1    0.000    0.000    0.000    0.000
base_events.py:743(call_later)
        1    0.000    0.000    0.000    0.000
base_events.py:767(call_at)
        2    0.000    0.000    0.000    0.000
base_events.py:785(call_soon)
        2    0.000    0.000    0.000    0.000
base_events.py:814(_call_soon)
       24    0.000    0.000    0.000    0.000 enum.py:1129(__new__)
       51    0.000    0.000    0.000    0.000 enum.py:1544(_get_value)
        9    0.000    0.000    0.000    0.000 enum.py:1551(__or__)
        8    0.000    0.000    0.000    0.000 enum.py:1562(__and__)
       24    0.000    0.000    0.000    0.000 enum.py:726(__call__)
        1    0.000    0.000    0.000    0.000 events.py:111(__init__)
        1    0.000    0.000    0.000    0.000 events.py:127(__lt__)
        3    0.000    0.000    0.000    0.000 events.py:36(__init__)
        3    0.000    0.000    0.001    0.000 events.py:86(_run)
        2    0.000    0.000    0.000    0.000 ioloop.py:541(time)
        1    0.000    0.000    0.000    0.000
ioloop.py:596(call_later)
        2    0.000    0.000    0.001    0.000
```

```
ioloop.py:742(_run_callback)
        1    0.000    0.000    0.000    0.000
iostream.py:137(_event_pipe)
        2    0.000    0.000    0.000    0.000
iostream.py:156(_handle_event)
        1    0.000    0.000    0.000    0.000
iostream.py:212(_is_master_process)
        1    0.000    0.000    0.000    0.000
iostream.py:215(_check_mp_mode)
        1    0.000    0.000    0.000    0.000 iostream.py:254(closed)
        1    0.000    0.000    0.000    0.000
iostream.py:258(schedule)
        1    0.000    0.000    0.000    0.000
iostream.py:275(<lambda>)
        1    0.000    0.000    0.000    0.000
iostream.py:277(_really_send)
        2    0.000    0.000    0.000    0.000
iostream.py:519(_is_master_process)
        2    0.000    0.000    0.000    0.000
iostream.py:546(_schedule_flush)
        1    0.000    0.000    0.000    0.000
iostream.py:556(_schedule_in_thread)
      1/0    0.000    0.000    0.000           iostream.py:624(write)
        1    0.000    0.000    0.000    0.000
proactor_events.py:792(_loop_self_reading)
        2    0.000    0.000    0.000    0.000
proactor_events.py:883(_process_events)
        3    0.000    0.000    0.000    0.000 queue.py:209(_qsize)
        3    0.000    0.000    0.000    0.000 queue.py:97(empty)
        1    0.000    0.000    0.000    0.000 queues.py:173(qsize)
        1    0.000    0.000    0.000    0.000 queues.py:225(get)
        1    0.000    0.000    0.000    0.000
queues.py:256(get_nowait)
        1    0.000    0.000    0.000    0.000
queues.py:322(_consume_expired)
        1    0.000    0.000    0.000    0.000
queues.py:59(_set_timeout)
      9/8    0.000    0.000    0.000    0.000 socket.py:621(send)
        1    0.000    0.000    0.000    0.000
socket.py:698(send_multipart)
        2    0.000    0.000    0.000    0.000
socket.py:777(recv_multipart)
        1    0.000    0.000    0.000    0.000
threading.py:1155(_wait_for_tstate_lock)
        1    0.000    0.000    0.000    0.000
threading.py:1222(is_alive)
        2    0.000    0.000    0.000    0.000
threading.py:299(__enter__)
        2    0.000    0.000    0.000    0.000
threading.py:302(__exit__)
```

