**Python Programs**

**Program 1:** **distance\_converter**

# Write a program to convert distances in kilometers to miles. Use a `for` loop to iterate over a sequence of distances in kilometers (using `range(1, 11)`) and calculate their equivalent in miles (using the conversion factor: 1 kilometer is approximately 0.621371 miles). Display the result as a conversion table.

def km\_to\_miles(km):

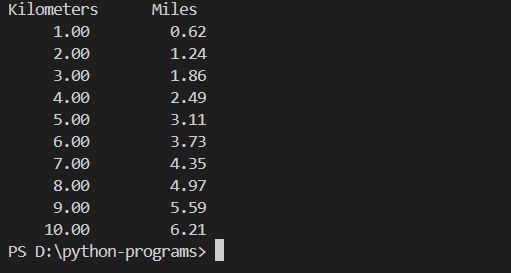
    return km \* 0.621371

print("Kilometers\tMiles")

for km in range(1, 11):

    miles = km\_to\_miles(km)

    print("{:9.2f}\t{:6.2f}".format(km, miles))

****

**Program 2:** **Grading-System-with-Class-Curve**

# Develop a Python script to calculate and display the final grades of students. Accept an unknown number of grades until the user decides to stop. For each student, the script should use a `while` loop to ensure entered grades are valid (between 0 and 100). After all grades are entered, find the highest grade and add five points to each student’s grade (but not exceeding 100). Display the adjusted grades.

def get\_valid\_grade(prompt):

    while True:

        try:

            grade = float(input(prompt))

            if grade == -1:

                return grade

            elif 0 <= grade <= 100:

                return grade

            else:

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

        except ValueError:

            print("Please enter a valid numerical grade.")

def calculate\_adjusted\_grades():

    grades = []

    while True:

        grade = get\_valid\_grade("Enter student's grade (or -1 to stop): ")

        if grade == -1:

            break

        grades.append(grade)

    if not grades:

        print("No grades entered.")

        return

    highest\_grade = max(grades)

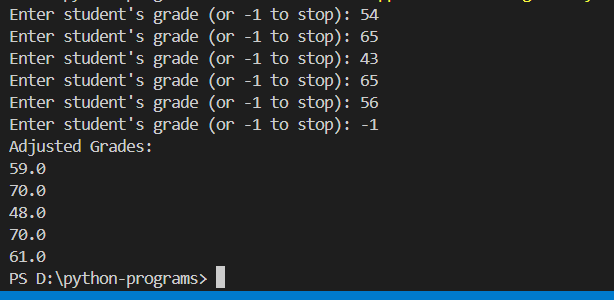
    adjusted\_grades = [min(grade + 5, 100) for grade in grades]

    print("Adjusted Grades:")

    for grade in adjusted\_grades:

        print(grade)

calculate\_adjusted\_grades()

****

**Program 3:** **monthly-expense-tracker**

# 50Write a script where a user can input their expected monthly income. Then, using a loop, allow them to enter various expenses until they choose to stop. Afterward, calculate and display the total expenses and the remaining amount from the income. Display a warning if expenses exceed income.

def calculate\_expenses(income):

    total\_expenses = 0

    while True:

        expense = float(input("Enter expense (or 0 to stop): "))

        if expense == 0:

            break

        total\_expenses += expense

    remaining\_income = income - total\_expenses

    print("Total expenses:", total\_expenses)

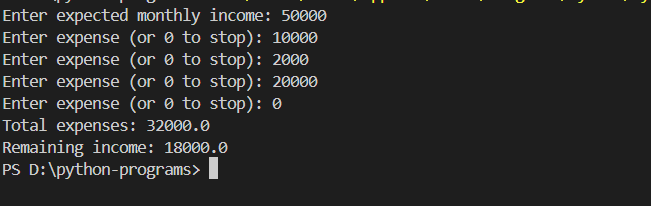
    print("Remaining income:", remaining\_income)

    if remaining\_income < 0:

        print("WARNING: Expenses exceed income!")

