PHY1112 Lab 3

Loops and functions

January 23rd, 2024

|  |  |  |  |
| --- | --- | --- | --- |
| Part | 1 | 2 | Total |
| Points | 10 | 10 | 20 |
| Score |  |  |  |

Objectives

1. Loop over a list of items and conditionally interact with the entries.
2. Refactor previous code into a function.

Part 1: Loops and conditionals – how’s the weather?

1. (10 points) Open the file “lab3.py” and fill out the header. There will be a list of temperatures there called temperatures that includes floats corresponding to the daily high temperatures in Ottawa over the month of September 2023 (in Celsius).

Continuing to work in lab3.py:

* 1. Use a for loop and if statements to count the number of days that were:
     1. hot (temperature equal to or above 30oC)
     2. warm (temperature equal to or above 20oC, and less than 30oC)
     3. temperate (temperature less than 20oC)

Report the values you obtained.

* 1. Use your code to determine the total number of days; report the value you obtained. Also, use your code to make sure the total number of categorized days is equal to the total number of temperatures given in temperatures.

Hot: 4

Warm: 21

Temperate: 5

A screen shot of a computer

Description automatically generated

* Note the function only indicates (returns False) if the total number of categorized days is not equal to the total number of temperatures given, otherwise returns the categorized list.

Part 2: Functions – refactoring the quadratic equation root finder

1. (10 points) Continuing to work in the lab3.py file, create a function quad\_root\_finder() that takes three inputs – the parameters for a quadratic equation – and returns the two roots of that quadratic equation.

This is the same problem as in Lab 2, but now you are writing your code as a function. Remember to have your function return both roots.

Do not ask the user for the parameters via the input()function as you did last time, but instead pass them directly as the inputs to the quad\_root\_finder() function in your .py file.

Use your function to find the roots for the following parameters:



(1.0, -1.0)

(1.0883036880224506, 0.24502964531088273)



((0.5-0.10910894511799618j), (0.5+0.10910894511799618j))



(0.8545686454136975, -1.521235312080364)



1.0

Code

'''

Filename:       lab3.py

Author:         Patrick Geraghty

Date Created:   2023-10-10

Date Modified:  2024-01-23

Description:    Contains a function that categorizes average daily temperatures into 'hot', 'warm', and 'temperate' from a given list. Contains an additional function that returns the roots of a quadratic equation given its 'a', 'b', and 'c' values as parameters.

'''

# daily high temperatures in Ottawa for the month of Septermber 2023

# data retrieved from https://climate.weather.gc.ca/climate\_data/daily\_data\_e.html?StationID=49568&timeframe=2&StartYear=1840&EndYear=2023&Day=9&Year=2023&Month=9#

temperatures = [24.1, 25.0, 29.5, 30.3, 32.5, 32.6, 30.1, 20.2, 22.7, 21.7,

        24.5, 19.8, 22.0, 16.4, 21.9, 22.9, 24.4, 21.6, 20.2, 18.6, 21.3, 22.8,

        21.9, 24.8, 21.2, 18.8, 19.2, 22.0, 21.9, 24.0] # Celsius

# Part 1

*def* temp\_stats():

        '''

        () -> list or bool

        Counts the number of days the temperature was above 30 C, between 20 C and 30 C, and less than 20 C, listing them as 'hot', 'warm', and 'temperate' respectively. Additionally, it checks if the summation of 'hot', 'warm', and 'temperate' days matches the number of days in 'temperatures'. If the summation does not match, the function returns False.

        Preconditions: predetermined list of temperatures 'temperatures'.

        '''

        # categorize temperatures into blank values (0)

        hot = 0

        warm = 0

        temperate = 0

        # iterate through list of temperatures and categorize them into respective categories

        for i in temperatures:

                if i >= 30:

                        hot += 1

                elif 30 > i >= 20:

                        warm += 1

                else:

                        temperate += 1

        # create a list of the categorized temperatures

        stats = [hot, warm, temperate]

        # check if the summation of the categorized temperatures matches the number of days in 'temperatures'

        count = 0

        for i in stats:

                count += i

        if count != len(temperatures):

                return False

        # return the list of categorized temperatures only if the count loop passes (does not return False)

        return stats

# Part 2

# function to find the discriminant of a quadratic equation (for simpler root finding)

*def* discriminant(*a*,*b*,*c*):

        '''

        (num,num,num) -> num or complex

        Takes three numbers as parameters 'a', 'b', and 'c', and returns the discriminant of the quadratic equation.

        '''

        return b\*\*2 - 4\*a\*c

# function to find the roots of a quadratic equation

*def* quad\_root\_finder(*a*,*b*,*c*):

        '''

        (num,num,num) - > tuple or num

        Takes three numbers as parameters 'a', 'b', and 'c', and returns the roots of the quadratic equation.

        Preconditions: 'a', 'b', and 'c' are all numbers.

        '''

        # check if the discriminant is positive, negative, or zero and return the roots characteristic of the discriminant

        if discriminant(a,b,c) < 0:

                return *complex*((-b)/(2\*a), + (-(-discriminant(a,b,c))\*\*0.5)/(2\*a)), *complex*((-b)/(2\*a), - (-(-discriminant(a,b,c))\*\*0.5)/(2\*a))

        elif discriminant(a,b,c) == 0:

                return (-b)/(2\*a)

        else:

                return (-b + (discriminant(a,b,c)\*\*0.5))/(2\*a), (-b - (discriminant(a,b,c)\*\*0.5))/(2\*a)