```
        1    0.000    0.000    0.000    0.000 threading.py:323(wait)
        1    0.000    0.000    0.000    0.000 threading.py:394(notify)
        1    0.000    0.000    0.000    0.000 threading.py:601(is_set)
        3    0.000    0.000    0.000    0.000
typing.py:1221(__instancecheck__)
        4    0.000    0.000    0.000    0.000 typing.py:1285(__hash__)
        3    0.000    0.000    0.000    0.000
typing.py:1492(__subclasscheck__)
        2    0.000    0.000    0.000    0.000 typing.py:2182(cast)
        6    0.000    0.000    0.000    0.000 typing.py:392(inner)
        1    0.000    0.000    0.000    0.000
windows_events.py:429(_check_closed)
      2/0    0.000    0.000    0.000
windows_events.py:443(select)
        1    0.000    0.000    0.000    0.000
windows_events.py:482(recv)
        1    0.000    0.000    0.000    0.000
windows_events.py:55(__init__)
        1    0.000    0.000    0.000    0.000
windows_events.py:704(_register_with_iocp)
        1    0.000    0.000    0.000    0.000
windows_events.py:714(_register)
      2/0    0.000    0.000    0.000
windows_events.py:761(_poll)
        5    0.000    0.000    0.000    0.000
zmqstream.py:562(receiving)
        3    0.000    0.000    0.000    0.000
zmqstream.py:566(sending)
        2    0.000    0.000    0.000    0.000
zmqstream.py:580(_run_callback)
        2    0.000    0.000    0.001    0.000
zmqstream.py:607(_handle_events)
        2    0.000    0.000    0.000    0.000
zmqstream.py:648(_handle_recv)
        3    0.000    0.000    0.001    0.000
zmqstream.py:687(_rebuild_io_state)
        3    0.000    0.000    0.001    0.000
zmqstream.py:710(_update_handler)
        2    0.000    0.000    0.001    0.000
zmqstream.py:718(<lambda>)
        3    0.000    0.000    0.000    0.000 {built-in method
_abc._abc_subclasscheck}
        2    0.000    0.000    0.000    0.000 {built-in method
_asyncio.get_running_loop}
        3    0.000    0.000    0.000    0.000 {built-in method
_contextvars.copy_context}
        1    0.000    0.000    0.000    0.000 {built-in method
_heapq.heappush}
      2/0    0.000    0.000    0.000          {built-in method
_overlapped.GetQueuedCompletionStatus}
```

```
        1    0.000    0.000    0.000    0.000 {built-in method
_thread.allocate_lock}
        5    0.000    0.000    0.000    0.000 {built-in method
builtins.getattr}
        5    0.000    0.000    0.000    0.000 {built-in method
builtins.hasattr}
        4    0.000    0.000    0.000    0.000 {built-in method
builtins.hash}
  110/104    0.000    0.000    0.000    0.000 {built-in method
builtins.isinstance}
        3    0.000    0.000    0.000    0.000 {built-in method
builtins.issubclass}
       14    0.000    0.000    0.000    0.000 {built-in method
builtins.len}
        2    0.000    0.000    0.000    0.000 {built-in method
builtins.max}
        1    0.000    0.000    0.000    0.000 {built-in method
builtins.min}
        3    0.000    0.000    0.000    0.000 {built-in method
math.ceil}
        3    0.000    0.000    0.000    0.000 {built-in method
nt.getpid}
        4    0.000    0.000    0.000    0.000 {built-in method
time.monotonic}
        2    0.000    0.000    0.000    0.000 {built-in method
time.time}
        1    0.000    0.000    0.000    0.000 {method 'WSARecv' of
'_overlapped.Overlapped' objects}
        2    0.000    0.000    0.000    0.000 {method '__enter__' of
'_thread.RLock' objects}
        4    0.000    0.000    0.000    0.000 {method '__exit__' of
'_thread.RLock' objects}
        3    0.000    0.000    0.000    0.000 {method '__exit__' of
'_thread.lock' objects}
        1    0.000    0.000    0.000    0.000 {method
'_acquire_restore' of '_thread.RLock' objects}
        2    0.000    0.000    0.000    0.000 {method '_is_owned' of
'_thread.RLock' objects}
        1    0.000    0.000    0.000    0.000 {method '_release_save'
of '_thread.RLock' objects}
        3    0.000    0.000    0.000    0.000 {method 'acquire' of
'_thread.lock' objects}
        1    0.000    0.000    0.000    0.000 {method
'add_done_callback' of '_asyncio.Future' objects}
        5    0.000    0.000    0.000    0.000 {method 'append' of
'collections.deque' objects}
        2    0.000    0.000    0.000    0.000 {method 'clear' of
'list' objects}
        1    0.000    0.000    0.000    0.000 {method 'disable' of
'_lsprof.Profiler' objects}
```

