CS 37 C++ Programming

Instructor: Chris Macadam **Spring 2020**
Email: cmacadam@ivc.edu

Programming Assignment #1

Date Due:
02/20/20

Deliverables:
- A hard copy of your program (.cpp file only) with the output results appended at the end.
- A copy of the same uploaded to Canvas under the Assignment 1 section.

Requirements:
- Use the template provided on Canvas (Assignment_1_Template.txt).
- **Staple** your hard copy together if it is more than one page long.

Grading:
- 25 points possible
  - 3 pts - Comments
  - 4 pts - Consistency/Readability (Spacing / Indentation / Style)
  - 10 pts - Program works and generates results in the expected format
  - 8 pts - Use appropriate tools/structures, include required elements

- While you may collaborate, you are expected to turn in your own work.

Background
Scotty, a zoology student at IVC, has been tasked with finding out how far, as 'the crow flies', a number of national parks around the world are from IVC. To calculate these distances by hand would be too time consuming so he has asked you to help him write a program that will automate this task.

To find the distance between two points on the surface of the Earth two things are necessary:
1- A way to determine the position of each point on the surface of the Earth. *Latitude* and l*ongitude* have been used for many years now to describe such positions.
2- A formula to calculate the distance using latitude and longitude.

The program
Your program will read the following information from the 'npproject.txt' file provided (do not change the name of the file). The structure of the file is as follows:

- The first 5 lines are the starting location:
    - Name
    - State
    - Country
    - Latitude
    - Longitude
- The next line is the number of parks in the list.
- For each park, 5 lines of data:
    - Name
    - State (some of these are empty)
    - Country
    - Latitude
    - Longitude

Latitude values range from 0 to -90 (South of the Equator) and from 0 to 90 (North of the Equator).
Longitude values range from 0 to -180 (this is West) and from 0 to 180 (this is East) with -180 and 180 being equivalent.

The program should calculate the distance between the starting point and each of the parks to determine which park is closest to the starting point and which park is farthest. Information for the starting point, the closest park and the farthest park should be displayed on the screen.

The output
The output of the program should be in the following format.
- For the starting point
The first line will have a label, 'Starting Point', followed by a colon. Lines 2 through 4 will each be indented 5 spaces. Line 2 will have a label, 'Name', followed by a colon, a space, the name of the location, a space, the state and country separated by a comma and a space all enclosed in parentheses. Line 3 will have a label, 'Latitude', followed by a colon, a space, and the latitude. Line 4 will have a label, 'Longitude', followed by a colon, a space, and the longitude.

- For the closest park
The first line will have a label, 'Closest Park', followed by a colon. Lines 2 through 5 will each be indented 5 spaces. Line 2 will have a label, 'Name', followed by a colon, a space, the name of the location, a space, the state and country separated by a comma and a space all enclosed in parentheses. Line 3 will have a label, 'Latitude', followed by a colon, a space, and the latitude. Line 4 will have a label, 'Longitude', followed by a colon, a

space, and the longitude. Line 5 will have a label, 'Distance', followed by a colon, a space, the distance in kilometers from the starting point, and a space followed by 'kms' for units

- For the farthest park
The first line will have a label, 'Farthest Park', followed by a colon. Lines 2 through 5 will each be indented 5 spaces. Line 2 will have a label, 'Name', followed by a colon, a space, the name of the location, a space, the state and country separated by a comma and a space all enclosed in parentheses. Line 3 will have a label, 'Latitude', followed by a colon, a space, and the latitude. Line 4 will have a label, 'Longitude', followed by a colon, a space, and the longitude. Line 5 will have a label, 'Distance', followed by a colon, a space, the distance in kilometers from the starting point, and a space followed by 'kms' for units.

Values for latitude and longitude will have 6 digits of precision to the right of the decimal point.
Values for distance will have 2 digits of precision to the right of the decimal point.

Starting Point:
    Name: Irvine Valley College (CA, USA)
    Latitude: 33.675373
    Longitude: -117.777549
Closest Park:
    Name: Mojave National Preserve (CA, USA)
    Latitude: 35.141689
    Longitude: -115.510399
    Distance: 297.00 kms
Farthest Park:
    Name: Tierra del Fuego National Park (Tierra del Fuego, Argentina)
    Latitude: -54.799968
    Longitude: -68.299998
    Distance: 10964.02 kms

These are not actually the closest and farthest parks but these values are correct and can be useful in determining whether the formula you have written is calculating the distances properly. Depending upon rounding because of the use of float/double types, your results for the calculated distance may be off by +/- .01 and that's fine.

Formula to calculate distance
For our purposes, we'll consider that the Earth is a sphere (which it mostly is) so the distance between two points on the surface is an arc, whose length is known as the *great-circle distance*. Different formulas exist for calculating this distance but we will use this one:

temp = $\sin^2(latDiff/2) + \cos(lat1) * \cos(lat2) * \sin^2(longDiff/2)$
angRad = $2 * atan2(\sqrt{temp}, \sqrt{(1-temp)})$
distance = eRadius * angRad

where

lat1 is the latitude of the starting point
lat2 is the latitude of the park
long1 is the longitude of the starting point
long2 is the longitude of the park
latDiff is the difference in latitudes between locations
longDiff is the difference in longitudes between locations
eRadius is the average radius of the Earth - use 6371 kilometers

Trigonometric functions
The trigonometric functions, **sin**, **cos**, and **atan2** are available when you include the <**cmath**> library. It is important to note that the arguments to these functions are expected to be in **radians** while the coordinates (latitude and longitude) are given in degrees. You will need to convert from degrees into radians given that there are 360 degrees in a circle and equivalently there are 2π radians in a circle.

$360 = 2\pi$

Other functions you will also need from the <**cmath**> library are **pow** (to raise an expression to a power) and **sqrt** (to take the square root of an expression).

For information on the syntax/usage of these functions you may find these two resources useful (search for the name of the function):

https://cppreference.com
http://cplusplus.com