**PROGRAM – 01**

**Program : Write a Python program to perform arithmetic operations on integers. 10**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

num1 = int(input("Enter the num1 : "))

num2 = int(input("Enter the num2 : "))

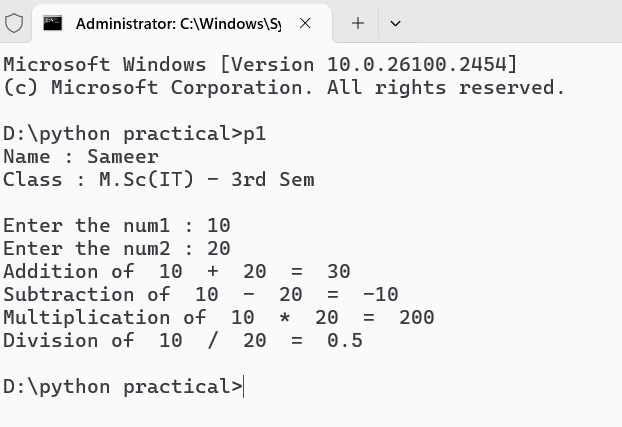
print("Addition of ", num1, " + ", num2, " = ", num1 + num2)

print("Subtraction of ", num1, " - ", num2, " = ", num1 - num2)

print("Multiplication of ", num1, " \* ", num2, " = ", num1 \* num2)

print("Division of ", num1, " / ", num2, " = ", num1 / num2)

**Output :-**

****

**PROGRAM – 02**

**Program : Write a Python program to check and print the types of at least 5 different inbuilt objects.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

a = 60

print("Type of ", a, "is ", type(a))

b = 2.5

print("Type of ", b, "is ", type(b))

c = True

print("Type of ", c, "is ", type(c))

d = "sameer"

print("Type of ", d, "is ", type(d))

e = [1, 2, 3, 'a', 'b', 'c']

print("Type of ", e, "is ", type(e))

**Output :-**

**A screenshot of a computer program

Description automatically generated**

**PROGRAM – 03**

**Program : Write a Python program to check if a number is EVEN or ODD.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

num1 = int(input("Enter the number : "))

if num1 % 2 == 0:

    print(num1, "is Even number.")

else:

    print(num1, "is Odd number.")

**Output :-**

A screenshot of a computer program

Description automatically generated

**PROGRAM – 04**

**Program : Write a Python program to check if a number is Positive, Negative or Zero.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

num1 = int(input("Enter the num1 : "))

if (num1 > 0):

    print(num1, " is a positive number.")

elif (num1 < 0):

    print(num1, " is a negative number.")

else:

    print(num1, " is zero.")

**Output :-**

**A screenshot of a computer program

Description automatically generated**

**PROGRAM – 05**

**Program : Write a Python program to check if a number is PRIME or NOT.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

num1 = int(input("Enter the number : "))

if num1 > 1:

    for i in range(2, num1):

        if num1 % i == 0:

            print(num1, "is not a prime number.")

            break

        else:

            print(num1, "is prime number.")

            break

else:

    print(num1, "is not a prime number.")

**Output :-**

A screenshot of a computer program

Description automatically generated

**PROGRAM – 06**

**Program : Write a Python program to check whether a string entered by the user is a valid decimal number or not.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

str = input("Enter the number : ")

try:

    no = float(str)

    print(str, "is a valid decimal number.")

except ValueError:

    print(str, "is not a valid decimal number.")

**Output :-**

**A screenshot of a computer

Description automatically generated**

**PROGRAM – 07**

**Program : Write a Python program to check if a year entered by the user is a Leap Year or NOT.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

num1 = int(input("Enter the year : "))

if ((num1 % 4 == 0 and num1 % 100 != 0) or (num1 % 400 == 0)):

    print(num1, "is a leap year.")

else:

    print(num1, "is not a leap year.")

Output :-

A screenshot of a computer

Description automatically generated

**PROGRAM – 08**

**Program : Write a Python program to check whether a string entered by the user is a palindrome or not.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

str = input("Enter a string : ")

rev\_str = str[ : : -1]

if str == rev\_str:

    print(str, "is a palindrome.")

else:

    print(str, "is not a palindrome.")

**Output :-**

**A screenshot of a computer

Description automatically generated**

**PROGRAM – 9**

**Program : Write a Python program to get a Decimal number from user and convert it into Binary, Octal and Hexadecimal.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem")

decimal = int(input("Enter a decimal number : "))

binary = bin(decimal)

octal = oct(decimal)

hexadecimal = hex(decimal)

print("Decimal : ", decimal)

print("Binary : ", binary[2:])

print("Octal : ", octal[2:])

print("Hexadecimal : ", hexadecimal[2:])

Output :-   
A screenshot of a computer program

Description automatically generated

**PROGRAM – 10**

**Program : Write a Python program to find sum of natural numbers, up to N.**

print("Name : Sameer\nClass : M.Sc(IT) 3rd sem\n")

num1 = int(input("Enter a integer : "))

sum = 0

for i in range(1, num1 + 1):

    sum += i

print("The sum of natural numbers up to ", num1, "is ", sum)

**Output :-**

**A screenshot of a computer program

Description automatically generated**

**PROGRAM – 11**

**Program : Write a Python program to get marks in five subjects from user and calculate average marks, percentage and grade of a student.**

print("Name : Sameer\nClass : M.Sc(IT) 3rd Sem")

marks = []

for i in range(1, 6):

    mark = int(input("Enter the subject marks (0-100): "))

    if 0 <= mark <= 100:

        marks.append(mark)

    else:

        print("Please enter valid marks between 0 and 100.")

total\_marks = sum(marks)

print("Total marks is : ", total\_marks)

average\_marks = total\_marks/5

print("Average marks is : ", average\_marks)

percentage = (total\_marks/500) \* 100

print("Percentage is : ", percentage)

if percentage >= 80:

    print("Grange A")

elif percentage >= 70:

    print("Grade B")

elif percentage >= 60:

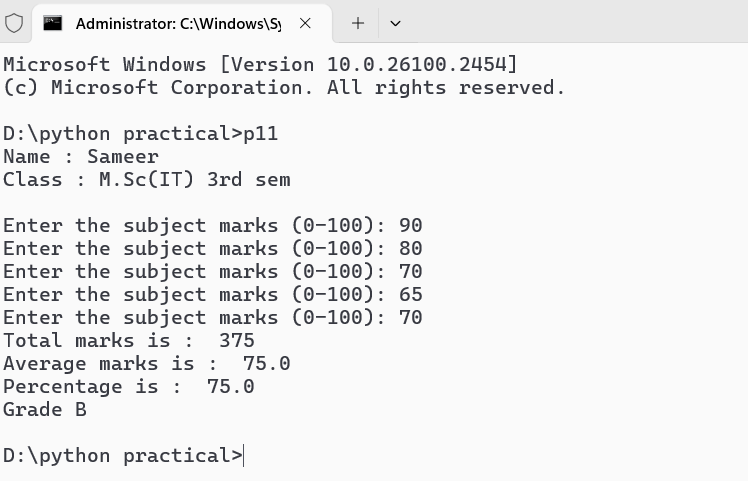
    print("Grade C")

elif percentage >= 33:

    print("Grade D")

else:

    print("Fail")

Output :-   


**PROGRAM – 12**

**Program : Write a Python program to get a number and find the sum and product of its digits.**

def sum\_of\_digits(number):

    digits = str(number)

    digit\_sum = 0

    for digit in digits:

        digit\_value = int(digit)

        digit\_sum += digit\_value

    return digit\_sum

def product\_of\_digits(number):

    digits = str(number)

    digit\_product = 1

    for digit in digits:

        digit\_value = int(digit)

        digit\_product \*= digit\_value

    return digit\_product

try:

    print("Name : Sameer\nClass : M.Sc(IT) 3rd Sem\n")

    user\_input = int(input("Enter a number: "))

    total\_sum = sum\_of\_digits(user\_input)

    total\_product = product\_of\_digits(user\_input)

    print(f"Sum of digits: {total\_sum}")

    print(f"Product of digits: {total\_product}")

except ValueError:

    print("Please enter a valid integer.")

**Output :-**

**A screenshot of a computer

Description automatically generated**

**PROGRAM – 13**

**Program : Write a Python program to get two integers and find their GCD and LCM.**

def gcd(a, b):

    while b:

        a, b = b, a % b

    return a

def lcm(a, b):

    return abs(a \* b) // gcd(a, b)

def main():

    try:

        print("Name : Sameer\nClass : M.Sc(IT) 3rd Sem\n")

        num1 = int(input("Enter the first integer: "))

        num2 = int(input("Enter the second integer: "))

        gcd\_result = gcd(num1, num2)

        lcm\_result = lcm(num1, num2)

        print(f"The GCD of {num1} and {num2} is: {gcd\_result}")

        print(f"The LCM of {num1} and {num2} is: {lcm\_result}")

    except ValueError:

        print("Please enter valid integers.")

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

    main()

Output :-

A screenshot of a computer

Description automatically generated

**PROGRAM – 14**

**Program : Write a Python program to find factorial of a number using while loop.**

def factorial(n):

    if n < 0:

        return "Factorial is not defined for negative numbers."

    result = 1

    count = 1

    while count <= n:

        result \*= count

        count += 1

    return result

def main():

    try:

        num = int(input("Enter a non-negative integer: "))

        fact = factorial(num)

        print(f"The factorial of {num} is: {fact}")

    except ValueError:

        print("Please enter a valid integer.")

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

    print("Name : Sameer\nClass : M.Sc(IT) 3rd Sem\n")

    main()

**Output :-**

A screenshot of a computer program

Description automatically generated

**PROGRAM – 15**

**Program : Write a Python program to print Fibonacci series up to N terms.**

def fibonacci(n):

    if n <= 0:

        return 0

    elif n == 1:

        return 1

    else:

        return fibonacci(n - 1) + fibonacci(n - 2)

def print\_fibonacci\_series(n):

    for i in range(n):

        print(fibonacci(i), end=' ')

def main():

    try:

        num\_terms = int(input("Enter the number of terms for Fibonacci series: "))

        print(f"Fibonacci series up to {num\_terms} terms: ")

        print\_fibonacci\_series(num\_terms)

    except ValueError:

        print("Please enter a valid integer.")

