1. Write a Python program to calculate the area of a rectangle given its length and width.

def rect\_area(l, w):

    area = l \* w

    return area

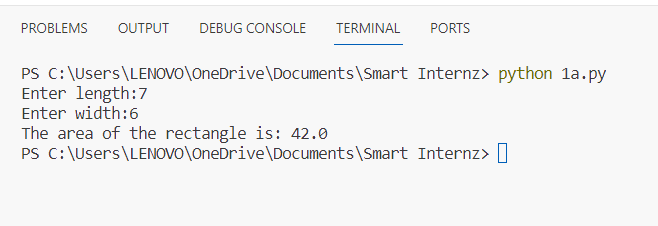
l = float(input("Enter length:"))

w= float(input("Enter width:"))

area = rect\_area(l, w)

print("Area of the rectangle:", area)

Output:



2. Write a program to convert miles to kilometres.

def miles\_to\_kilometers(miles):

    kilometers = miles \* 1.60934

    return kilometers

def main():

    miles = float(input("Distance in miles: "))

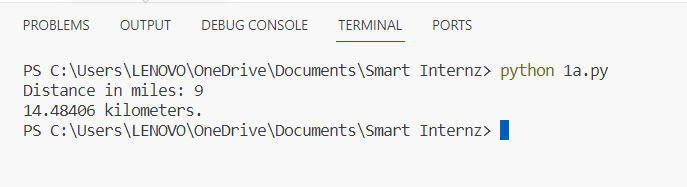
    kilometers = miles\_to\_kilometers(miles)

    print(f"{kilometers} kilometers.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



3. Write a function to check if a given string is a palindrome.

def palindrome(s):

    s = s.replace(" ", "").lower()

    return s == s[::-1]

def main():

    test\_string = input("Enter string:")

    if palindrome(test\_string):

        print("String is palindrome.")

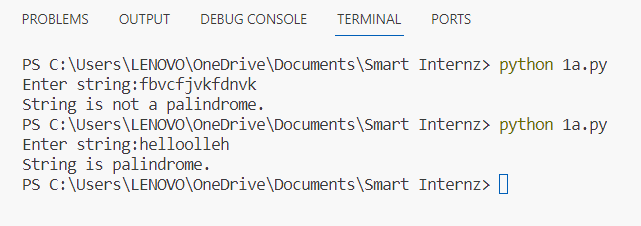
    else:

        print("String is not a palindrome.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



4. Write a Python program to find the second largest element in a list.

def second\_large(nums):

    if len(nums) < 2:

        return "List should have at least 2 elements"

    largest = second\_largest = float('-inf')

    for num in nums:

        if num > largest:

            second\_largest = largest

            largest = num

        elif num > second\_largest and num != largest:

            second\_largest = num

    return second\_largest

def main():

    nums = [int(x) for x in input("Enter elements in the list:").split()]

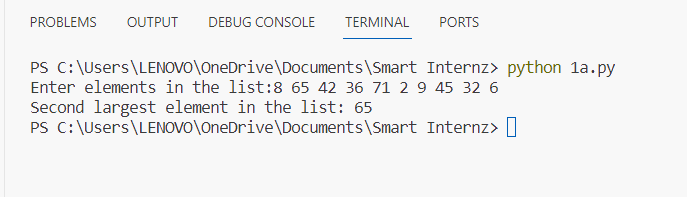
    second\_largest = second\_large(nums)

    print(f"Second largest element in the list: {second\_largest}")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



5. Explain what indentation means in Python.

* Indentation refers to the whitespace (spaces or tabs) that is used at the beginning of a line to define the grouping of statements. Indentation is not just for visual formatting; it is a fundamental part of Python's syntax and is used to indicate the structure and nesting of code blocks.

Indentation Rules:

* Python uses four spaces as default indentation spaces. However, the number of spaces can be anything; it is up to the user. But a minimum of one space is needed to indent a statement.
* The first line of Python code cannot have an indentation.
* Indentation is mandatory in Python to define the blocks of statements.
* The number of spaces must be uniform in a block of code.
* It is preferred to use whitespaces instead of tabs to indent in Python. Also, either use whitespace or tabs to indent; intermixing of tabs and whitespaces in indentation can cause wrong indentation errors.

Benefits of Indentation in Python:

* Indentation of code leads to better readability, although the primary reason for indentation in Python is to identify block structures.
* Missing, errors that sometimes pop up in C, and C++ languages can be avoided in Python and also the number of lines of code is reduced.

6. Write a program to perform a set difference operation.

def set\_difference(set1, set2):

    return set1.difference(set2)

def main():

    set1 = {4, 5, 6, 7, 8}

    set2 = {1, 2, 3, 4, 5}

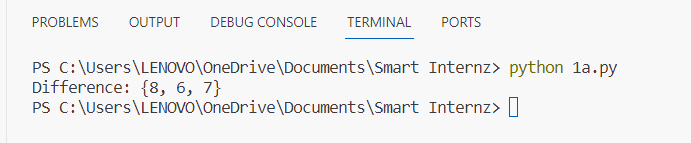
    difference = set\_difference(set1, set2)

    print("Difference:", difference)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



7. Write a Python program to print numbers from 1 to 10 using a while loop.

def print\_numbers():

    num = 1

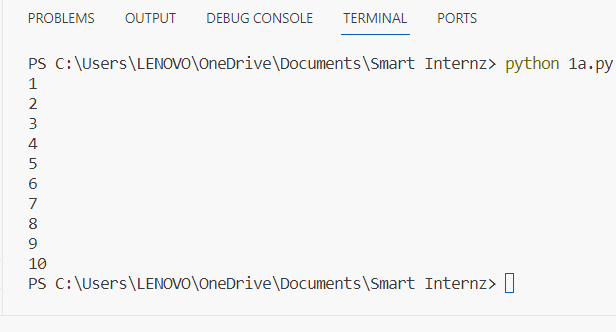
    while num <= 10:

        print(num)

        num += 1

print\_numbers()

