#Q. 1 Write a program to enter name and display as “Hello, Name”.

name = input("Enter your name: ")

print("Hello,", name)

Output: -

#Q 2. Write a menu driven program to enter two numbers and print the arithmetic operations like

# a. +

#  b.

# c.   \*

#  d.    /    e.    //     f.     %.

while True:

    print("Menu driven program")

    print("A. Addition")

    print("B. Subtraction")

    print("C. Multiplication")

    print("D. Division")

    print("E. floor division(//)")

    print("F. Modulo(%)")

    print("G. Exit")

    choice = (input("Enter your choice: "))

    if choice == "A":

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

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

        print("Addition: ", x+y)

    elif choice == "B":

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

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

        print("Subtraction: ", x-y)

    elif choice == "C":

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

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

        print("Multiplication: ", x\*y)

    elif choice == "D":

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

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

        print("Division: ", x/y)

    elif choice == "E":

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

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

        print("floor division(//): ", x//y)

    elif choice == "F":

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

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

        print("Modulo(%): ", x%y)

    elif choice == "G":

        break

    else:

        print("Please enter valid choice")

Output: -

#Q 3. Write a program to compute the roots of a quadratic equation.

# ax^2 + bx + c = 0

# x = (-b ± √(b^2 - 4ac)) / 2a

import math

a = float(input("Enter the value of a: "))

b = float(input("Enter the value of b: "))

c = float(input("Enter the value of c: "))

d = b\*\*2 - 4\*a\*c

if d > 0:

    root1 = (-b + math.sqrt(d)) / (2 \* a)

    root2 = (-b - math.sqrt(d)) / (2 \* a)

    print("Roots are real and different")

    print("Root1: ", root1)

    print("Root2: ", root2)

else:

    if d == 0:

        root = -b / (2 \* a)

        print("Roots are real and same")

        print("Root: ", root)

    else:

        print("Roots are complex and different")

        real = -b / (2 \* a)

        imaginary = math.sqrt(abs(d)) / (2 \* a)

        print("Root1: ", real, "+", imaginary, "i")

        print("Root2: ", real, "-", imaginary, "i")

Output: -

#Q. 4 Write a menu driven Program to reverse the entered numbers and print the sum of digits entered.

while True:

    print("\nMenu:")

    print("A. Reverse the entered number")

    print("B. Sum of digits")

    print("C. Exit")

    choice = input("Enter your choice: ")

    if choice == 'A':

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

        rev = 0

        while num != 0:

            rem = num % 10

            rev = rev \* 10 + rem

            num = num // 10

        print("Reversed number: ", rev)

    elif choice == 'B':

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

        sum = 0

        while num != 0:

            rem = num % 10

            sum += rem

            num = num // 10

        print("Sum of digits: ", sum)

    elif choice == 'C':

        break

    else:

        print("Please enter valid choice")

Output: -

#Q 5. Write a menu driven Program to enter the number and print whether the number is

# a. odd or even

#  b. prime.

while True:

    print("\nMenu")

    print("A. Odd or Even")

    print("B. Prime")

    print("C. Exit")

    choice = input("Enter your choice: ")

    if choice == 'A' or choice == 'a':

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

        if num % 2 == 0:

            print(num, "is even")

        else:

            print(num, "is odd")

    elif choice == 'B' or choice == 'b':

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

        if num > 1:

            for i in range(2, num):

                if (num % i) == 0:

                    print(num, "is not a prime number")

                    break

            else:

                print(num, "is a prime number")

        else:

            print(num, "is not a prime number")

    elif choice == 'C' or choice == 'c':

        break

    else:

        print("Please enter valid choice")

Output: -

#Q 6. Program to find maximum out of entered 3 numbers.