income = float(input("Enter expected monthly income: "))

calculate\_expenses(income)

****

**Program 4:** **Temperature-Checker-for-Merc1237**

'''A project in Pggc. requires Merc1237 to be continually heated in a vat. You have been asked to check the temperature every 20 minutes. If the temperature does not exceed 107.5 degrees Celsius, then you do nothing. However, if the temperature is greater than 107.5 degrees Celsius, then you

must turn down the vat’s thermostat, wait 10 minutes, and check the temperature again. You repeat these steps until the temperature does not exceed 107.5 degrees Celsius. You have been asked to write a program to guides you through this process.

Here is the algorithm:

1. Get the Merc1237 temperature.

2. Repeat the following steps as long as the temperature is greater than 107.5 degrees

Celsius:

a. Tell the technician to turn down the thermostat, wait 10 minutes, and check the

temperature again.

b. Get the temperature.

3. After the loop finishes, let the program state the temperature is acceptable and to

check it again in 20 minutes.'''

import time

def get\_temperature():

    while True:

        try:

            temperature = float(input("Enter the temperature (in degrees Celsius): "))

            return temperature

        except ValueError:

            print("Please enter a valid numerical temperature.")

def check\_temperature():

    while True:

        temperature = get\_temperature()

        if temperature <= 107.5:

            print("Temperature is acceptable.")

            break

        else:

            print("Temperature is too high. Turn down the thermostat and wait 10 minutes.")

            time.sleep(600)  # Wait for 10 minutes (10 minutes = 600 seconds)

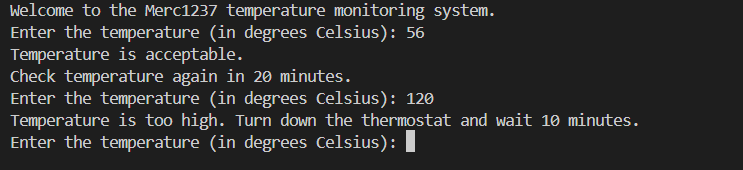
print("Welcome to the Merc1237 temperature monitoring system.")

while True:

    check\_temperature()

    print("Check temperature again in 20 minutes.")

    time.sleep(1200)  # Wait for 20 minutes (20 minutes = 1200 seconds)

****

**Program 5 :** **Volume-Comparison-of-Cylinders**

# Write a program that asks the user to input the radius and height of two cylinders. Calculate and compare their volumes (use the formula `Volume = π \* radius^2 \* height`), and inform the user which cylinder has a larger volume, or if they are equal

import math

def calculate\_volume(radius, height):

    return math.pi \* radius\*\*2 \* height

def compare\_volumes(radius1, height1, radius2, height2):

    volume1 = calculate\_volume(radius1, height1)

    volume2 = calculate\_volume(radius2, height2)

    if volume1 > volume2:

        print("Cylinder 1 has a larger volume.")

    elif volume1 < volume2:

        print("Cylinder 2 has a larger volume.")

    else:

        print("Both cylinders have equal volume.")

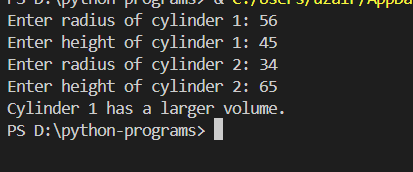
radius1 = float(input("Enter radius of cylinder 1: "))

height1 = float(input("Enter height of cylinder 1: "))

radius2 = float(input("Enter radius of cylinder 2: "))

height2 = float(input("Enter height of cylinder 2: "))

compare\_volumes(radius1, height1, radius2, height2)

****

**Program 6 :** **Weighted\_Grade\_Calculator**

# Write a script that asks the user to enter their scores for three different assignment categories: Homework, Quizzes, and Exams. Each category should have three scores. The Homework category is worth 25% of the final grade, Quizzes are worth 30%, and Exams are worth 45%. Calculate and display the final grade based on the entered scores. Use `while` loops to ensure users enter valid scores between 0 and 100.