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

    print("Name : Sameer\nClass : M.Sc(IT) 3rd Sem\n")

    main()

**Output :-**

A screenshot of a computer

Description automatically generated

**PROGRAM – 16**

**Program : Write a Python program to print multiplication table.**

def multiplication(n):

    for i in range(1, 10):

        print(n, "\*", i, "=", n \* i)

def main():

    n = int(input("Enter the value of n : "))

    multiplication(n)

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**

**A screenshot of a computer

Description automatically generated**

**PROGRAM – 17**

**Program : Write a Python program to access each element of a string in forward and backward orders using the ‘while’ loop.**

def access\_string\_elements(s):

    print("Forward order:")

    index = 0

    while index < len(s):

        print(s[index], end=' ')

        index += 1

    print()

    print("Backward order:")

    index = len(s) - 1

    while index >= 0:

        print(s[index], end=' ')

        index -= 1

    print()

def main():

    user\_input = input("Enter a string: ")

    access\_string\_elements(user\_input)

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**

**A screenshot of a computer

Description automatically generated**

**PROGRAM – 18**

**Program : Write a Python program to access each element of a string in forward and backward orders using the ‘for’ loop.**

def access\_string\_elements(s):

    print("Forward order:")

    for char in s:

        print(char, end=' ')

    print()

    print("Backward order:")

    for char in reversed(s):

        print(char, end=' ')

    print()

def main():

    user\_input = input("Enter a string: ")

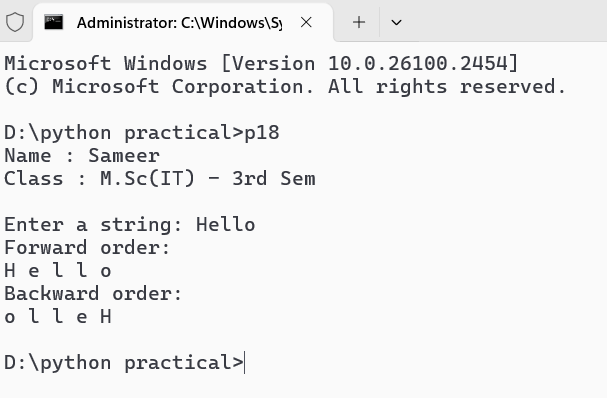
    access\_string\_elements(user\_input)

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**

****

**PROGRAM – 19**

**Program : Write a Python program to find whether a substring exists in main string or not.**

def check\_substring(main\_string, substring):

    if substring in main\_string:

        return True

    else:

        return False

def main():

    main\_string = input("Enter the main string: ")

    substring = input("Enter the substring to search for: ")

    if check\_substring(main\_string, substring):

        print(f"The substring '{substring}' exists in the main string.")

    else:

        print(f"The substring '{substring}' does not exist in the main string.")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**

**A screenshot of a computer

Description automatically generated**

**PROGRAM – 20**

**Program : Write a Python program to find the first occurrence of a substring in the main string.**

def find\_first\_occurrence(main\_string, substring):

    index = main\_string.find(substring)

    if index != -1:

        print(f"The first occurrence of '{substring}' in '{main\_string}' is at index: {index}")

    else:

        print(f"The substring '{substring}' was not found in '{main\_string}'.")

def main():

    main\_string = input("Enter the main string: ")

    substring = input("Enter the substring to find: ")

    find\_first\_occurrence(main\_string, substring)

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**

**A screenshot of a computer

Description automatically generated**

**PROGRAM – 21**

**Program : Write a Python program to count the number of times a substring appears in the main string.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

main\_string = input("Enter the main string: ")

substring = input("Enter the substring to count: ")

count = 0

for i in range(len(main\_string) - len(substring) + 1):

    if main\_string[i:i + len(substring)] == substring:

        count += 1

print(f"The substring '{substring}' appears {count} times in '{main\_string}'.")

**Outptut :-**

**A screenshot of a computer

Description automatically generated**

**PROGRAM – 22**

**Program : Write a Python program to demonstrate the use of all “casing” methods and display a string in different cases.**

def demonstrate\_string\_casing(input\_string):

    print("Original String: ", input\_string)

    print("Lowercase: ", input\_string.lower())

    print("Uppercase: ", input\_string.upper())

    print("Title Case: ", input\_string.title())

    print("Capitalized: ", input\_string.capitalize())

    print("Swapcase: ", input\_string.swapcase())

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    example\_string = "hello, World! Welcome to Python Programming."

    demonstrate\_string\_casing(example\_string)

**Output :-**

**A screenshot of a computer program

Description automatically generated**

**PROGRAM – 23**

**Program : Write a Python program to demonstrate the use of all string testing {isXXX()} methods.**

def demonstrate\_string\_testing\_methods(input\_string):

    print(f"Testing the string: '{input\_string}'")

    print("isalpha(): ", input\_string.isalpha())

    print("isdigit(): ", input\_string.isdigit())

    print("isalnum(): ", input\_string.isalnum())

    print("isspace(): ", input\_string.isspace())

    print("islower(): ", input\_string.islower())

    print("isupper(): ", input\_string.isupper())

    print("istitle(): ", input\_string.istitle())

    print("\n")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    test\_strings = [

        "HelloWorld",

        "12345",

        "Hello123",

        "   ",

        "hello",

        "HELLO",

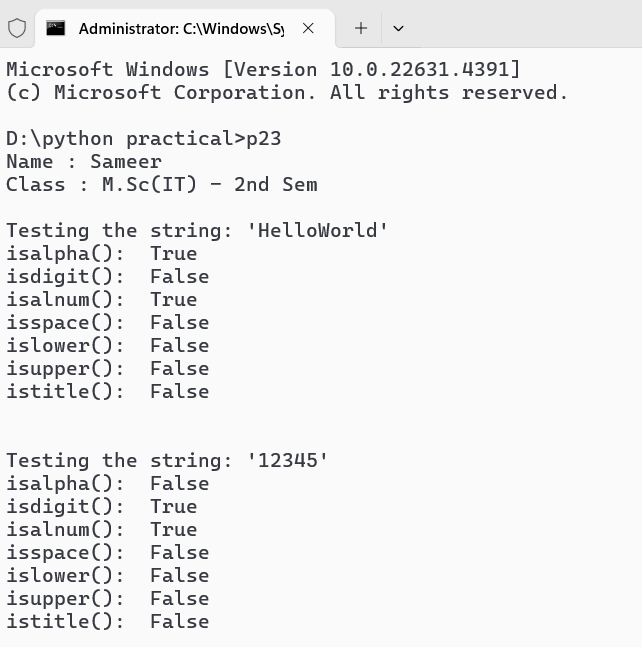
        "Hello World"

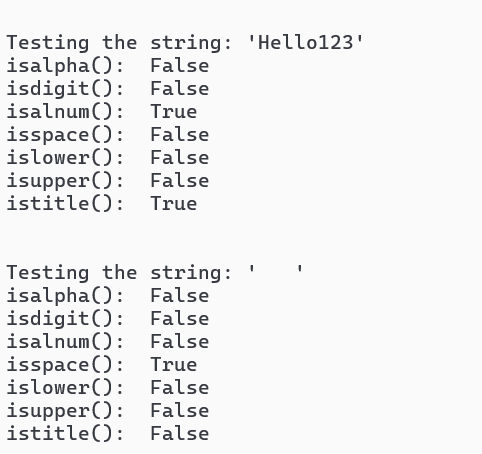
    ]

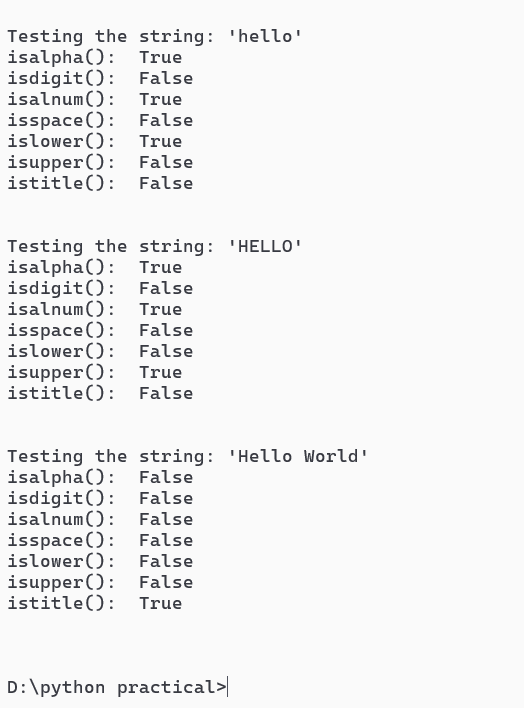
    for test\_string in test\_strings:

        demonstrate\_string\_testing\_methods(test\_string)

**Output :-**

****

****

****

**PROGRAM – 24**

**Program : Write a Python function to take a list of integers as input and return the average.**

def calculate\_average(numbers):

    total\_number = sum(numbers)

    avg = total\_number/len(numbers)

    return avg

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

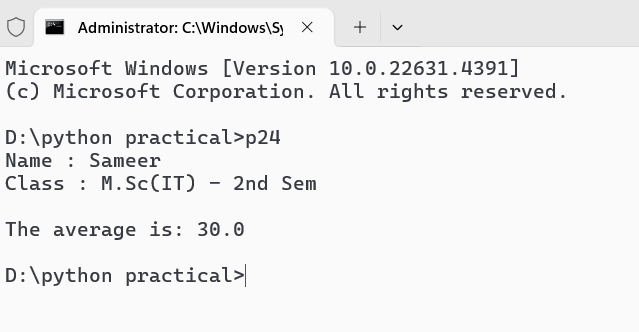
    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    numbers = [10, 20, 30, 40, 50]

    average = calculate\_average(numbers)

    print("The average is:", average)

**Output :-**

****

**PROGRAM – 25**

**Program : Write a Python function to take two distinct integers as input and print all prime numbers between them.**

def is\_prime(num):

    if num <= 1:

        return False

    for i in range(2, int(num\*\*0.5) + 1):

        if num % i == 0:

            return False

    return True

def print\_primes\_between(a, b):

    if a > b:

        a, b = b, a

    print(f"Prime numbers between {a} and {b}:")

    for num in range(a + 1, b):

        if is\_prime(num):

            print(num, end=' ')

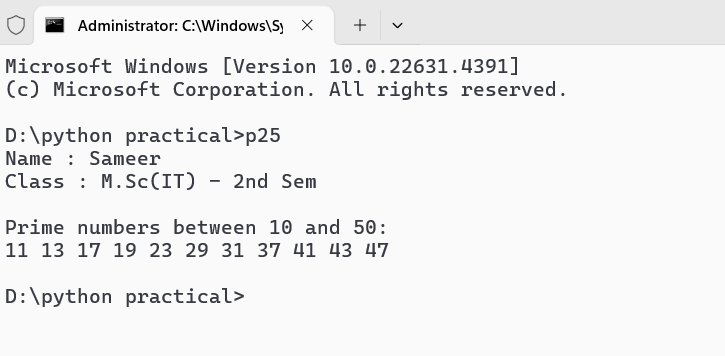
    print()

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    print\_primes\_between(10, 50)

**Output :-**

****

**PROGRAM – 26**

**Program : Write a Python function to take two integers as input and return both their sum and product.**

def sum\_and\_product(a, b):

    total\_sum = a + b

    total\_product = a \* b

    return total\_sum, total\_product

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    num1 = 5

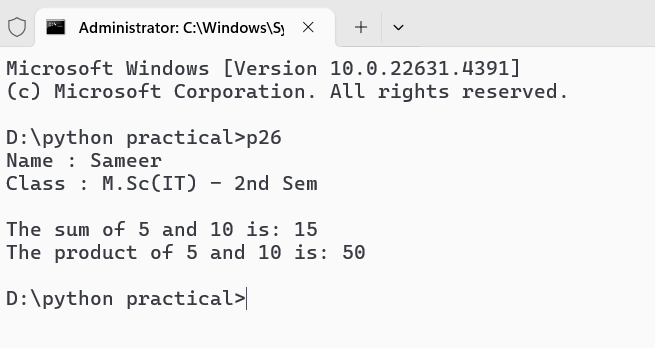
    num2 = 10

    result\_sum, result\_product = sum\_and\_product(num1, num2)

    print(f"The sum of {num1} and {num2} is: {result\_sum}")

    print(f"The product of {num1} and {num2} is: {result\_product}")

**Output :-**

****

**PROGRAM – 27**

**Program : Write a Python program to demonstrate the positional arguments of a function.**

def describe\_pet(pet\_name, pet\_type):

    """Display information about a pet."""

    print(f"I have a {pet\_type} named {pet\_name}.")