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

num2 = int(input("Enter second number: "))

num3 = int(input("Enter third number: "))

if num1 > num2 and num1 > num3:

    print("Maximum number is", num1)

elif num2 > num1 and num2 > num3:

    print("Maximum number is", num2)

else:

    print("Maximum number is", num3)

Output: -

#Q 7. Write a program to display ASCII code of a character and vice versa.

while True:

    print("\n1. Display ASCII code of a character")

    print("2. Display character of a ASCII code")

    print("3. Exit")

    choice = input("Enter your choice: ")

    if choice == '1':

        char = input("Enter a character: ")

        print("ASCII code of", char, "is", ord(char))

    elif choice == '2':

        code = int(input("Enter a ASCII code: "))

        print("Character of ASCII code", code, "is", chr(code))

    elif choice == '3':

        break

    else:

        print("Please enter valid choice")

Output: -

#Q 8. Write a Program to check if the entered number is Armstrong or not.

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

sum = 0

temp = num

while temp > 0:

    digit = temp % 10

    sum += digit \*\* 3

    temp //= 10

if num == sum:

    print(num, "is an Armstrong number")

else:

    print(num, "is not an Armstrong number")

Output: -

#Q 9. Write a Program to find factorial of the entered number using recursion.

def factorial(n):

    if n == 0:

        return 1

    else:

        return n \* factorial(n-1)

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

print("Factorial of", n, "is", factorial(n))

Output: -

#Q 10. Write a Program to enter the number of terms and to print the Fibonacci Series.

n = int(input("Enter the number of terms: "))

n1 = 0

n2 = 1

count = 0

while count < n:

    print(n1, end=" ")

    nth = n1 + n2

    n1 = n2

    n2 = nth

    count += 1

    end = ""

#    print("Fibonacci Series is: ", end="")

Output: -

#Q 11. Write a Program to enter the numbers and to print greatest number using loop.

n = int(input("Enter the number of terms: "))

num = []

for i in range(n):

    num.append(int(input("Enter a number: ")))

print("Greatest number is", max(num))

Output: -

#Q 12. Write a Program to enter the string and to check if it’s palindrome or not (using loop).

string = input("Enter a string: ")

if string == string[::-1]:

    print("String is palindrome")

else:

    print("String is not palindrome")

Output: -

#Q 13. Write a Program to enter the 5 subjects numbers and print the grades A/B/C/D/E.

def calculate\_grade(marks):

    if marks >= 90:

        return 'A'

    elif marks >= 80:

        return 'B'

    elif marks >= 70:

        return 'C'

    elif marks >= 60:

        return 'D'

    else:

        return 'E'

def main():

    subjects = ['Subject 1', 'Subject 2', 'Subject 3', 'Subject 4', 'Subject 5']

    marks = []

    for subject in subjects:

        mark = int(input(f"Enter marks for {subject}: "))

        marks.append(mark)

    print("\nGrades for each subject:")

    for i, mark in enumerate(marks):

        grade = calculate\_grade(mark)

        print(f"{subjects[i]}: {grade}")

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

    main()

Output: -

#Q 14. Write a program in python language to display the given pattern:

        5

      4 5

    3 4 5

  2 3 4 5

1 2 3 4 5

n = int(input("Enter the number of rows: "))

for i in range(n, 0, -1):

    for space in range(i - 1):

        print("  ", end="")

    for j in range(i, n + 1):

        print(j, end=" ")

    print()

Output: -

#Q 15. Write a python function sin(x,n) to calculate the value of sin(x) using its Taylor series expansion up to n terms.

sin(x)=x−x^3/3!​+x^5/5!​−x^7/7!​+…

import math

def sin\_taylor(x, n):

    """

    Calculate sin(x) using the Taylor series expansion up to n terms.

    Args:

        x: The angle in radians.

        n: Number of terms in the Taylor series expansion.

    Returns:

        Approximated value of sin(x).