def get\_valid\_score(prompt):

    while True:

        score = float(input(prompt))

        if 0 <= score <= 100:

            return score

        else:

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

def calculate\_final\_grade():

    homework\_scores = [get\_valid\_score("Enter homework score {}: ".format(i+1)) for i in range(3)]

    quizzes\_scores = [get\_valid\_score("Enter quiz score {}: ".format(i+1)) for i in range(3)]

    exams\_scores = [get\_valid\_score("Enter exam score {}: ".format(i+1)) for i in range(3)]

    homework\_weight = 0.25

    quizzes\_weight = 0.30

    exams\_weight = 0.45

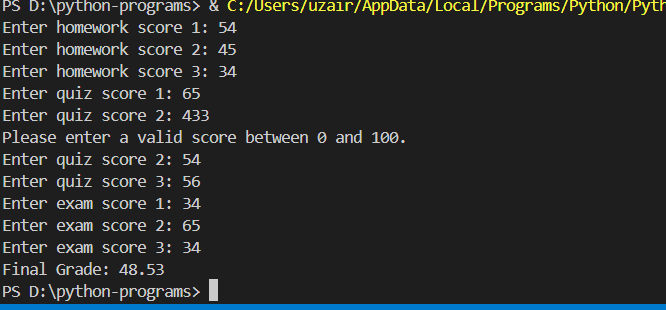
    final\_grade = (sum(homework\_scores) / len(homework\_scores)) \* homework\_weight + \

                  (sum(quizzes\_scores) / len(quizzes\_scores)) \* quizzes\_weight + \

                  (sum(exams\_scores) / len(exams\_scores)) \* exams\_weight

    print("Final Grade: {:.2f}".format(final\_grade))

calculate\_final\_grade()

****

**LAB 5**

**Program 7:Magic Date**

# Consider the date June 10, 1960. It's intriguing because when represented as 6/10/60, multiplying the month (6) by the day (10) gives the year (60). Your task is to develop a program that prompts the user to input a month (numerically), a day, and a two-digit year. The program should then ascertain if the product of the month and day matches the year. If there's a match, the program should display "The date is magic." If not, it should indicate "The date is not magic. Have two function, main function and is\_magic\_date that will accept the month, day, year. For example is\_magic\_date(month, day, year). Have the main function get the month, day and year and pass it to the is\_magic\_date function

def is\_magic\_date(month, day, year):

    if month \* day == year:

        return True

    else:

        return False

def main():

    month = int(input("Enter the month (numerically): "))

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

    year = int(input("Enter the two-digit year: "))

    if is\_magic\_date(month, day, year):

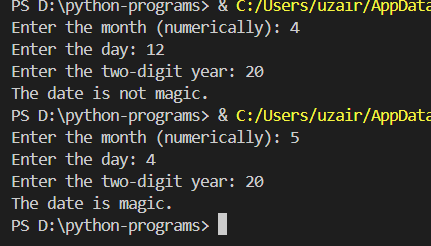
        print("The date is magic.")

    else:

        print("The date is not magic.")

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

    main()

****

**Program 8:Second.py**

'''Write a program that asks the user to enter a number of seconds, and works as follows:

There are 60 seconds in a minute. If the number of seconds entered by the user is greater than or equal to 60, the program should display the number of minutes in that many seconds.

There are 3,600 seconds in an hour. If the number of seconds entered by the user is greater than or equal to 3,600, the program should display the number of hours in that many seconds.