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

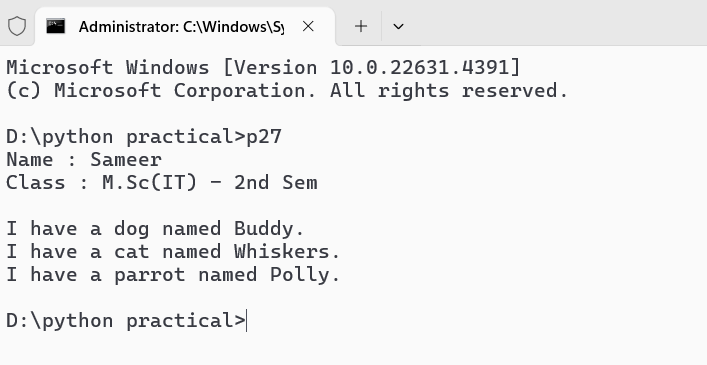
    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    describe\_pet("Buddy", "dog")

    describe\_pet("Whiskers", "cat")

    describe\_pet("Polly", "parrot")

**Output :-**



**PROGRAM – 28**

**Program : Write a Python program to demonstrate the keyword arguments of a function.**

def describe\_pet(pet\_name, pet\_type='dog'):

    print(f"I have a {pet\_type} named {pet\_name}.")

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

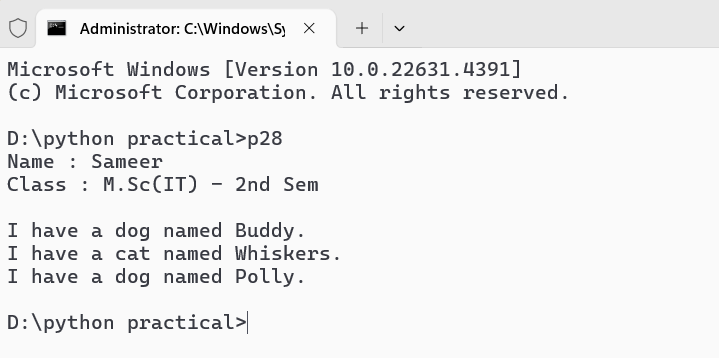
    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    describe\_pet(pet\_name="Buddy", pet\_type="dog")

    describe\_pet(pet\_type="cat", pet\_name="Whiskers")

    describe\_pet(pet\_name="Polly")

**Output :-**

****

**PROGRAM – 29**

**Program : Write a Python program to demonstrate the default arguments of a function.**

def describe\_pet(pet\_name, pet\_type='dog'):

    print(f"I have a {pet\_type} named {pet\_name}.")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

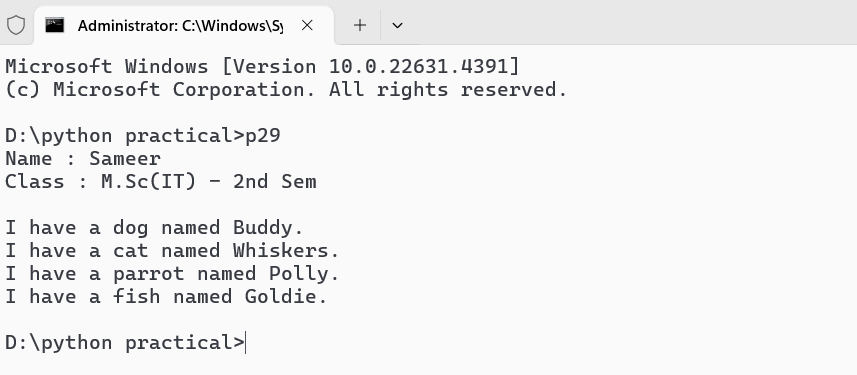
    describe\_pet("Buddy")

    describe\_pet("Whiskers", "cat")

    describe\_pet("Polly", "parrot")

    describe\_pet("Goldie", "fish")

**Output :-**

****

**PROGRAM – 30**

**Program : Write a Python function to demonstrate variable length arguments.**

def demonstrate\_variable\_length\_args(\*args, \*\*kwargs):

    print("Non-keyword arguments (args):")

    for arg in args:

        print(arg)

    print("\nKeyword arguments (kwargs):")

    for key, value in kwargs.items():

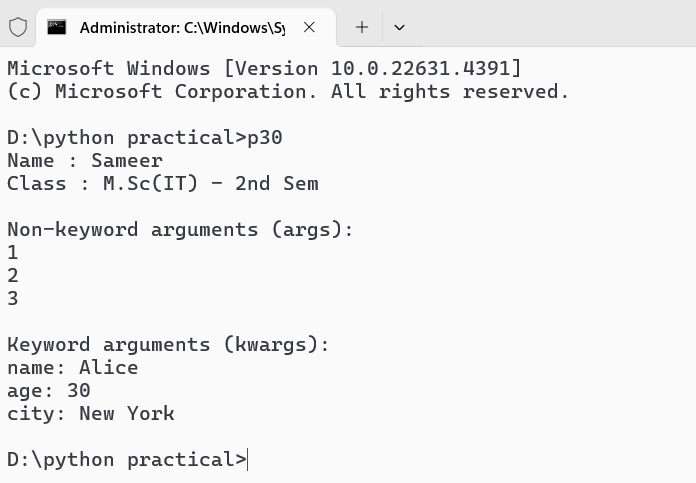
        print(f"{key}: {value}")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    demonstrate\_variable\_length\_args(1, 2, 3, name="Alice", age=30, city="New York")

**Output :-**

****

**PROGRAM – 31**

**Program : Write a Python function to demonstrate keyword variable length arguments.**

def display\_keyword\_arguments(\*\*kwargs):

    print("Keyword arguments:")

    for key, value in kwargs.items():

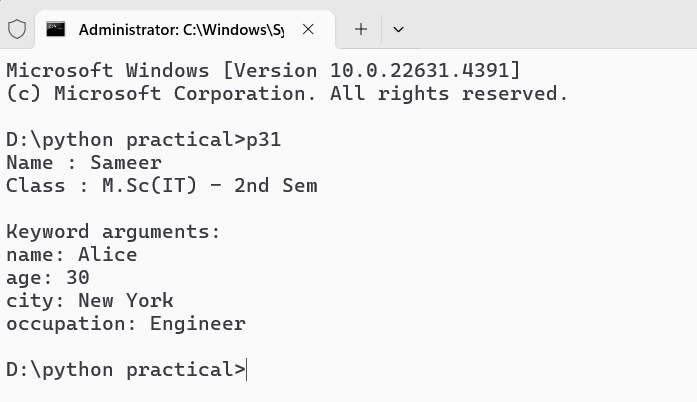
        print(f"{key}: {value}")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    display\_keyword\_arguments(name="Alice", age=30, city="New York", occupation="Engineer")

**Output :-**

****

**PROGRAM – 32**

**Program : Write a Python program to demonstrate global and local variables.**

global\_var = "I am a global variable"

def demonstrate\_variables():

    global global\_var

    local\_var = "I am a local variable"

    print(local\_var)

    print(global\_var)

    global\_var = "I have been modified globally"

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    print("Before function call:")

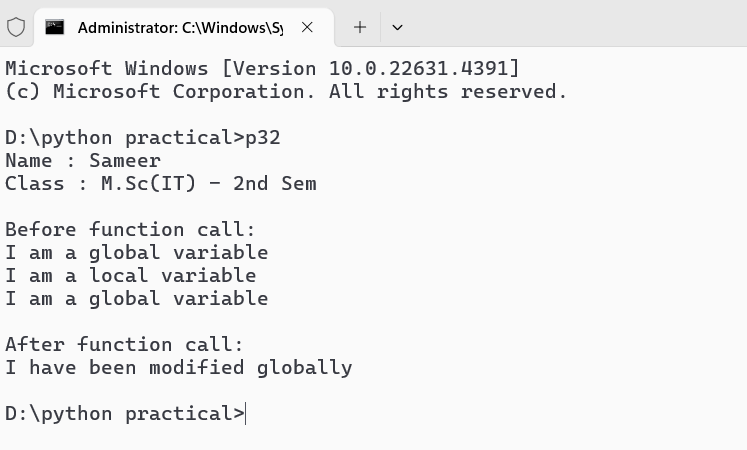
    print(global\_var)

    demonstrate\_variables()

    print("\nAfter function call:")

    print(global\_var)

**Output :-**

****

**PROGRAM – 33**

**Program : Write a Python function that takes an integer as input and calculates its factorial using recursion.**

def factorial(num):

    if num == 0 or num == 1:

        return 1

    else:

        return num \* factorial(num - 1)

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

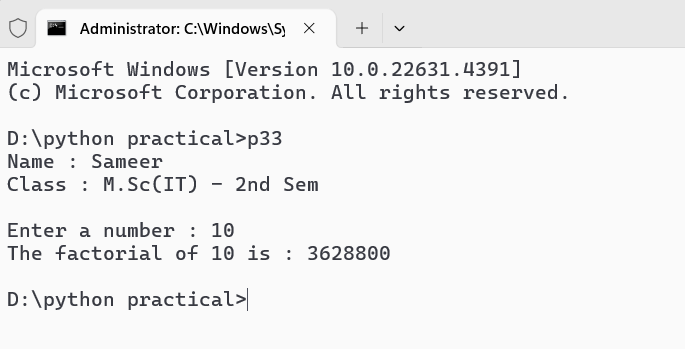
    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    num = int(input("Enter a number : "))

    fac = factorial(num)

    print(f"The factorial of {num} is : {fac}")

**Output :-**

****

**PROGRAM – 34**

**Program : Write a Python program to demonstrate the use of lambda functions.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

add = lambda x, y: x + y

print("Addition using lambda function:", add(5, 3))

numbers = [1, 2, 3, 4, 5]

squared\_numbers = list(map(lambda x: x \*\* 2, numbers))

print("Squared numbers using map and lambda:", squared\_numbers)

even\_numbers = list(filter(lambda x: x % 2 == 0, numbers))

print("Even numbers using filter and lambda:", even\_numbers)

from functools import reduce

product = reduce(lambda x, y: x \* y, numbers)