    """

    sin\_x = 0  # Initialize result

    for i in range(n):

        # Calculate the ith term: ((-1)^i \* x^(2i+1)) / (2i+1)!

        term = ((-1)\*\*i \* x\*\*(2\*i + 1)) / math.factorial(2\*i + 1)

        sin\_x += term  # Add the term to the result

    return sin\_x

# Example usage

angle = float(input("Enter the angle in radians: "))

terms = int(input("Enter the number of terms: "))

result = sin\_taylor(angle, terms)

print(f"sin({angle}) ≈ {result} (using {terms} terms)")

Output: -

#Q 16. Write a Program to determine EOQ using various inventory models.

Code: -

def model1(A,lamb,Ic):

    val = (2\*A\*lamb/Ic)\*\*(0.5)

    print(val)

def model2(A,lamb,Ic,xi):

    val = ((2\*A\*lamb/Ic)\*xi/(xi-lamb))\*\*(0.5)

    print(val)

def model3(A,lamb,Ic,pi):

    val = ((2\*A\*lamb/Ic)\*(pi + Ic)/pi)\*\*(0.5)

    print(val)

def model4(A,lamb,Ic,pi,xi):

    val = ((2\*A\*lamb/Ic)\*(xi/(xi-lamb))\*((pi + Ic)/pi))\*\*(0.5)

    print(val)

A = float(input("Ordering cost per order : "))

lamb = float(input("Demand rate per year : "))

Ic = float(input("Inventory Cost per unit per year : "))

xi = float(input("Production Rate : "))

pi = float(input("Shortage cost per unit : "))

model = int(input("Choose Model : "))

print("The optimal Order Quantity is : ", end = "")

if model == 1:

    model1(A,lamb,Ic)

if model == 2:

    model2(A,lamb,Ic,xi)

if model == 3:

    model3(A,lamb,Ic,pi)

if model == 4:

    model4(A,lamb,Ic,pi,xi)

if model == 5:

    pass

Output: -

#Q 17. Write a Program to determine different characteristics using various Queuing models

Code: -

def mm1(l,m,c=1):

    slen = int(l/(m-l))

    qlen = int(l\*l/(m\*(m-l)))

    qlen\_dash = int(m/(m-l))

    swait = 1/(m-l)

    qwait = l/(m\*(m-l))

    print('Expected system size is : ',slen)

    print('Expected queue size is : ',qlen)

    print('Expected queue size of non-empty queues is : ',qlen\_dash)

    print('Expected waiting time is : ',swait)

    print('Expected waiting time in queue is', qwait)

l = float(input('Enter arrival rate : '))

m = float(input('Enter service rate : '))

mm1(l,m)

Output: -

#Q 18.  Write a Program to implement Inheritance. Create a class Employee inherit two classes Manager and Clerk from Employee.

Code: -

class Employee:

    def \_\_init\_\_(self, name, age, position):

        self.name = name

        self.age = age

        self.position = position

    def show(self):

        print("Name    : ",self.name)

        print("Age     : ",self.age)

        print("Position: ",self.position)

        print()

class Manager(Employee):

    def \_\_init\_\_(self, name, age):

        super().\_\_init\_\_(name, age, "Manager")

class Clerk(Employee):

    def \_\_init\_\_(self, name,age):

        super().\_\_init\_\_(name, age, "Clerk")

emp1 = Clerk("Ritoban Sen", 22)

emp1.show()

emp2 = Manager("Ashish Shakya", 35)

emp2.show()

Output: -

#Q 19.  Program to fit Poisson distribution on a given data.

#     Input:  data - a list of integers

#     Output: lambda - the parameter of the Poisson distribution

#             data - the original data

#             poisson\_data - the Poisson distribution fitted on the data

Code: -

from scipy.stats import poisson

import numpy as np

from scipy.optimize import curve\_fit

def poisson\_func(x, lambda\_):

    return poisson.pmf(x, lambda\_)

def fit\_poisson(data):

    # Fit the Poisson distribution to the data

    counts, bins = np.histogram(data, bins=range(max(data) + 2))

    bins = bins[:-1]

    popt, \_ = curve\_fit(poisson\_func, bins, counts)

    lambda\_ = popt[0]

    # Generate the Poisson distribution

    poisson\_data = poisson.pmf(bins, lambda\_)

    return lambda\_, data, poisson\_data

# Example usage

data = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

lambda\_, data, poisson\_data = fit\_poisson(data)

print(f"lambda: {lambda\_}")

print(f"Data: {data}")

print(f"Poisson data: {poisson\_data}")

Output: -

#Q 20. Write a program to implement linear regression using python.

Code: -

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

# Generate some data

np.random.seed(0)

X = 2 \* np.random.rand(100, 1)

y = 3 + 4 \* X + np.random.randn(100, 1)

# Split the data into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2)

# Fit the model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Make predictions

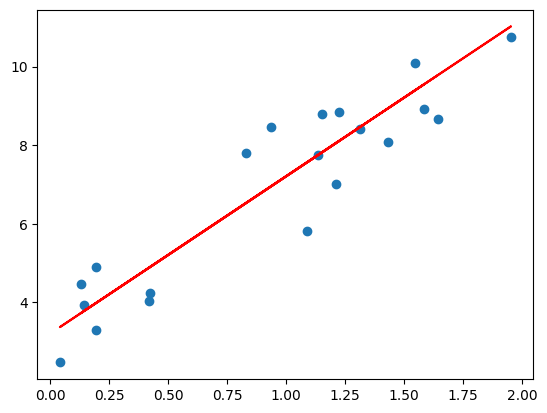
y\_pred = model.predict(X\_test)

# Plot the data

plt.scatter(X\_test, y\_test)

plt.plot(X\_test, y\_pred, color='red')

plt.show()



#Q 21. Write a program to perform read and write operation with .csv file.

#Answer:

import csv

import os

# Create a csv file

with open('data.csv', 'w', newline='') as csvfile:

    fieldnames = ['Name', 'Age', 'City']

    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

    writer.writeheader()

    writer.writerow({'Name': 'Alice', 'Age': 25, 'City': 'New York'})

    # Read the csv file

    with open('data.csv', 'r') as csvfile:

        reader = csv.DictReader(csvfile)

        for row in reader:

            print(row['Name'], row['Age'], row['City'])

# Delete the csv file

os.remove('data.csv')

#Q 22. Write a Program to enter multiple values-based data in multiple columns/rows and show that data in python using DataFrames and pandas.

#Answer:

import pandas as pd

import numpy as np

# Create a dictionary with multiple values-based data in multiple columns/rows

data = {

    "Name": ["John", "Anna", "Peter", "Linda"],

    "Age": [28, 23, 25, 27],

    "City": ["New York", "Paris", "Berlin", "London"]

}

# Create a DataFrame from the dictionary

df = pd.DataFrame(data)

print(df)

Output: -

#Q 23. WAP in python to perform various statistical measures using pandas library.

# A. Mean

# B. Median

# C. Mode

# D. Standard deviation

# E. Variance

# F. Correlation

# G. Skewness

# H. Kurtosis

# I. Covariance

# J. Describe

import pandas as pd

def calculate\_statistics():

    """

    Perform various statistical measures using pandas.

    """

    # Input data

    n\_rows = int(input("Enter the number of data points: "))

    column\_name = input("Enter the column name for the data: ")

    # Collect data

    data = []

    print("Enter the data values:")

    for \_ in range(n\_rows):

        data.append(float(input()))

    # Create DataFrame

    df = pd.DataFrame(data, columns=[column\_name])

    # Calculate statistics

    print("\n=== Statistical Measures ===")

    print(f"Mean: {df[column\_name].mean():.2f}")

    print(f"Median: {df[column\_name].median():.2f}")

    print(f"Mode: {df[column\_name].mode().values}")

    print(f"Standard Deviation: {df[column\_name].std():.2f}")

    print(f"Variance: {df[column\_name].var():.2f}")

    print(f"Minimum: {df[column\_name].min():.2f}")

    print(f"Maximum: {df[column\_name].max():.2f}")

    print(f"25th Percentile (Q1): {df[column\_name].quantile(0.25):.2f}")

    print(f"50th Percentile (Median): {df[column\_name].quantile(0.5):.2f}")

    print(f"75th Percentile (Q3): {df[column\_name].quantile(0.75):.2f}")

    print(f"Correlation: {df[column\_name].corr(df[column\_name]):.2f}")

    print(f"Skewness: {df[column\_name].skew():.2f}")

    print(f"Kurtosis: {df[column\_name].kurtosis():.2f}")

    print(f"Covariance: {df[column\_name].cov(df[column\_name]):.2f}")

    print("\n=== Description ===")

    print(df[column\_name].describe())

    calculate\_statistics()

#Q 1. Write a program to plot a bar chart in python to display the result of a school for five consecutive years.

#Answer:

import matplotlib.pyplot as plt

# Data

years = ['2016', '2017', '2018', '2019', '2020']

marks = [90, 85, 88, 92, 89]

# Plot

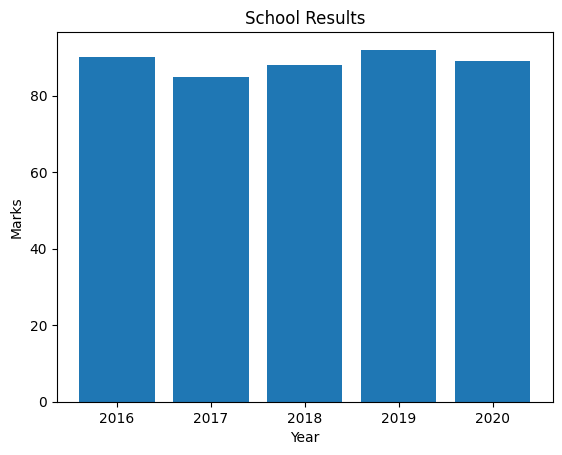
plt.bar(years, marks)

plt.xlabel('Year')

plt.ylabel('Marks')

plt.title('School Results')

plt.show()



#Q 2. Write a program in python to plot a graph for the function y = x^2.

#Ans:

import matplotlib.pyplot as plt

import numpy as np

# Define the function

def f(x):

    return x\*\*2

# Generate x values from -10 to 10

x = np.linspace(-10, 10, 100)

# Generate y values

y = f(x)

# Create the plot

plt.plot(x, y)

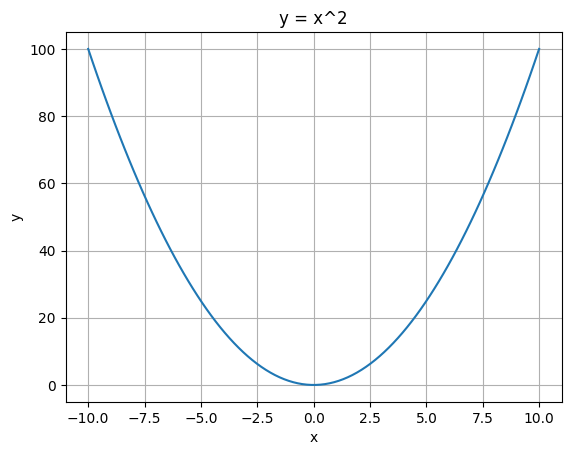
plt.xlabel("x")

plt.ylabel("y")

plt.title("y = x^2")

plt.grid(True)

plt.show()



#Q 3. Write a program in python to plot a pie chart on consumption of water in daily life.

#Ans:

import matplotlib.pyplot as plt

# Data to plot

labels = ['Bathing', 'Cooking', 'Drinking', 'Cleaning']

sizes = [20, 10, 40, 30]

colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue']

# Plot

plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%')

plt.axis('equal')

plt.show()