Output:



8. Write a program to calculate the factorial of a number using a while loop.

def factorial\_num(n):

    factorial = 1

    while n > 1:

        factorial \*= n

        n -= 1

    return factorial

def main():

    number = int(input("Enter num:"))

    if number < 0:

        print("Factorial not defined for -ve  nums.")

    else:

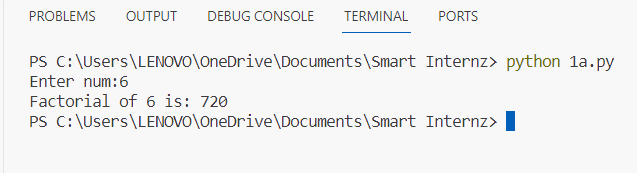
        factorial = factorial\_num(number)

        print(f"Factorial of {number} is: {factorial}")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



9. Write a Python program to check if a number is positive, negative, or zero using if-elif-else statements.

def check\_num(n):

    if n > 0:

        print("Positive.")

    elif n < 0:

        print("Negative.")

    else:

        print("Zero.")

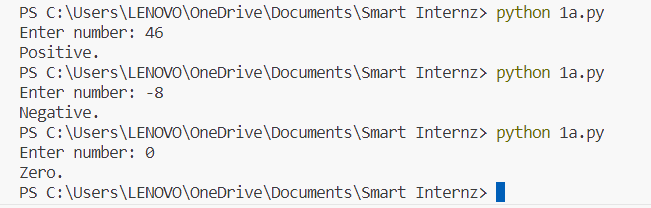
def main():

    number = float(input("Enter number: "))

    check\_num(number)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output: 

10. Write a program to determine the largest among three numbers using conditional statements.

def largest\_num(n1, n2, n3):

    if n1 >= n2 and n1 >= n3:

        largest = n1

    elif n2 >= n1 and n2 >= n3:

        largest = n2

    else:

        largest = n3

    return largest

def main():

    n1 = float(input("Enter num1: "))

    n2 = float(input("Enter num2: "))

    n3 = float(input("Enter num3: "))

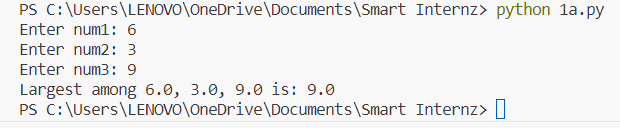
    largest = largest\_num(n1, n2, n3)

    print(f"Largest among {n1}, {n2}, {n3} is: {largest}")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



11. Write a Python program to create a numpy array filled with ones of the given shape.

import numpy as np

def create\_ones\_array(shape):

    return np.ones(shape)

def main():

    shape = tuple(map(int, input("Enter the shape of the array: ").split()))

    ones\_array = create\_ones\_array(shape)

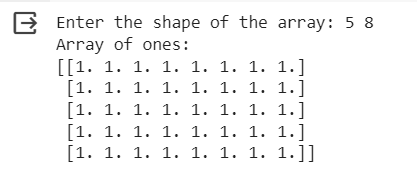
    print("Array of ones:")

    print(ones\_array)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



12. Write a program to create a 2D numpy array initialized with random integers.

import numpy as np

def array(rows, cols):

    return np.random.randint(low=0, high=100, size=(rows, cols))

def main():

    rows = int(input("Enter no.of rows:"))

    cols = int(input("Enter no.of columns:"))

    random\_array = array(rows, cols)

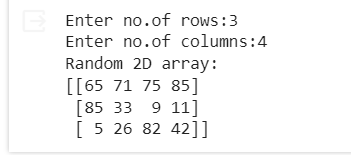
    print("Random 2D array:")

    print(random\_array)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



13. Write a Python program to generate an array of evenly spaced numbers over a specified range using linspace.

import numpy as np

def linspace(start, stop, num):

    return np.linspace(start, stop, num)

def main():

    start = float(input("Enter start value:"))

    stop = float(input("Enter stop value:"))

    num = int(input("Enter no.of values:"))

    linspace\_array = linspace(start, stop, num)

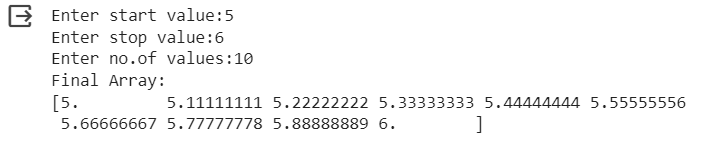
    print("Final Array:")

    print(linspace\_array)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



14. Write a program to generate an array of 10 equally spaced values between 1 and 100 using linspace.

import numpy as np

def array():

    return np.linspace(1, 100, 10)

def main():

    final\_array = array()

    print("Final Array:")

    print(final\_array)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



15. Write a Python program to create an array containing even numbers from 2 to 20 using arange.

import numpy as np

def array():

    return np.arange(2, 21, 2)

def main():

    even\_array = array()

    print("Final Array:")

    print(even\_array)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



16. Write a program to create an array containing numbers from 1 to 10 with a step size of 0.5 using arange.

import numpy as np

def create\_array():

    return np.arange(1, 10.5, 0.5)

def main():

    n\_array = create\_array()

    print("Final Array:")

    print(n\_array)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:

