* **ASSIGNEMENT**

 Tasks

Now apply what you learned and complete the following tasks:

Task 1:

Let’s start with some basic geometry. The Python script triangle\_area.py calculates the area of a triangle. Comment the code, replacing each instance of [explain] with your own explanation as to what the block of code does, and save the modified file.

**SOLUTION ATTEMPTED**

# python program to find area of triangle

# Triangle has three sides

# If a, b, c, are the three sides

# a = (input ('Enter dimension of first side: ' ))

# b = (input ('Enter dimension of second side: ' ))

# c = (input ('Enter dimension of thrid side: ' ))

# to get area, first calculate semi-perimeter

# s = semiperimeter

a = height = 2

b = width = 2

c = length = 2

# calculate the semi-perimeter of the triangle

s = (a + b + c) / 2

perimeter = s

area = (s\*(s-a)\*(s-b)\*(s-c)) \*\* 0.5

area = (s\*(s-a)\*(s-b)\*(s-c)) \*\* 0.5

print('The area of the triangle is %0.2f' %area)

Task 2:

Modify the code from Task 1 so that it first asks the user to either calculate the area of a triangle or a trapezoid, prompts the user to enter the necessary values to perform the right calculations, then performs the calculations and outputs the result. Make sure your code is commented appropriately, and save it as **triortrap\_area.py**.

**SOLUTION ATTEMPTED**

# Python Program - Calculate Area of Triangle

# Read input values when "enter 'y'

while True:

print("Enter 'y' for exit.")

side1 = input("Enter dimension of first side: ")

side2 = input("Enter dimension of second side: ")

side3 = input("Enter dimension of third side: ")

if side1 == 'y':

break

else:

a = float(side1)

b = float(side2)

c = float(side3)

# calculate the semi-perimeter

s = (a + b + c)/2

# calculate the area

area = (s\*(s-a)\*(s-b)\*(s-c)) \*\* 0.5

print("Area of Triangle = %0.2f" %area)

# area of triangle value displayed on screen

Task 3:

In the previous program, imagine that you are a user in a hurry that types 2O (oh) rather than 20 (zero) as the height. What happens to your program? In “**try\_except\_example.py**”, you can see an example that shows how to avoid such problem.

Now let’s try to combine the example with the code from Task 2. If the user enters a value that could not be converted to numeric type, allow 3 additional opportunities to enter a new value. If the user fails to enter a correct value after 4 attempts, inform them of such failure and allow the program to end without crashing. Save your new code as **triortrap\_advanced.py**.

Task 4:

Write a Python script that asks the user to input the latitude and longitude of a location. After the user has entered the information, the script should display one of the following messages to describe the latitude entered:

| **Your program should display** | **If the latitude entered** |
| --- | --- |
| That location is on the equator. | Is 0 |
| That location is north of the equator | Is between 0 and 90 |
| That location is south of the equator | Is between -90 and 0 |
| That location does not have a valid latitude! | Is greater than 90 or less than -90 |

and one of the following messages to describe the longitude entered:

| **Your program should display** | **If the latitude entered** |
| --- | --- |
| That location is on the prime meridian. | Is 0 |
| That location is east of the prime meridian. | Is between 0 and 180 |
| That location is west of the prime meridian | Is between -180 and 0 |
| That location does not have a valid longitude! | Is greater than 180 or less than -180 |

If the user enters something other than a number, give them an error message and restart the program. Save your script as **latlon.py**.

**SOLUTION ATTEMPTED**

lat = int(input ("Enter the latitude:"))

if lat > 0 or lat < 90:

print("ERROR Please enter a valid latitude")

else :

if lat > 0 or lat < 90:

lat = int(input ("Enter the latitude:"))

print("That location is on the equator.")

lon = int(input ("Enter the longitude:"))

if lon > 180 or lon < 0 or lon > 0 or lon < 180:

print("ERROR Please enter a valid longitude");

lon = int(input ("Enter the longitude:"))

print("That location is east of the prime meridean.")

Task 5

Time for a GIS task. Write a Python script that calculates the distance between any two points on the Earth’s surface, given their latitude and longitude.

The script should first prompt the user to enter the coordinates of each location, then calculate the spherical distance along the Earth’s surface (use the following distance formula). If the user does not enter numbers for latitude or longitude values, prompt them to re-enter.

To make the problem simpler, you may assume that numbers entered are valid latitude or longitude coordinates in decimal degrees, but feel free to experiment with responding to violations of this assumption.

Finally, save your script as **great\_circle\_dist.py**.

Spherical distance formula:

screenshot of Spherical distance formula

whereby **d** is the angle between two points, λ1 and λ2 are the longitudes of the points, and Φ1 and Φ2 are the latitudes of the points. To convert the angle to a spherical distance, multiply by the radius of the earth, which is ~6300 km.

Hints

* To use trigonometric functions in Python, you will need to import the math module at the beginning of the script.
* Python’s math module assumes that angular values are given in radians. Latitude and longitude values are always given in *degrees*, so you will need to convert the values entered by the user from degrees to radians.
* Standard convention is to express northern latitudes and eastern longitudes as positive numbers, and southern latitudes and western longitudes with negative numbers.

Sample outputs

Here are some sample locations to test your code with:

Madison: 43.13972 ºN, 89.3375 ºW

New York: 40.77725 ºN, 73.87261 ºW

Jakarta: 6.12555 ºS, 106.6558 ºE

The output distances should be:

Madison -­ New York: 1289 km

Madison -­ Jakarta: 15437 km

New York -‐ Jakarta: 15982 km

**SOLUTIONS ATTEMPTED**

**# -\*- coding: utf-8 -\*-**

**# distance = 6300 × arccos(sin(t1) × sin(t2) + cos(t1) × cos(t2) × cos(g1 − g2))**

**#Python's program to calculate Distance Between Two Points on Earth**

**##**

**from math import radians, cos, sin, asin, sqrt**

**#Read the input from user**

**print("Enter the latitude and longitude of two points on the Earth in degrees:")**

**lat1 = float(input(" Latitude 1: "))**

**lat2 = float(input(" Latitude 2: "))**

**lon1 = float(input(" Longitude 1: "))**

**lon2 = float(input(" Longitude 2: "))**

**# try and convert the string input to a number**

**# tell the user off**

**# The math module contains a function named radians which converts from degrees to radians.**

**lon1 = radians(lon1)**

**lon2 = radians(lon2)**

**lat1 = radians(lat1)**

**lat2 = radians(lat2)**

**# Haversine formula**

**dlon = lon2 - lon1**

**dlat = lat2 - lat1**

**a = sin(dlat/2)\*\*2 + cos(lat1) \* cos(lat2) \* sin(dlon/2)\*\*2**

**c = 2 \* asin(sqrt(a))**

**r = 6300 # Radius of earth in kilometers.**

**#Display the result**

**print("Distance is: ",c\*r,"Kilometers")**