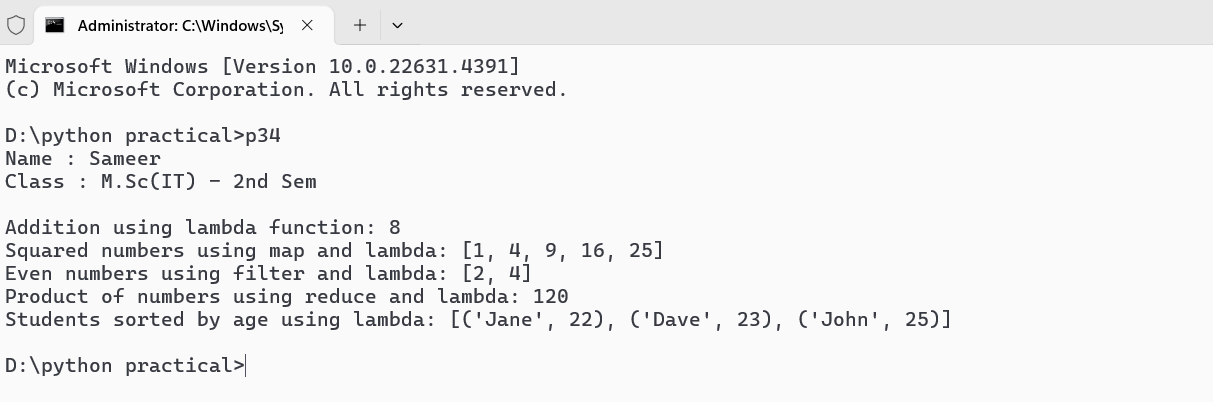
print("Product of numbers using reduce and lambda:", product)

students = [("John", 25), ("Jane", 22), ("Dave", 23)]

sorted\_students = sorted(students, key=lambda student: student[1])

print("Students sorted by age using lambda:", sorted\_students)

**Output :-**



**PROGRAM – 35**

**Program : Write a Python program to demonstrate the use of lambda functions and map.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

numbers = [1, 2, 3, 4, 5]

squared\_numbers = list(map(lambda x: x \*\* 2, numbers))

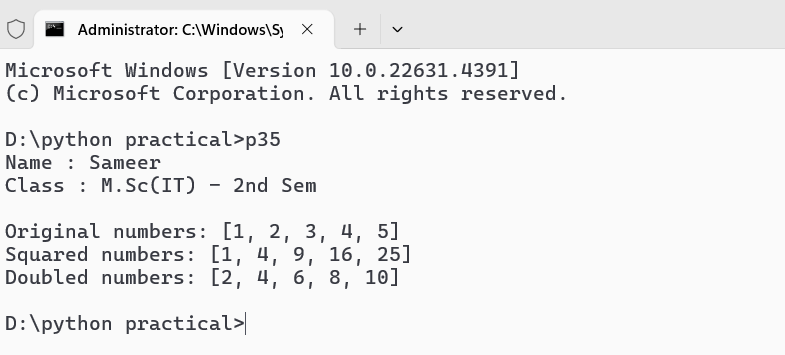
doubled\_numbers = list(map(lambda x: x \* 2, numbers))

print("Original numbers:", numbers)

print("Squared numbers:", squared\_numbers)

print("Doubled numbers:", doubled\_numbers)

**Output :-**

****

**PROGRAM – 36**

**Program : Write a Python program to demonstrate the use of lambda functions and reduce.**

from functools import reduce

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

numbers = [1, 2, 3, 4, 5]

sum\_of\_numbers = reduce(lambda x, y: x + y, numbers)

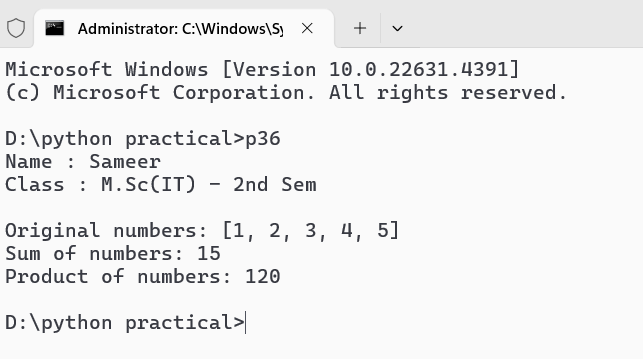
product\_of\_numbers = reduce(lambda x, y: x \* y, numbers)

print("Original numbers:", numbers)

print("Sum of numbers:", sum\_of\_numbers)

print("Product of numbers:", product\_of\_numbers)

**Output :-**

****

**PROGRAM – 37**

**Program : Write a Python program to demonstrate the various list processing methods.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

fruits = ['apple', 'banana', 'cherry', 'date']

fruits.append('elderberry')

print("After append:", fruits)

fruits.insert(1, 'blueberry')

print("After insert:", fruits)

fruits.remove('banana')

print("After remove:", fruits)

popped\_fruit = fruits.pop()

print("Popped fruit:", popped\_fruit)

print("After pop:", fruits)

first\_fruit = fruits[0]

print("First fruit:", first\_fruit)

sliced\_fruits = fruits[1:3]

print("Sliced fruits (index 1 to 2):", sliced\_fruits)

fruits.sort()

print("After sorting:", fruits)

fruits.reverse()

print("After reversing:", fruits)

fruit\_lengths = [len(fruit) for fruit in fruits]

print("Lengths of fruit names:", fruit\_lengths)

is\_apple\_present = 'apple' in fruits

print("Is apple present?", is\_apple\_present)

count\_of\_cherry = fruits.count('cherry')

print("Count of cherry:", count\_of\_cherry)

more\_fruits = ['fig', 'grape']

fruits.extend(more\_fruits)

print("After extending:", fruits)

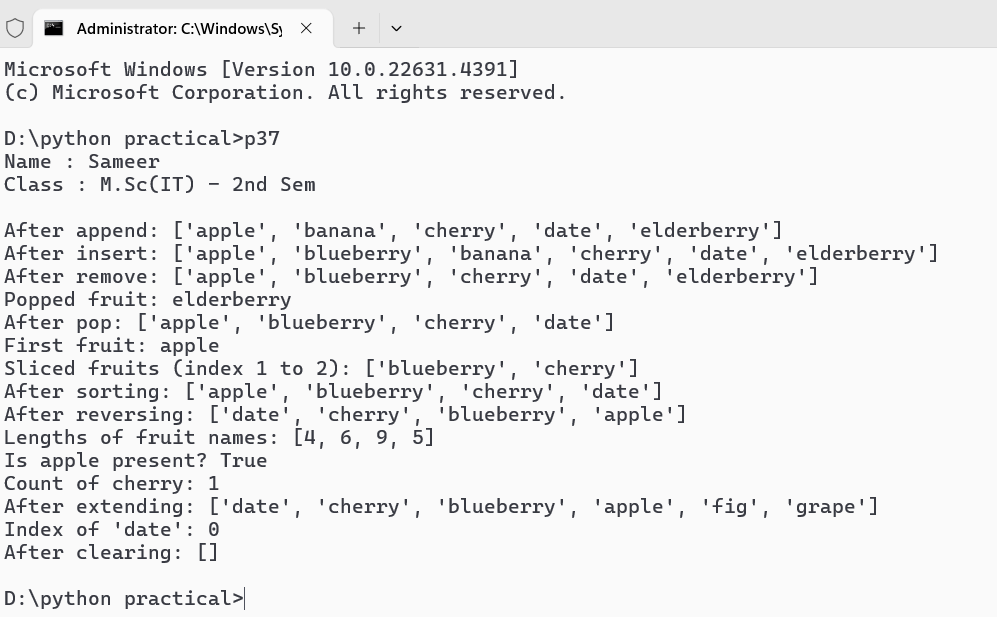
index\_of\_date = fruits.index('date')

print("Index of 'date':", index\_of\_date)

fruits.clear()

print("After clearing:", fruits)

**Output :-**

****

**PROGRAM – 38**

**Program : Write a Python program to find the biggest and smallest numbers in a list of integers.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

numbers = [34, 67, 23, 89, 12, 45, 78, 90, 3, 56]

biggest\_number = max(numbers)

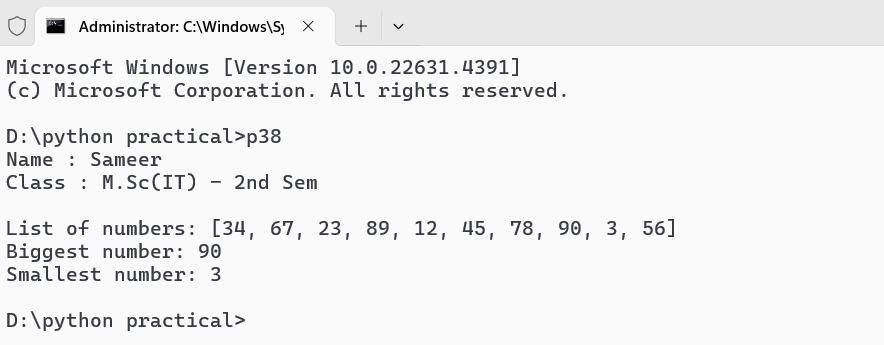
smallest\_number = min(numbers)

print("List of numbers:", numbers)

print("Biggest number:", biggest\_number)

print("Smallest number:", smallest\_number)

**Output :-**

****

**PROGRAM – 39**

**Program : Write a Python program to find common elements in two lists.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

list1 = [1, 2, 3, 4, 5, 6]

list2 = [4, 5, 6, 7, 8, 9]

common\_elements\_set = set(list1).intersection(set(list2))

common\_elements\_list = [element for element in list1 if element in list2]

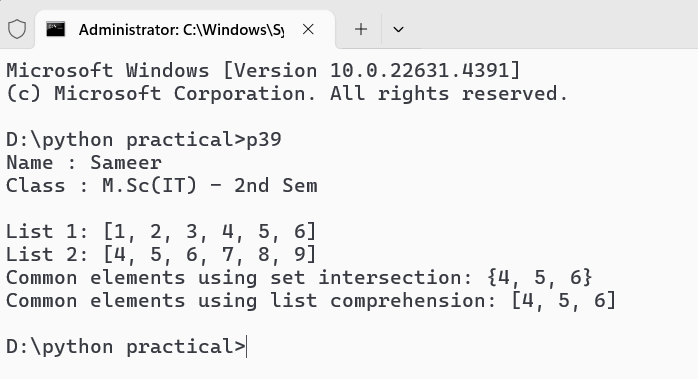
print("List 1:", list1)

print("List 2:", list2)

print("Common elements using set intersection:", common\_elements\_set)

print("Common elements using list comprehension:", common\_elements\_list)

**Output :-**

****

**PROGRAM – 40**

**Program : Write a Python program to demonstrate the various tuple processing methods.**

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

tuple1 = (1, 2, 3, 4, 5)

tuple2 = ('apple', 'banana', 'cherry')

tuple3 = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

print("Tuple 1:", tuple1)

print("Tuple 2:", tuple2)

print("\nAccessing elements:")

print("First element of tuple1:", tuple1[0])

print("Last element of tuple2:", tuple2[-1])

print("\nSlicing tuples:")

print("Slice of tuple1 (from index 1 to 3):", tuple1[1:4])

print("Slice of tuple3 (first 5 elements):", tuple3[:5])

print("\nConcatenating tuples:")

tuple4 = tuple1 + tuple2

print("Concatenated tuple (tuple1 + tuple2):", tuple4)

print("\nRepeating tuples:")

tuple5 = tuple2 \* 2

print("Repeated tuple2 (2 times):", tuple5)

print("\nMembership check:")

print("'apple' in tuple2:", 'apple' in tuple2)

print("'grape' in tuple2:", 'grape' in tuple2)

print("\nLength of tuples:")

print("Length of tuple1:", len(tuple1))

print("Length of tuple2:", len(tuple2))

print("\nCounting occurrences:")

print("Count of 2 in tuple1:", tuple1.count(2))

print("Count of 'banana' in tuple2:", tuple2.count('banana'))

print("\nFinding index of an element:")

print("Index of 3 in tuple1:", tuple1.index(3))

print("Index of 'cherry' in tuple2:", tuple2.index('cherry'))

print("\nConverting tuple to list:")

list\_from\_tuple1 = list(tuple1)

print("List from tuple1:", list\_from\_tuple1)

print("\nConverting list to tuple:")

list\_example = [10, 20, 30]

tuple\_from\_list = tuple(list\_example)

print("Tuple from list\_example:", tuple\_from\_list)

print("\nNested tuples:")

nested\_tuple = (tuple1, tuple2)

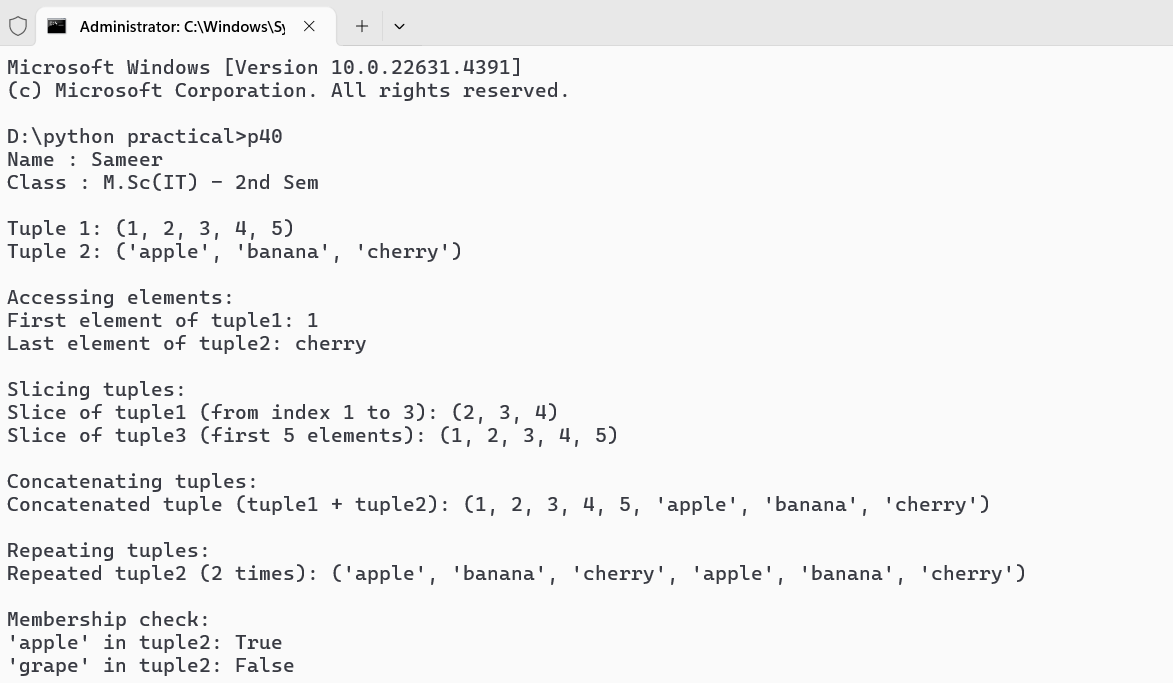
print("Nested tuple:", nested\_tuple)

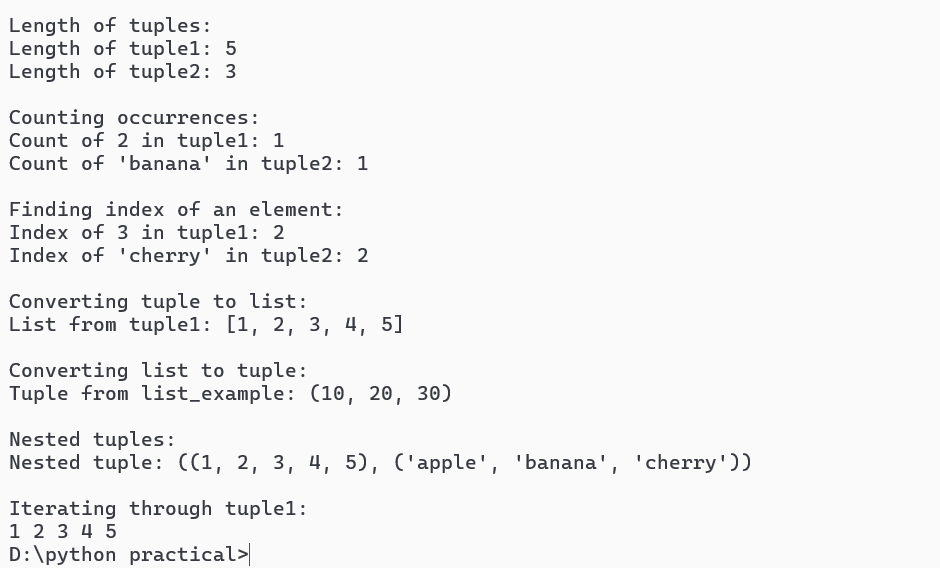
print("\nIterating through tuple1:")

for item in tuple1:

    print(item, end=' ')

**Output :-**

****

****

**PROGRAM – 41**

**Program : Write a Python program to demonstrate the use of dictionaries.**

def display\_menu():

    print("\nContact Book Menu:")

    print("1. Add Contact")

    print("2. View Contacts")

    print("3. Delete Contact")

    print("4. Exit")

def add\_contact(contact\_book):

    name = input("Enter contact name: ")

    phone = input("Enter contact phone number: ")

    contact\_book[name] = phone

    print(f"Contact '{name}' added.")

def view\_contacts(contact\_book):

    if contact\_book:

        print("\nContacts:")

        for name, phone in contact\_book.items():

            print(f"{name}: {phone}")

    else:

        print("No contacts available.")

def delete\_contact(contact\_book):

    name = input("Enter the name of the contact to delete: ")

    if name in contact\_book:

        del contact\_book[name]

        print(f"Contact '{name}' deleted.")

    else:

        print(f"Contact '{name}' not found.")

def main():

    contact\_book = {}

    while True:

        display\_menu()

        choice = input("Choose an option (1-4): ")

        if choice == '1':

            add\_contact(contact\_book)

        elif choice == '2':

            view\_contacts(contact\_book)

        elif choice == '3':

            delete\_contact(contact\_book)

        elif choice == '4':

            print("Exiting the contact book. Goodbye!")

            break

        else:

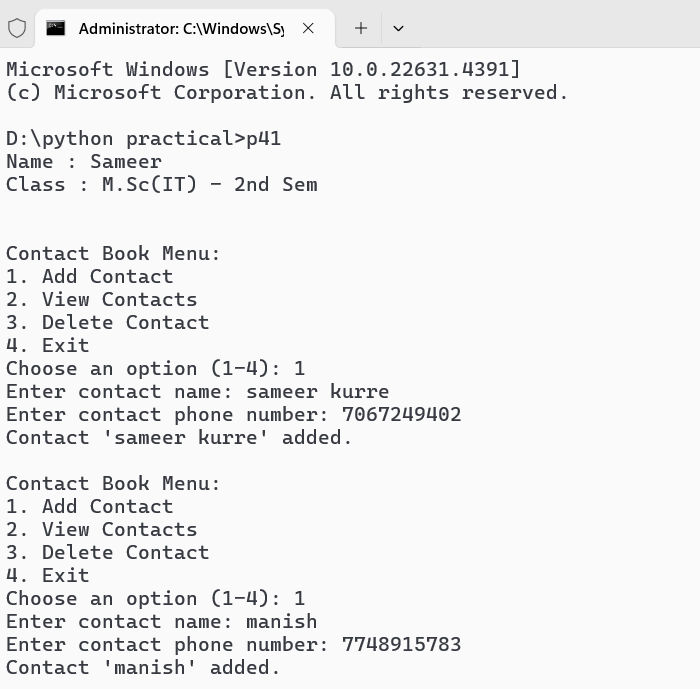
            print("Invalid choice. Please select a valid option.")

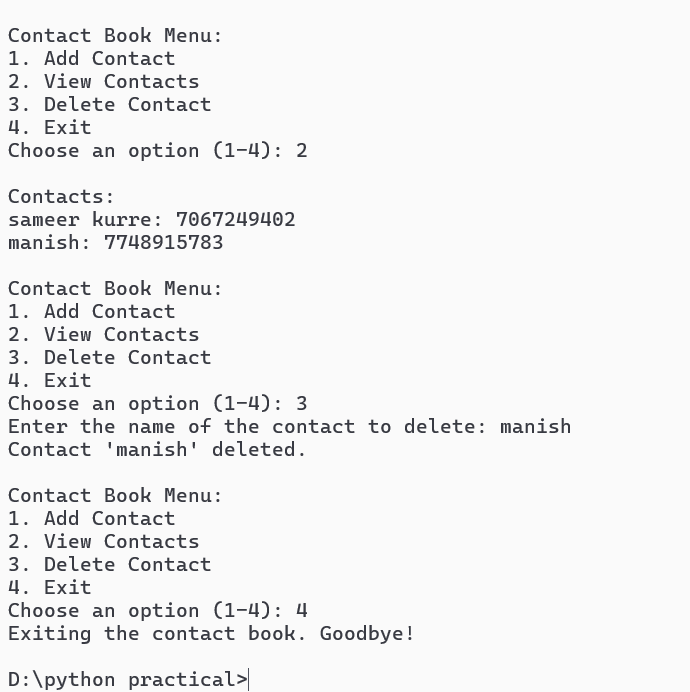
if \_\_name\_\_ == "\_\_main\_\_":

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**





**PROGRAM – 42**

**Program : Write a Python program to find the number of occurrences of each letter in a string using dictionaries.**

def count\_letters(input\_string):

    letter\_count = {}

    input\_string = input\_string.lower()

    for char in input\_string:

        if char.isalpha():

            if char in letter\_count:

                letter\_count[char] += 1

            else:

                letter\_count[char] = 1

    return letter\_count

def main():

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

    letter\_counts = count\_letters(input\_string)

    print("\nLetter occurrences:")

    for letter, count in letter\_counts.items():

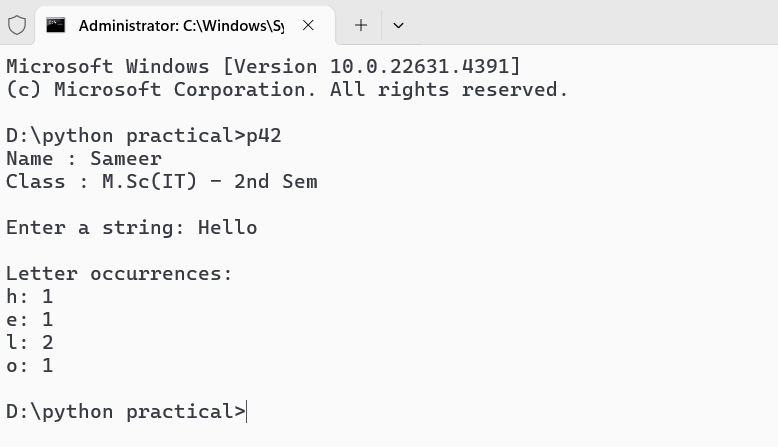
        print(f"{letter}: {count}")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**