There are 86,400 seconds in a day. If the number of seconds entered by the user is greater than or equal to 86,400, the program should display the number of days in that many seconds'''

def convert\_seconds(seconds):

    days = seconds // 86400

    seconds %= 86400

    hours = seconds // 3600

    seconds %= 3600

    minutes = seconds // 60

    seconds %= 60

    return days, hours, minutes, seconds

def main():

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

    days, hours, minutes, remaining\_seconds = convert\_seconds(seconds)

    if days > 0:

        print("Days:", days)

    if hours > 0:

        print("Hours:", hours)

    if minutes > 0:

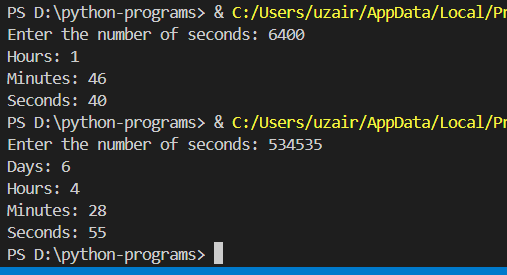
        print("Minutes:", minutes)

    if remaining\_seconds > 0:

        print("Seconds:", remaining\_seconds)

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

    main()

****

**Program 9 : Shipping-cost**

'''Fast Freight Shipping Co. has set its pricing based on the weight of packages as follows:

Up to 2 pounds: $1.10 per pound

More than 2 pounds, up to 6 pounds: $2.20 per pound

More than 6 pounds, up to 10 pounds: $3.70 per pound

Above 10 pounds: $3.80 per pound

Please design a program that prompts the user to input the weight of their package. The program should then calculate and display the total shipping cost. The check has to be done in a function and the main function passes the variable to the other function.'''

def calculate\_shipping\_cost(weight):

    if weight <= 2:

        return weight \* 1.10

    elif weight <= 6:

        return weight \* 2.20

    elif weight <= 10:

        return weight \* 3.70

    else:

        return weight \* 3.80

def main():

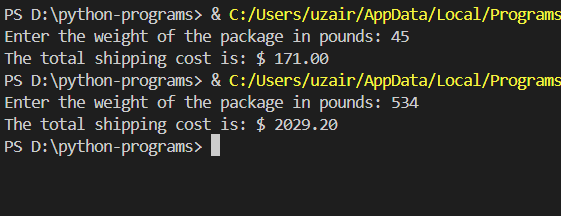
    weight = float(input("Enter the weight of the package in pounds: "))

    total\_cost = calculate\_shipping\_cost(weight)

    print("The total shipping cost is: $", format(total\_cost, ".2f"))

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

    main()

****

**Program 10 :** **budget-tracker**

'''Write a program that asks the user to enter the amount that he or she has budgeted for a

month. A loop should then prompt the user to enter each of his or her expenses for the

month, and keep a running total. When the loop finishes, the program should display the

amount that the user is over or under budget'''

# Ask the user to input the budget for the month

budget = float(input("Enter your budget for the month: "))

# Initialize a variable to keep track of total expenses

total\_expenses = 0

# Loop to prompt the user to enter each expense

while True:

    expense = input("Enter your expense for the month (or 'done' to finish): ")

    if expense.lower() == 'done':

        break

    else:

        # Convert the expense to a float and add it to total expenses

        total\_expenses += float(expense)

# Calculate the difference between budget and total expenses

difference = total\_expenses - budget

# Display the result

if difference > 0:

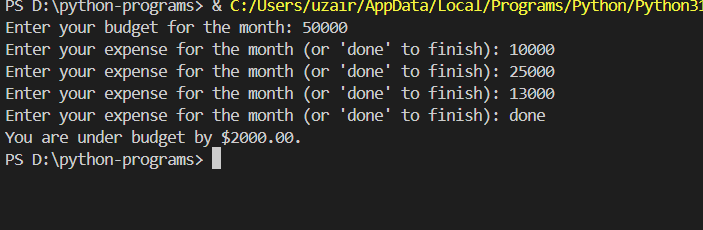
    print(f"You are over budget by ${difference:.2f}.")

elif difference < 0:

    print(f"You are under budget by ${-difference:.2f}.")

else:

    print("You have spent exactly your budget for the month.")

****

**Program 11: Fahrenheit to Celsius**

'''Your friend Chris wants you to convert Fahrenheit (F) to celsius (C). She has asked you to write a program that displays a table of temperature in C with their values converted to F.  The formula for converting C to F is F =((9/5)\*C + 32).   After thinking about this table of values, you decide that you will write a for loop. The list of values that the loop will iterate over will be Celsius(C). In the loop   you will call the range function like this: range(10, 51, 5) . Notice that the third argument specifies 5 as the step value. This means that the numbers in the list will be 10, 15, 20, and so forth. The second

argument specifies 51 as the sequence’s ending limit, so the last number in the sequence will be 50. Inside the loop you will use the target variable to calculate a Fahrenheit'''

# Loop over a range of temperatures in Celsius

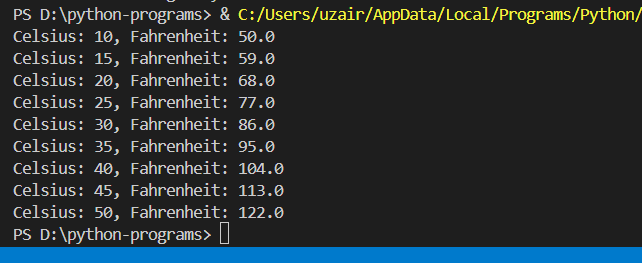
for celsius in range(10, 51, 5):

    # Convert Celsius to Fahrenheit

    fahrenheit = (9/5) \* celsius + 32

    # Display the converted temperature

    print(f"Celsius: {celsius}, Fahrenheit: {fahrenheit}")

****

**Program 12 :** **test-score**

'''Write a code to accept 3 test score and display them for a student. Use a while loop for this code.

Sample output:

Enter test score: 90

Test score you entered is : 90

Enter test score: 80

Test score you entered is 80

Enter test score: 85

Test score you entered is 85'''

# Initialize a counter to keep track of the number of test scores entered

count = 0

# Loop until 3 test scores are entered

while count < 3:

    # Ask the user to input a test score

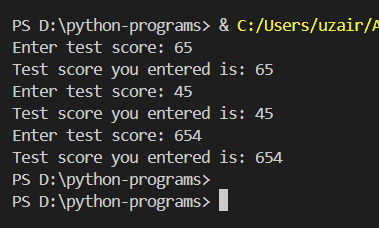
    score = input("Enter test score: ")

    # Display the entered test score

    print("Test score you entered is:", score)

    # Increment the counter

    count += 1

****

**Program 13:** **Adding Numbers**

def additionNum():

    total = 0

    for i in range(num):

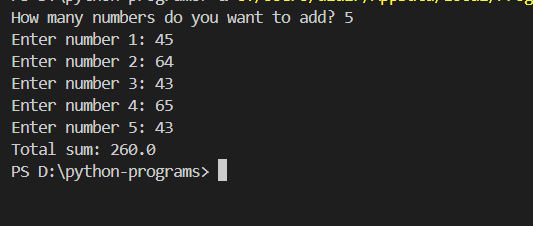
        num\_input = float(input(f"Enter number {i+1}: "))

        total += num\_input

    print("Total sum:", total)

num = int(input("How many numbers do you want to add? "))

additionNum()

****

**Program 14 :** **Profit Calculation for iPhone Sales**

def profitcalc():

    while True:

        cost = float(input("Enter the item's wholesale cost: $"))

        if cost < 0:

            print("ERROR: The cost cannot be negative.")

        else:

            profit = cost \* 2.5

            print(f"Profit price: ${profit:.2f}.")

            return

def main():

    while True:

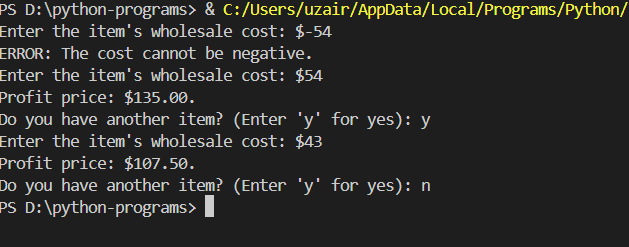
        profitcalc()

        choice = input("Do you have another item? (Enter 'y' for yes): ")

        if choice.lower() != 'y':

            break

main()

****

**Program 15 : Pattren Design**

rows = 9  # Number of rows in the pattern

# Outer loop for rows

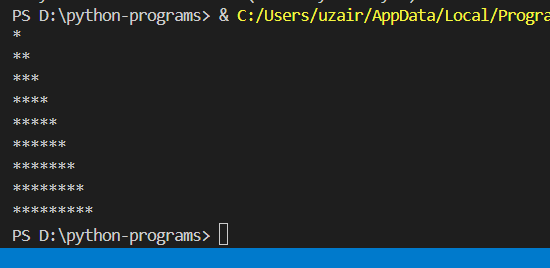
for i in range(1, rows + 1):

    # Inner loop for columns

    for j in range(1, i + 1):

        print('\*', end='')  # Print '\*' without a newline

    print()  # Move to the next line after printing each row

****

**Program 16 : Total of series**

total = 0

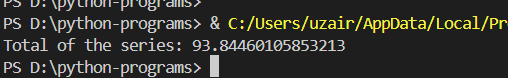
# Loop through the series from 1 to 30

for i in range(1, 31):

    # Add each term of the series to the total

    total += i / (31 - i)

print("Total of the series:", total)

****

**Program 17 : Coin-toss Simulator**

import random

def coin\_toss():

    for \_ in range(10):

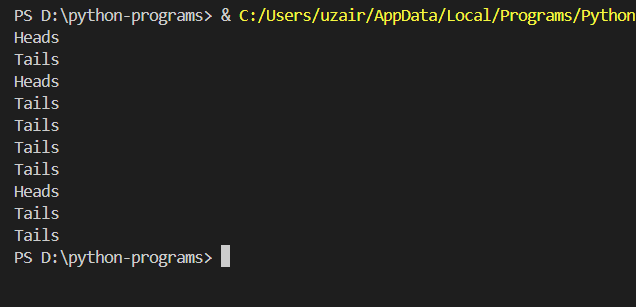
        if random.randint(1, 2) == 1:

            print("Heads")

        else:

            print("Tails")

coin\_toss()

****

**Program 18 : Dice Rolling Simulator**

import random

def roll\_dice():

    print("Rolling the dice...")

    print("Their values are:")

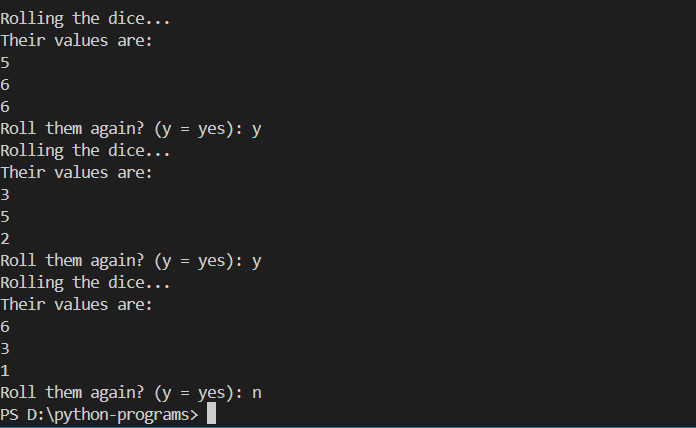
    for \_ in range(3):

        print(random.randint(1, 6))

    return input("Roll them again? (y = yes): ").lower() == 'y'

while roll\_dice():

    pass

****