```
        1    0.000    0.000    0.000    0.000 {method 'fileno' of
'_socket.socket' objects}
        2    0.000    0.000    0.000    0.000 {method 'keys' of 'dict'
objects}
        5    0.000    0.000    0.000    0.000 {method 'popleft' of
'collections.deque' objects}
        1    0.000    0.000    0.000    0.000 {method 'release' of
'_thread.lock' objects}
        1    0.000    0.000    0.000    0.000 {method 'remove' of
'collections.deque' objects}
        1    0.000    0.000    0.000    0.000 {method 'result' of
'_asyncio.Future' objects}
        3    0.000    0.000    0.001    0.000 {method 'run' of
'_contextvars.Context' objects}
        5    0.000    0.000    0.000    0.000 {method 'upper' of 'str'
objects}
        2    0.000    0.000    0.000    0.000 {method 'write' of
'_io.StringIO' objects}
```

Q52 = Write a Python program to print to STDERR.

```
import sys

print("This is an error message.", file=sys.stderr)
```

```
This is an error message.
```

Q53 = Write a Python program to access environment variables.

```
import os

# Print all environment variables
print("All Environment Variables:")
for key, value in os.environ.items():
    print(f'{key}: {value}')

# Access a specific environment variable
print("\nAccessing a specific variable:")
print("PATH:", os.environ.get('PATH'))  # or 'HOME', 'USERNAME', etc.
```

```
All Environment Variables:
ALLUSERSPROFILE: C:\ProgramData
APPDATA: C:\Users\ASUS\AppData\Roaming
COMMONPROGRAMFILES: C:\Program Files\Common Files
COMMONPROGRAMFILES(X86): C:\Program Files (x86)\Common Files
COMMONPROGRAMW6432: C:\Program Files\Common Files
COMPUTERNAME: DESKTOP-FAP3T6Q
COMSPEC: C:\WINDOWS\system32\cmd.exe
CONDA_DEFAULT_ENV: base
```

```
CONDA_EXE: C:\Users\ASUS\anaconda\Scripts\conda.exe
CONDA_PREFIX: C:\Users\ASUS\anaconda
CONDA_PROMPT_MODIFIER: (base)
CONDA_PYTHON_EXE: C:\Users\ASUS\anaconda\python.exe
CONDA_ROOT: C:\Users\ASUS\anaconda
CONDA_SHLVL: 1
DRIVERDATA: C:\Windows\System32\Drivers\DriverData
EFC_17636_1262719628: 1
EFC_17636_1592913036: 1
EFC_17636_2283032206: 1
EFC_17636_2775293581: 1
EFC_17636_3789132940: 1
FPS_BROWSER_APP_PROFILE_STRING: Internet Explorer
FPS_BROWSER_USER_PROFILE_STRING: Default
HOMEDRIVE: C:
HOMEPATH: \Users\ASUS
IPY_INTERRUPT_EVENT: 4176
JPY_INTERRUPT_EVENT: 4176
JPY_PARENT_PID: 4364
JPY_SESSION_NAME: C:\Users\ASUS\A,,I neuron classses 50 lpa manoj
package\python self practice\python practice.ipynb
LOCALAPPDATA: C:\Users\ASUS\AppData\Local
LOGONSERVER: \\DESKTOP-FAP3T6Q
NUMBER_OF_PROCESSORS: 8
ONEDRIVE: C:\Users\ASUS\OneDrive
OS: Windows_NT
PATH: C:\Users\ASUS\anaconda;C:\Users\ASUS\anaconda\Library\mingw-w64\
bin;C:\Users\ASUS\anaconda\Library\usr\bin;C:\Users\ASUS\anaconda\
Library\bin;C:\Users\ASUS\anaconda\Scripts;C:\Users\ASUS\anaconda\
bin;C:\Users\ASUS\anaconda;C:\Users\ASUS\anaconda\Library\mingw-w64\
bin;C:\Users\ASUS\anaconda\Library\usr\bin;C:\Users\ASUS\anaconda\
Library\bin;C:\Users\ASUS\anaconda\Scripts;C:\WINDOWS\system32;C:\
WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\
WindowsPowerShell\v1.0;C:\WINDOWS\System32\OpenSSH;C:\Users\ASUS\
anaconda3\Scripts;C:\Program Files\Git\cmd;C:\Users\ASUS\anaconda\
condabin;C:\Program Files\MySQL\MySQL Shell 8.0\bin;C:\Users\ASUS\
AppData\Local\Microsoft\WindowsApps;C:\Users\ASUS\AppData\Local\
Programs\Microsoft VS Code\bin;.
PATHEXT: .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE: AMD64
PROCESSOR_IDENTIFIER: Intel64 Family 6 Model 140 Stepping 1,
GenuineIntel
PROCESSOR_LEVEL: 6
PROCESSOR_REVISION: 8c01
PROGRAMDATA: C:\ProgramData
PROGRAMFILES: C:\Program Files
PROGRAMFILES(X86): C:\Program Files (x86)
PROGRAMW6432: C:\Program Files
PROMPT: (base) $P$G
```