**PROGRAM – 43**

**Program : Write a Python program to print the CWD and change the CWD.**

**PROGRAM – 44**

**Program : Write a Python program that takes a list of words from the user and writes them into a file. The program should stop when the user enters the word ‘quit’.**

def main():

    filename = "words.txt"

    with open(filename, 'w') as file:

        print("Enter words to write to the file. Type 'quit' to stop.")

        while True:

            word = input("Enter a word: ")

            if word.lower() == 'quit':

                print("Exiting the program.")

                break

            else:

                file.write(word + '\n')

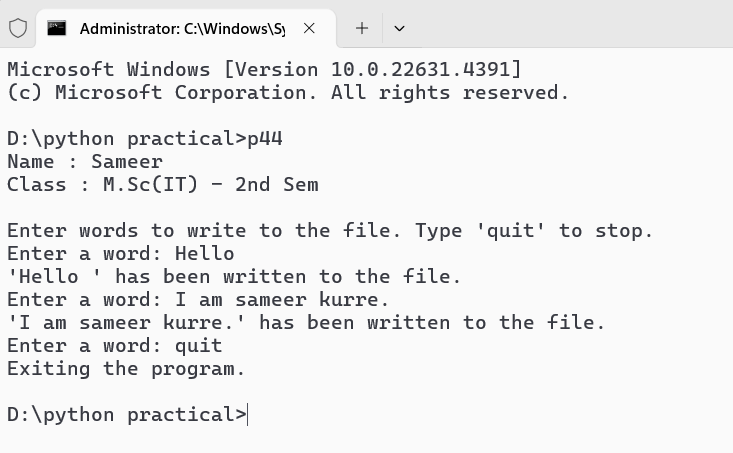
                print(f"'{word}' has been written to the file.")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**

****

**PROGRAM – 45**

**Program : Write a Python program that reads a file in text mode and counts the number of words that contain anyone of the letters [‘w’, ‘o’, ‘r’, ‘d’, ‘s’].**

def count\_words\_with\_specific\_letters(file\_path):

    letters\_to\_check = {'w', 'o', 'r', 'd', 's'}

    count = 0

    try:

        with open(file\_path, 'r', encoding='utf-8') as file:

            for line in file:

                words = line.split()

                for word in words:

                    if any(letter in word.lower() for letter in letters\_to\_check):

                        count += 1

    except FileNotFoundError:

        print(f"The file '{file\_path}' was not found.")

        return

    except Exception as e:

        print(f"An error occurred: {e}")

        return

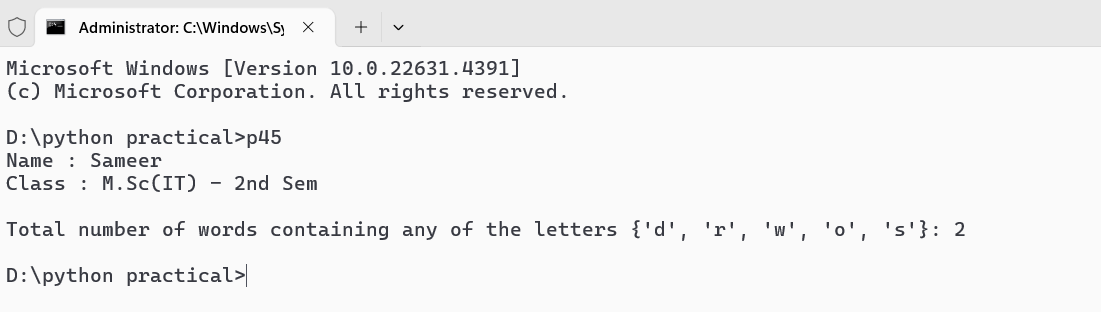
    print(f"Total number of words containing any of the letters {letters\_to\_check}: {count}")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    count\_words\_with\_specific\_letters('p45.txt')

**Output :-**

****

**PROGRAM – 46**

**Program : Python programs to demonstrate the creation and use of “modules”.**

Math\_operations.py :-   
def add(a, b):

    return a + b

def subtract(a, b):

    return a - b

def multiply(a, b):

    return a \* b

def divide(a, b):

    if b == 0:

        return "Cannot divide by zero"

    return a / b

p46.py :-   
import math\_operations

def main():

    a = 10

    b = 5

    print(f"Adding {a} and {b}: {math\_operations.add(a, b)}")

    print(f"Subtracting {b} from {a}: {math\_operations.subtract(a, b)}")

    print(f"Multiplying {a} and {b}: {math\_operations.multiply(a, b)}")

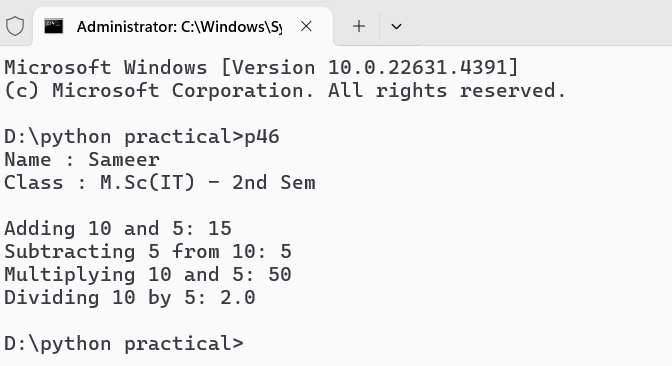
    print(f"Dividing {a} by {b}: {math\_operations.divide(a, b)}")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    main()

**Output :-**

****

**PROGRAM – 47**

**Program : Exception Handling Program that uses try and except.**

def divide\_numbers():

    try:

        num1 = float(input("Enter the first number (numerator): "))

        num2 = float(input("Enter the second number (denominator): "))

        result = num1 / num2

    except ZeroDivisionError:

        print("Error: Cannot divide by zero. Please enter a non-zero denominator.")

    except ValueError:

        print("Error: Invalid input. Please enter numeric values.")

    except Exception as e:

        print(f"An unexpected error occurred: {e}")

    else:

        print(f"The result of {num1} divided by {num2} is: {result}")

    finally:

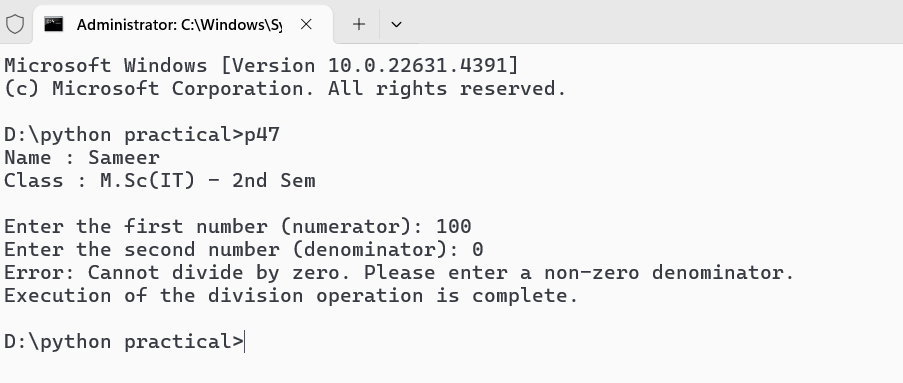
        print("Execution of the division operation is complete.")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    divide\_numbers()

**Output :-**

****

**PROGRAM – 48**

**Program : Exception Handling Program that handles multiple types of exceptions.**

def calculator():

    try:

        num1 = float(input("Enter the first number: "))

        num2 = float(input("Enter the second number: "))

        addition = num1 + num2

        subtraction = num1 - num2

        multiplication = num1 \* num2

        if num2 == 0:

            raise ZeroDivisionError("You cannot divide by zero.")

        division = num1 / num2

    except ZeroDivisionError as zde:

        print(f"Error: {zde}")

    except ValueError:

        print("Error: Invalid input. Please enter numeric values.")

    except Exception as e:

        print(f"An unexpected error occurred: {e}")

    else:

        print(f"Results:")

        print(f"Addition: {num1} + {num2} = {addition}")

        print(f"Subtraction: {num1} - {num2} = {subtraction}")

        print(f"Multiplication: {num1} \* {num2} = {multiplication}")

        print(f"Division: {num1} / {num2} = {division}")

    finally:

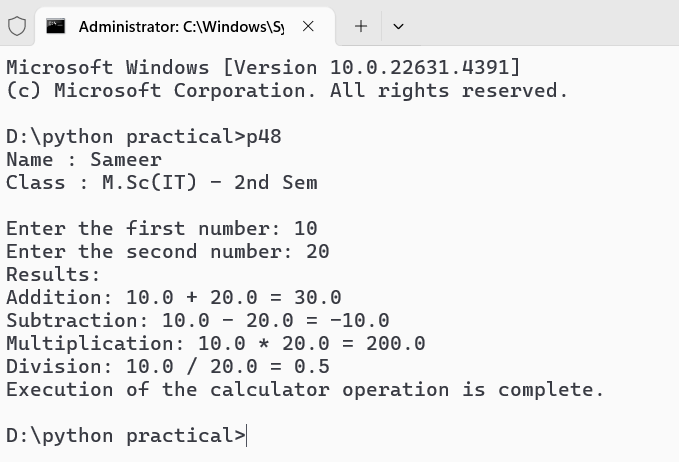
        print("Execution of the calculator operation is complete.")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    calculator()

**Output :-**

****

**PROGRAM – 49**

**Program : Exception Handling Program that uses try, except and else.**

import math

def calculate\_square\_root():

    try:

        number = float(input("Enter a number to find its square root: "))

        if number < 0:

            raise ValueError("Cannot calculate the square root of a negative number.")

        square\_root = math.sqrt(number)

    except ValueError as ve:

        print(f"Error: {ve}")

    except Exception as e:

        print(f"An unexpected error occurred: {e}")

    else:

        print(f"The square root of {number} is: {square\_root}")

    finally:

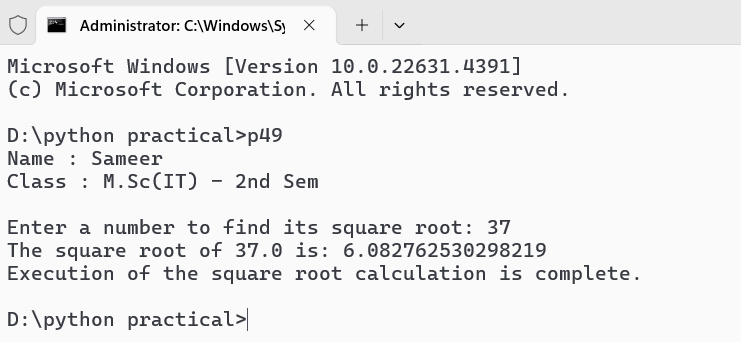
        print("Execution of the square root calculation is complete.")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    calculate\_square\_root()