```
PSMODULEPATH: C:\Program Files\WindowsPowerShell\Modules;C:\WINDOWS\
system32\WindowsPowerShell\v1.0\Modules
PUBLIC: C:\Users\Public
PYDEVD_USE_FRAME_EVAL: NO
SESSIONNAME: Console
SSL_CERT_FILE: C:\Users\ASUS\anaconda\Library\ssl\cacert.pem
SYSTEMDRIVE: C:
SYSTEMROOT: C:\WINDOWS
TEMP: C:\Users\ASUS\AppData\Local\Temp
TMP: C:\Users\ASUS\AppData\Local\Temp
USERDOMAIN: DESKTOP-FAP3T6Q
USERDOMAIN_ROAMINGPROFILE: DESKTOP-FAP3T6Q
USERNAME: ASUS
USERPROFILE: C:\Users\ASUS
WINDIR: C:\WINDOWS
ZES_ENABLE_SYSMAN: 1
__CONDA_OPENSLL_CERT_FILE_SET: "1"
TERM: xterm-color
CLICOLOR: 1
FORCE_COLOR: 1
CLICOLOR_FORCE: 1
PAGER: cat
GIT_PAGER: cat
MPLBACKEND: module://matplotlib_inline.backend_inline

Accessing a specific variable:
PATH: C:\Users\ASUS\anaconda;C:\Users\ASUS\anaconda\Library\mingw-w64\
bin;C:\Users\ASUS\anaconda\Library\usr\bin;C:\Users\ASUS\anaconda\
Library\bin;C:\Users\ASUS\anaconda\Scripts;C:\Users\ASUS\anaconda\
bin;C:\Users\ASUS\anaconda;C:\Users\ASUS\anaconda\Library\mingw-w64\
bin;C:\Users\ASUS\anaconda\Library\usr\bin;C:\Users\ASUS\anaconda\
Library\bin;C:\Users\ASUS\anaconda\Scripts;C:\WINDOWS\system32;C:\
WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\
WindowsPowerShell\v1.0;C:\WINDOWS\System32\OpenSSH;C:\Users\ASUS\
anaconda3\Scripts;C:\Program Files\Git\cmd;C:\Users\ASUS\anaconda\
condabin;C:\Program Files\MySQL\MySQL Shell 8.0\bin;C:\Users\ASUS\
AppData\Local\Microsoft\WindowsApps;C:\Users\ASUS\AppData\Local\
Programs\Microsoft VS Code\bin;.
```

Q54 = Write a Python program to get the current username.

```
import getpass

print(getpass.getuser())
```

```
ASUS
```

```
import os

m = os.getlogin()
```

```
print(m)
```

```
ASUS
```

Q55 - Write a Python program to find local IP addresses using Python's stdlib

```
import socket

def local_ip():
    hostname = socket.gethostname()
    local_ip = socket.gethostbyname(hostname)
    return local_ip

print(local_ip)
```

```
<function local_ip at 0x000002D76E7DAFC0>
```

Q56 - Write a Python program to get the height and width of the console window.

```
import shutil

size = shutil.get_terminal_size()

print('console width:', size.columns)
print('console height:', size.lines)
```

```
console width: 80
console height: 24
```

Q57 - Write a Python program to get the execution time of a Python method.

```
import time

def example_function():
    total = 0
    for i in range(1000000):
        total += i
    return total

# Start time
start_time = time.time()

# Run the function
example_function()

# End time
end_time = time.time()

# Calculate execution time
execution_time = end_time - start_time
print(f"Execution Time: {execution_time:.5f} seconds")
```

```
Execution Time: 0.16121 seconds
```

Q 58 - Write a Python program to sum the first n positive integer

```
n = int(input('first positive interger:'))

total = n*(n+1) // 2

print(total)
```

```
first positive interger: 9

45
```

Q59 -Write a Python program to convert height (in feet and inches) to centimeters.

```
feet = int(input("enter the feet "))
inches = int(input("enter the inches"))

total = feet * 12 + inches
centimeter = total * 2.54

print(f"Height in centimeter: {centimeter:.2f} cm")
```

```
enter the feet  4
enter the inches 6

Height in centimeter: 137.16 cm
```

Q60 - Write a Python program to convert all units of time into seconds.

```
days = int(input("enter the number of days"))
hours = int(input("enter the number of hours"))
minutes = int(input("enter the number of minutes"))
seconds = int(input("enter the number of seconds"))

total_seconds = (days * 86400) + (hours * 3600) + (minutes * 60) +
(seconds)


print(f"total time in seconds: {total_seconds}")
```

```
enter the number of days 4
enter the number of hours 4
enter the number of minutes 5
enter the number of seconds 5

total time in seconds: 360305
```

Q 61 = Write a Python program to convert the distance (in feet) to inches, yards, and miles.

```
feet = int(input("type feet to inch "))


finch = feet * 12
fyard = feet / 3
fmiles = feet / 52800

print(f'Feet to inches: {finch:.2f}\nFeet to yards: {fyard:.2f}\nFeet
to miles: {fmiles:1f}')
```

```
type feet to inch  4

Feet to inches: 48.00
Feet to yards: 1.33
Feet to miles: 0.000076
```

```
feet = int(input("type feet to inch "))
yards = int(input("now feeet to yards "))
miles  = int(input("now  feet to miles"))


finch = feet * 12
fyard = feet / 3
fmiles = feet / 52800

print(f'Feet to inches: {finch:.2f}\nFeet to yards: {fyard:.2f}\nFeet
to miles: {fmiles:.5f}')
```

```
type feet to inch  4
now feeet to yards  32
now  feet to miles 5

Feet to inches: 48.00
Feet to yards: 1.33
Feet to miles: 0.00008
```

Q62 - Write a Python program to calculate the hypotenuse of a right angled triangle.

```
import math

a = float(input('Enter the length of side A: '))
b = float(input('Enter the length of side B: '))

hypotenuse = math.sqrt(a**2 + b**2)

print(f"The hypotenuse is: {hypotenuse:.2f}")
```

```
Enter the length of side A:  4
Enter the length of side B:  4

The hypotenuse is: 5.66
```

Q63 - Write a Python program to get an absolute file path

```python
import os

# Ask user for filename (can be relative)
file_name = input("Enter the file name or path: ")

# Get the absolute path
abs_path = os.path.abspath(file_name)

# Print the result
print(f"The absolute path is: {abs_path}")
```

```
Enter the file name or path:  self

The absolute path is: C:\Users\ASUS\A,,I neuron classses 50 lpa manoj
package\python self practice\self
```

Q64 - Write a Python program that retrieves the date and time of file creation and modification.

```python
import os
import datetime

# Ask user for file name or path
file_path = input("Enter the file name or full path: ")

# Check if the file exists
if os.path.exists(file_path):
    # Get file creation time (in seconds since epoch)
    creation_time = os.path.getctime(file_path)

    # Convert to readable date and time
    readable_time = datetime.datetime.fromtimestamp(creation_time)

    print(f"□ File: {file_path}")
    print(f"□ Created on: {readable_time.strftime('%Y-%m-%d %H:%M:
%S')}")
else:
    print("□ File does not exist!")
```

```
Enter the file name or full path:  python practice

□ File does not exist!
```

Q65 = Write a Python program that converts seconds into days, hours, minutes, and seconds.

```
# Input: total seconds
total_seconds = int(input("Enter total seconds: "))

# Calculate days
days = total_seconds // 86400
remaining = total_seconds % 86400

# Calculate hours
hours = remaining // 3600
remaining = remaining % 3600

# Calculate minutes
minutes = remaining // 60

# Remaining seconds
seconds = remaining % 60

# Output
print(f"\n Converted Time:")
print(f"{days} days, {hours} hours, {minutes} minutes, {seconds}
seconds")
```

```
Enter total seconds:  1000


 Converted Time:
0 days, 0 hours, 16 minutes, 40 seconds
```

Q66 - Write a Python program to calculate the body mass index.

```
# Input from user
weight = float(input("Enter your weight in kilograms: "))
height = float(input("Enter your height in meters: "))

# Calculate BMI
bmi = weight / (height ** 2)

# Print result
print("Your BMI is:", round(bmi, 2))

# BMI Category
if bmi < 18.5:
    print("Category: Underweight")
elif 18.5 <= bmi < 24.9:
    print("Category: Normal weight")
elif 25 <= bmi < 29.9:
    print("Category: Overweight")
```

```
else:
    print("Category: Obese")
```

```
Enter your weight in kilograms:  55
Enter your height in meters:  4

Your BMI is: 3.44
Category: Underweight
```

Q67 - Write a Python program to convert pressure in kilopascals to pounds per square inch,a millimeter of mercury (mmHg) and atmosphere pressure.

```
kpa = float(input("Enter pressure in kilopascals (kPa): "))

psi = kpa * 0.145038
mmhg = kpa * 7.50062
atm = kpa / 101.325

# Output
print("Pressure in pounds per square inch:", (psi))
print("Pressure in millimeters of mercury (mmHg):", (mmhg))
print("Pressure in atmospheres:", (atm))
```

```
Enter pressure in kilopascals (kPa):  55

Pressure in pounds per square inch: 7.9770900000000005
Pressure in millimeters of mercury (mmHg): 412.53409999999997
Pressure in atmospheres: 0.5428077966938071
```

Q68 = Write a Python program to calculate sum of digits of a number

```
# Take input from the user
number = int(input("Enter a number: "))

# Convert to positive (if negative)
number = abs(number)

# Initialize sum
digit_sum = 0

# Loop through each digit
while number > 0:
    digit_sum += number % 10
    number //= 10

# Print the result
print("Sum of digits:", digit_sum)
```

```
Enter a number:  44

Sum of digits: 8
```

Q 69 = Write a Python program to sort three integers without using conditional statements and loops

```
a =  int(input('short the  first   n : '))
b =  int(input('short the  second   n : '))
c =  int(input('short the  third   n : '))


shorts = sorted([c,b,a])

print(shorts)
```

```
short the  first   n :  3
short the  second   n :  6
short the  third   n :  4

[3, 4, 6]
```

```
# Input a number
a = int(input("Enter a number to sort its digits: "))

# Convert number to string, sort digits, join and convert back to int
shorts = int("".join(sorted(str(a))))

# Output
print("Sorted digits:", shorts)
```

```
Enter a number to sort its digits:  4533

Sorted digits: 3345
```

```
a = input("Enter a number: ")
lst = list(a)
lst.sort()
print("Sorted digits:", "".join(lst))
```

```
Enter a number:  78543

Sorted digits: 34578
```

Q 70 = Write a Python program to sort files by dat

```
# Import the necessary libraries to work with file operations and
globbing.
import glob
import os

# Use the glob module to find all files in the current directory with
a ".txt" extension.
files = glob.glob("*.txt")

# Sort the list of file names based on the modification time
(getmtime) of each file.
```

```
files.sort(key=os.path.getmtime)

# Print the sorted list of file names, one per line.
print("\n".join(files))
```

Q71 - Write a Python program to get a directory listing, sorted by creation date.

```
import os
from pathlib import Path
from datetime import datetime

# Get all files/folders in current directory
items = list(Path('.').iterdir())

# Sort items by creation time
sorted_items = sorted(items, key=os.path.getctime)

# Print name with formatted creation date
for item in sorted_items:
    ctime = os.path.getctime(item)
    formatted_date = datetime.fromtimestamp(ctime).strftime('%d-%m-%Y
%H:%M:%S')
    print(f"{item.name} --> Created on: {formatted_date}")
```

```
python practice.ipynb --> Created on: 18-06-2025 16:10:39
.ipynb_checkpoints --> Created on: 18-06-2025 16:10:39
```

Q72 = Write a Python program to get the details of the math module.

```
import math
math_ls = dir(math)
print(math_ls)
```

```
['__doc__', '__loader__', '__name__', '__package__', '__spec__',
'acos', 'acosh', 'asin', 'asinh', 'atan', 'atan2', 'atanh', 'cbrt',
'ceil', 'comb', 'copysign', 'cos', 'cosh', 'degrees', 'dist', 'e',
'erf', 'erfc', 'exp', 'exp2', 'expm1', 'fabs', 'factorial', 'floor',
'fmod', 'frexp', 'fsum', 'gamma', 'gcd', 'hypot', 'inf', 'isclose',
'isfinite', 'isinf', 'isnan', 'isqrt', 'lcm', 'ldexp', 'lgamma',
'log', 'log10', 'log1p', 'log2', 'modf', 'nan', 'nextafter', 'perm',
'pi', 'pow', 'prod', 'radians', 'remainder', 'sin', 'sinh', 'sqrt',
'sumprod', 'tan', 'tanh', 'tau', 'trunc', 'ulp']
```

Q73 - Write a Python program to calculate the midpoints of a line.

```
x1 , y1 = 2 ,4
x2 , y2 = 6 ,8

mid_x = (x1 + x2) / 2
```

```
mid_y = (y1 + y2) / 2

print(mid_x ,('and') , mid_y )
```

```
4.0 and 6.0
```

Q74 = Write a Python program to hash a word.

```
import hashlib
word = input('enter a word')

hashed= hashlib.sha256(word.encode()).hexdigest()

print(hashed)
```

```
enter a word 4

4b227777d4dd1fc61c6f884f48641d02b4d121d3fd328cb08b5531fcacdabf8a
```

Q75 - Write a Python program to get the copyright information and write Copyright information in Python code.

```
import sys
print(sys.copyright)
```

```
Copyright (c) 2001-2023 Python Software Foundation.
All Rights Reserved.

Copyright (c) 2000 BeOpen.com.
All Rights Reserved.

Copyright (c) 1995-2001 Corporation for National Research Initiatives.
All Rights Reserved.

Copyright (c) 1991-1995 Stichting Mathematisch Centrum, Amsterdam.
All Rights Reserved.
```

Q76 - Write a Python program to get the command-line arguments (name of the script, the number of arguments, arguments) passed to a script.

```
import sys  # Command-line arguments ko access karne ke liye sys module chahiye

print("Script Name:", sys.argv[0])  # Pehla argument script ka naam hota hai
print("Number of Arguments:", len(sys.argv) - 1)  # Baaki arguments ginke count karna
print("Arguments:", sys.argv[1:])  # Sirf actual arguments print karna (script naam ke alawa)
```

```
Script Name: C:\Users\ASUS\anaconda\Lib\site-packages\
ipykernel_launcher.py
Number of Arguments: 2
Arguments: ['-f', 'C:\\Users\\ASUS\\AppData\\Roaming\\jupyter\\
runtime\\kernel-99dee772-a78b-47ff-9a9f-59a651cba8e4.json']
```

Q 77 - Write a Python program to test whether the system is a big-endian platform or a little-endian platform.

```
import sys
print(sys.byteorder)
```

```
little
```

Q78 -Write a Python program to find the available built-in modules.

```
import sys
print(sys.builtin_module_names)
```

```
('_abc', '_ast', '_bisect', '_blake2', '_codecs', '_codecs_cn',
'_codecs_hk', '_codecs_iso2022', '_codecs_jp', '_codecs_kr',
'_codecs_tw', '_collections', '_contextvars', '_csv', '_datetime',
'_functools', '_heapq', '_imp', '_io', '_json', '_locale', '_lsprof',
'_md5', '_multibytecodec', '_opcode', '_operator', '_pickle',
'_random', '_sha1', '_sha2', '_sha3', '_signal', '_sre', '_stat',
'_statistics', '_string', '_struct', '_symtable', '_thread',
'_tokenize', '_tracemalloc', '_typing', '_warnings', '_weakref',
'_winapi', '_xxinterpchannels', '_xxsubinterpreters', 'array',
'atexit', 'audioop', 'binascii', 'builtins', 'cmath', 'errno',
'faulthandler', 'gc', 'itertools', 'marshal', 'math', 'mmap',
'msvcrt', 'nt', 'sys', 'time', 'winreg', 'xxsubtype', 'zlib')
```

Q79 -Write a Python program to get the current value of the recursion limit.

```
import sys
print(sys.getrecursionlimit())
```

```
3000
```

Q 80 - Write a Python program to get the size of an object in bytes.

```
import sys

word = "manoj"

size = sys.getsizeof(word)
print(size)
```

```
46
```