**Output :-**



**PROGRAM – 50**

**Program : Exception Handling Program that uses finally with try.**

def read\_file(filename):

    try:

        with open(filename, 'r') as file:

            contents = file.read()

            print("File contents:")

            print(contents)

    except FileNotFoundError:

        print(f"Error: The file '{filename}' was not found.")

    except IOError:

        print(f"Error: An I/O error occurred while reading the file '{filename}'.")

    except Exception as e:

        print(f"An unexpected error occurred: {e}")

    finally:

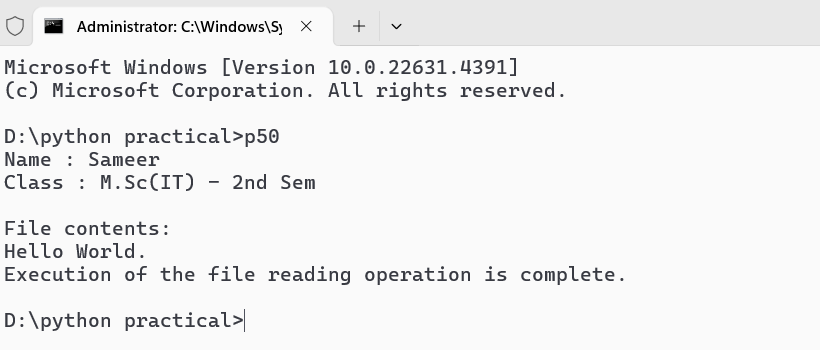
        print("Execution of the file reading operation is complete.")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    read\_file('example.txt')

**Output :-**

****

**PROGRAM – 51**

**Program : Write a Python program that creates a class “Person”, with attributes [aadhar, name, DoB]**

class Person:

    def \_\_init\_\_(self, aadhar, name, dob):

        self.aadhar = aadhar

        self.name = name

        self.dob = dob

    def display\_details(self):

        print(f"Aadhar Number: {self.aadhar}")

        print(f"Name: {self.name}")

        print(f"Date of Birth: {self.dob}")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    aadhar\_number = input("Enter Aadhar Number: ")

    name = input("Enter Name: ")

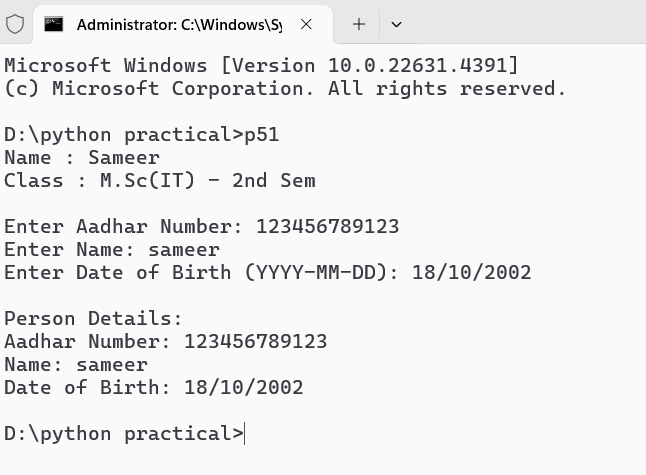
    dob = input("Enter Date of Birth (YYYY-MM-DD): ")

    person = Person(aadhar=aadhar\_number, name=name, dob=dob)

    print("\nPerson Details:")

    person.display\_details()

**Output :-**

****

**PROGRAM – 52**

**Program : Write a Python program that creates classes “Point” and “Rectangle” where the Rectangle class has a Point object as its attribute.**

class Point:

def \_\_init\_\_(self, x, y):

self.x, self.y = x, y

def \_\_str\_\_(self):

return f"Point({self.x}, {self.y})"

class Rectangle:

def \_\_init\_\_(self, bottom\_left: Point, width: float, height: float):

self.bottom\_left, self.width, self.height = bottom\_left, width, height

def area(self):

return self.width \* self.height

def perimeter(self):

return 2 \* (self.width + self.height)

def \_\_str\_\_(self):

return f"Rectangle(bottom\_left={self.bottom\_left}, width={self.width}, height={self.height})"

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

print("Name : Sameer\nClass : M.Sc(IT) – 3r Sem\n")

x, y = float(input("Enter x: ")), float(input("Enter y: "))

width, height = float(input("Enter width: ")), float(input("Enter height: "))

rectangle = Rectangle(Point(x, y), width, height)

print("\nRectangle Details:")

print(rectangle)

print(f"Area: {rectangle.area()}")

print(f"Perimeter: {rectangle.perimeter()}")

**Output :-**

**A screenshot of a computer program

Description automatically generated**

**PROGRAM – 53**

**Program : Write a Python program that creates a class Students which inherits the properties of the “Person” class; add attributes [roll\_no, class].**

class Person:

    def \_\_init\_\_(self, aadhar, name, dob):

        self.aadhar = aadhar

        self.name = name

        self.dob = dob

    def display\_details(self):

        print(f"Aadhar Number: {self.aadhar}")

        print(f"Name: {self.name}")

        print(f"Date of Birth: {self.dob}")

class Student(Person):

    def \_\_init\_\_(self, aadhar, name, dob, roll\_no, class\_name):

        super().\_\_init\_\_(aadhar, name, dob)

        self.roll\_no = roll\_no

        self.class\_name = class\_name

    def display\_details(self):

        super().display\_details()

        print(f"Roll Number: {self.roll\_no}")

        print(f"Class: {self.class\_name}")

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    aadhar\_number = input("Enter Aadhar Number: ")

    name = input("Enter Name: ")

    dob = input("Enter Date of Birth (YYYY-MM-DD): ")

    roll\_no = input("Enter Roll Number: ")

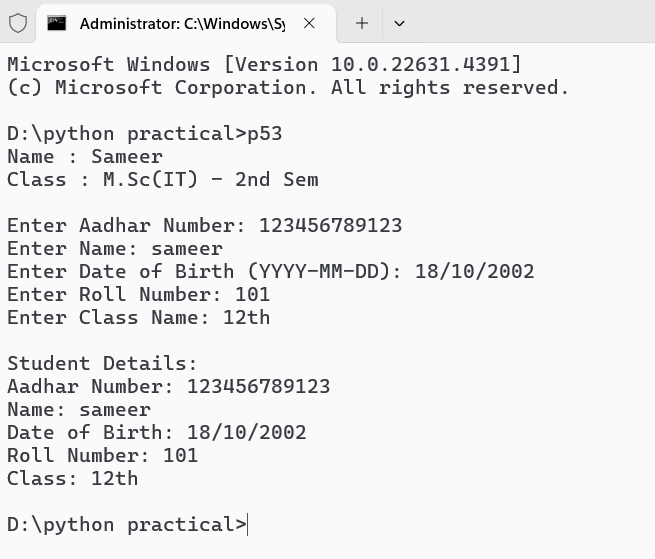
    class\_name = input("Enter Class Name: ")

    student = Student(aadhar=aadhar\_number, name=name, dob=dob, roll\_no=roll\_no, class\_name=class\_name)

    print("\nStudent Details:")

    student.display\_details()

**Output :-**



**PROGRAM – 54**

**Program : Write a Python program to demonstrate “Multiple Inheritance”.**

class Teacher:

    def \_\_init\_\_(self, name, employee\_id):

        self.name = name

        self.employee\_id = employee\_id

    def display\_teacher\_info(self):

        print(f"Teacher Name: {self.name}")

        print(f"Employee ID: {self.employee\_id}")

class Subject:

    def \_\_init\_\_(self, subject\_name, subject\_code):

        self.subject\_name = subject\_name

        self.subject\_code = subject\_code

    def display\_subject\_info(self):

        print(f"Subject Name: {self.subject\_name}")

        print(f"Subject Code: {self.subject\_code}")

class pythonTeacher(Teacher, Subject):

    def \_\_init\_\_(self, name, employee\_id, subject\_name, subject\_code):

        Teacher.\_\_init\_\_(self, name, employee\_id)

        Subject.\_\_init\_\_(self, subject\_name, subject\_code)

    def display\_info(self):

        self.display\_teacher\_info()

        self.display\_subject\_info()

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    name = input("Enter Teacher's Name: ")

    employee\_id = input("Enter Employee ID: ")

    subject\_name = "Python"

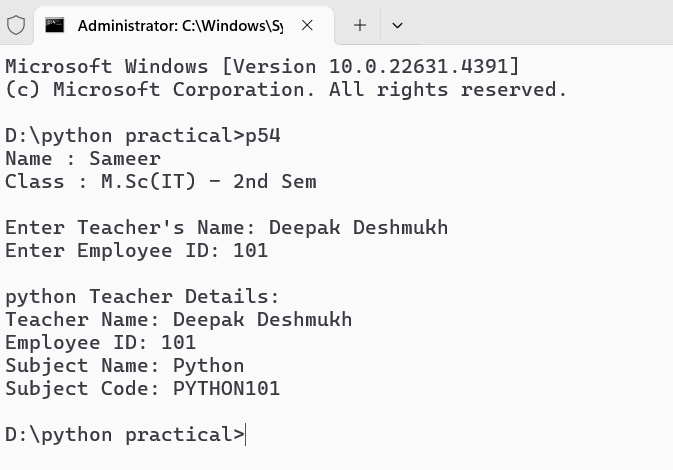
    subject\_code = "PYTHON101"

    python\_teacher = pythonTeacher(name, employee\_id, subject\_name, subject\_code)

    print("\npython Teacher Details:")

    python\_teacher.display\_info()

**Output :-**

****

**PROGRAM – 55**

**Program : Write a Python program to demonstrate “Method Overriding”.**

class base:

    def demo():

        print("I am demo function.")

class derived(base):

    def demo():

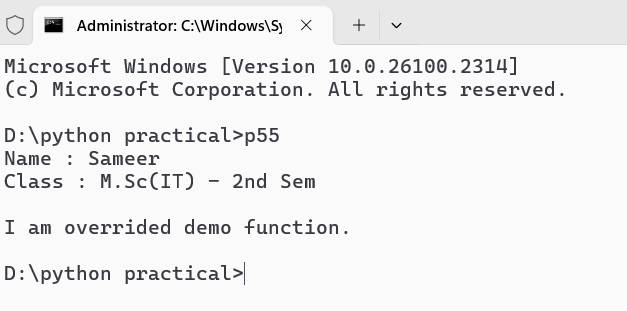
        print("I am overrided demo function.")

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

d = derived

d.demo()

**Output :-**

****

**PROGRAM – 56**

**Program : Write a Python program to demonstrate “Method Overloading”.**

class Calculator:

    def add(self, \*args):

        return sum(args)

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    calc = Calculator()

    result1 = calc.add(5, 10)

    print(f"Sum of 5 and 10 is: {result1}")

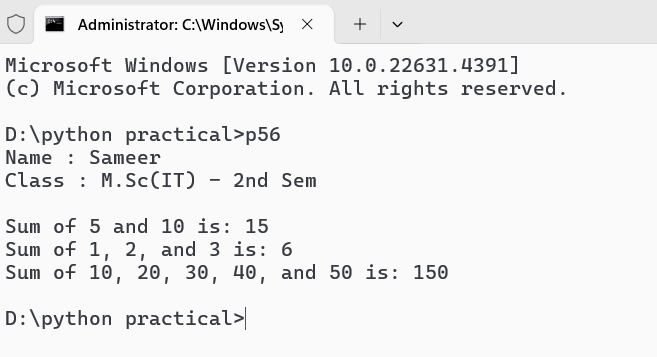
    result2 = calc.add(1, 2, 3)

    print(f"Sum of 1, 2, and 3 is: {result2}")

    result3 = calc.add(10, 20, 30, 40, 50)

    print(f"Sum of 10, 20, 30, 40, and 50 is: {result3}")

**Output :-**

****

**PROGRAM – 57**

**Program : Write a Python program to demonstrate “Operator Overloading” [+ and -] using a class “Book”.**

class Book:

    def \_\_init\_\_(self, title, pages):

        self.title = title

        self.pages = pages

    def \_\_add\_\_(self, other):

        if isinstance(other, Book):

            return Book(f"{self.title} & {other.title}", self.pages + other.pages)

        return NotImplemented

    def \_\_sub\_\_(self, other):

        if isinstance(other, Book):

            return abs(self.pages - other.pages)

        return NotImplemented

    def \_\_str\_\_(self):

        return f"'{self.title}' with {self.pages} pages"

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

    print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

    book1 = Book("Python Programming", 300)

    book2 = Book("Data Science", 250)

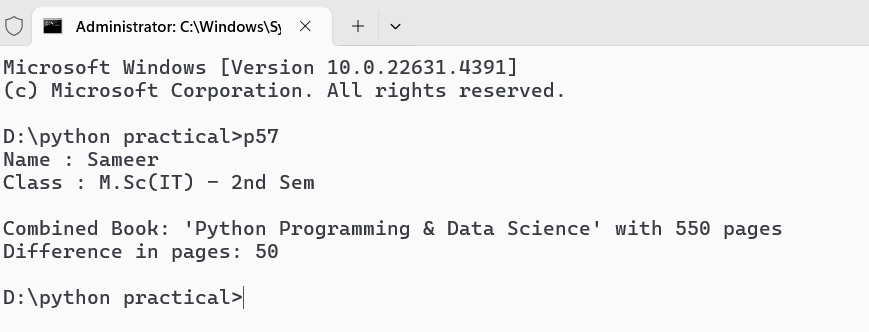
    combined\_book = book1 + book2

    print(f"Combined Book: {combined\_book}")

    page\_difference = book1 - book2

    print(f"Difference in pages: {page\_difference}")

**Output :-**

****

**PROGRAM – 58**

**Program : Use the “turtle” module to draw concentric circles with different colours.**

import turtle

import random

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

turtle.speed(0)

turtle.bgcolor("black")

colors = ["red", "green", "blue", "yellow", "purple", "orange", "cyan", "magenta", "white"]

num\_circles = 10

radius\_increment = 20

for i in range(num\_circles):

    turtle.color(random.choice(colors))

    turtle.penup()

    turtle.goto(0, -i \* radius\_increment)

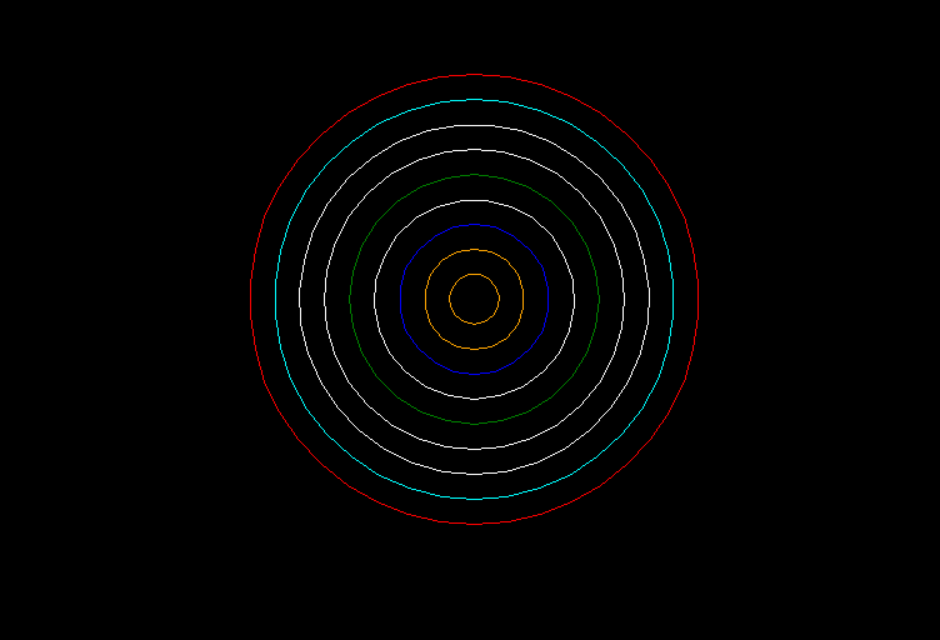
    turtle.pendown()

    turtle.circle(i \* radius\_increment)

turtle.hideturtle()

turtle.done()

**Output :-**

****

**PROGRAM – 59**

**Program : Use the “turtle” module to print the multiplication table.**

import turtle

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

turtle.speed(0)

turtle.penup()

turtle.goto(-200, 200)

turtle.pendown()

for i in range(1, 11):

    for j in range(1, 11):

        result = i \* j

        turtle.write(f"{result}", align="center", font=("Arial", 12, "normal"))

        turtle.penup()

        turtle.forward(50)

        turtle.pendown()

    turtle.penup()

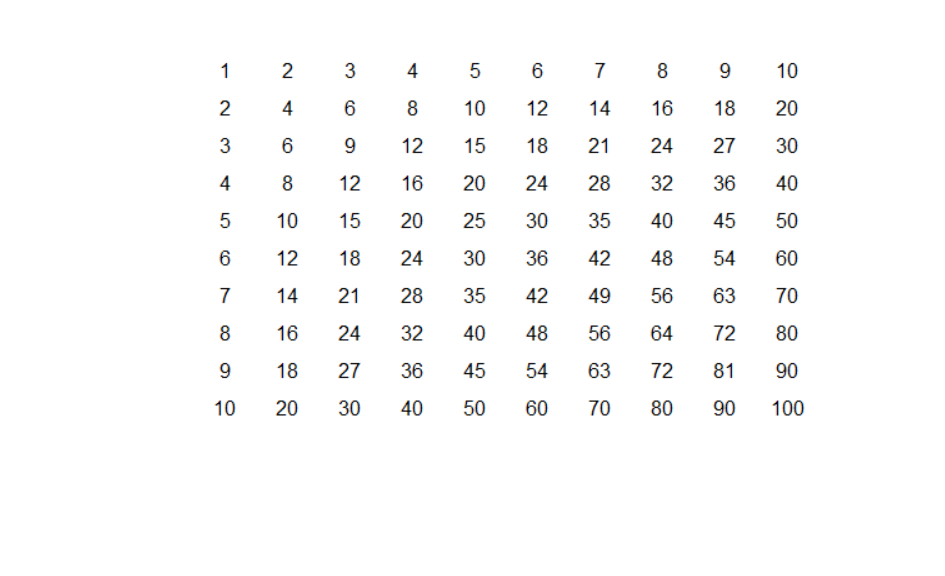
    turtle.goto(-200, turtle.ycor() - 30)

    turtle.pendown()

turtle.hideturtle()

turtle.done()

**Output :-**

****

**PROGRAM – 60**

**Program : Use the “turtle” module to draw (not write) your name.**

import turtle

print("Name : Sameer\nClass : M.Sc(IT) - 3rd Sem\n")

screen = turtle.Screen()

screen.bgcolor("white")

pen = turtle.Turtle()

pen.speed(0)

pen.pensize(10)

pen.bk(30)

pen.rt(90)

pen.fd(20)

pen.lt(90)

pen.fd(30)

pen.rt(90)

pen.fd(20)

pen.rt(90)

pen.fd(30)

pen.penup()

pen.bk(50)

pen.pendown()

pen.rt(90)

pen.fd(40)

pen.rt(90)

pen.fd(30)

pen.rt(90)

pen.fd(40)

pen.bk(20)

pen.rt(90)

pen.fd(30)

pen.penup()

pen.bk(50)

pen.pendown()

pen.lt(90)

pen.fd(20)

pen.bk(40)

pen.lt(45)

pen.fd(20)

pen.lt(90)

pen.fd(20)

pen.rt(135)

pen.fd(40)

pen.penup()

pen.lt(90)

pen.fd(20)

pen.pendown()

pen.lt(90)

pen.fd(40)

pen.rt(90)

pen.fd(30)

pen.bk(30)

pen.lt(90)

pen.bk(20)

pen.rt(90)

pen.fd(25)

pen.bk(25)

pen.lt(90)

pen.bk(20)

pen.rt(90)

pen.fd(30)

pen.penup()

pen.fd(20)

pen.pendown()

pen.lt(90)

pen.fd(40)

pen.rt(90)

pen.fd(30)

pen.bk(30)

pen.lt(90)

pen.bk(20)

pen.rt(90)

pen.fd(25)

pen.bk(25)

pen.lt(90)

pen.bk(20)

pen.rt(90)

pen.fd(30)

pen.penup()

pen.fd(20)

pen.pendown()

pen.lt(90)

pen.fd(40)

pen.rt(90)

pen.fd(25)

pen.rt(90)

pen.fd(20)

pen.rt(90)

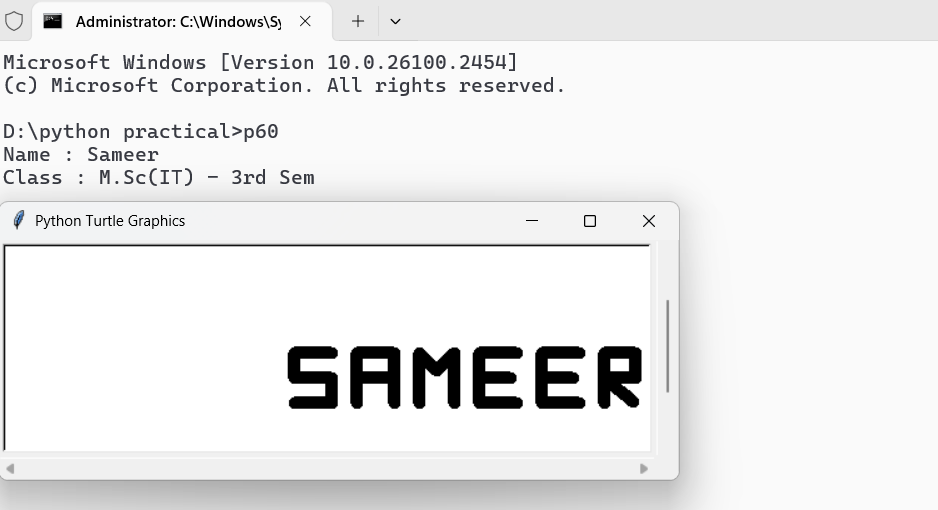
pen.fd(25)

pen.lt(140)

pen.fd(30)

turtle.done()

**Output :-**

****