Project Report

Spring 2025

Name: Rakshith
Project: How Far

Note: As you write each section, try to be as clear and detailed as possible. Your goal is to communicate your thought process and work clearly. Don't worry if you faced challenges or made mistakes; discussing these is a valuable part of learning and shows your problem-solving skills! Remember, there's no single 'right' way to do these tasks, so be creative and honest in your responses.

Problem Statement (2-3 Paragraphs):

The goal of this assignment was to create a program that calculates the distance between cities on Earth using the Great Circle Distance formula. The program loads a list of world cities from a CSV file and allows the user to enter a city name. It then finds that city's latitude and longitude and calculates how far it is from all other cities. The output shows the 10 closest cities in kms.

The inputs are the city names from the CSV file. The program outputs the distances in kilometers, formatted to four decimal places, along with the city and country names. The user can also type "list" to see all available cities or "quit" to exit. If the user enters an invalid city name, the program prints an error and asks again.

The program checks for input errors and skips any invalid or blank entries in the CSV file. This makes sure it keeps running without crashing even if there are issues with the data.

Design (1-3 Paragraphs):

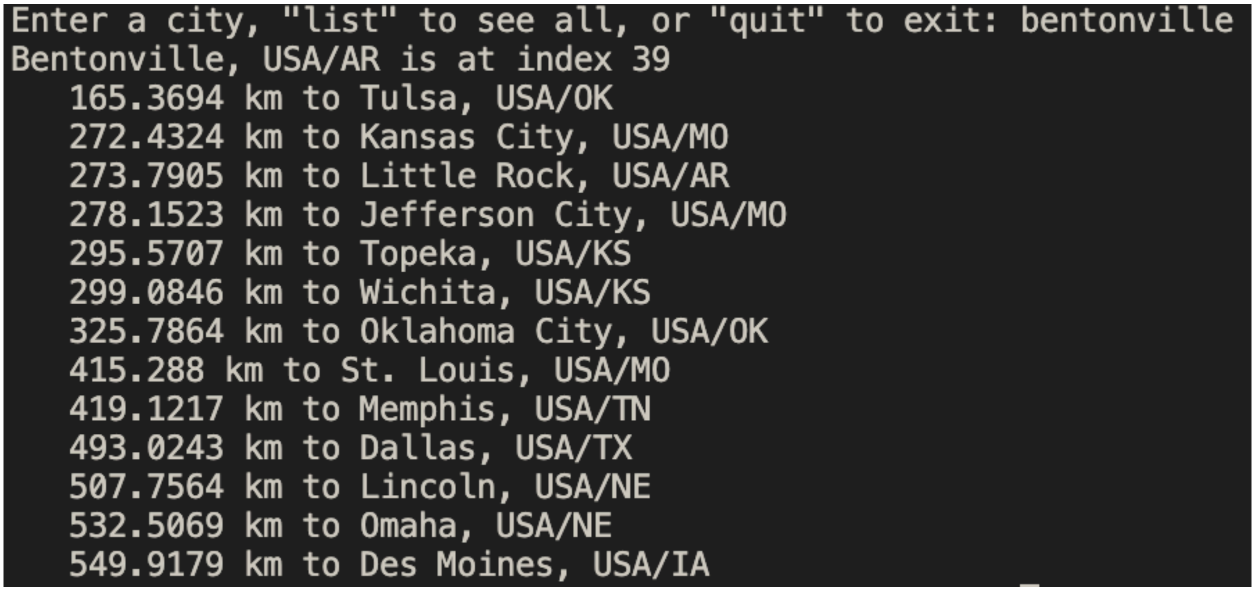
The program is build with simple logic and basic Python tools. It uses the csv library to read city data and the math library to perform calculations. Each city is stored as a small list containing its name, country, latitude, and longitude. When the user enters a city, the program loops through the list to find it, then compares it to every other city to calculate the distances.

The math formula uses trigonometry to measure distances on a sphere, converting degrees to radians before applying the formula. The results are stored in a list, sorted by distance, and the first 10 are displayed. The code avoids complex features and uses plain loops and lists to keep it beginner friendly.

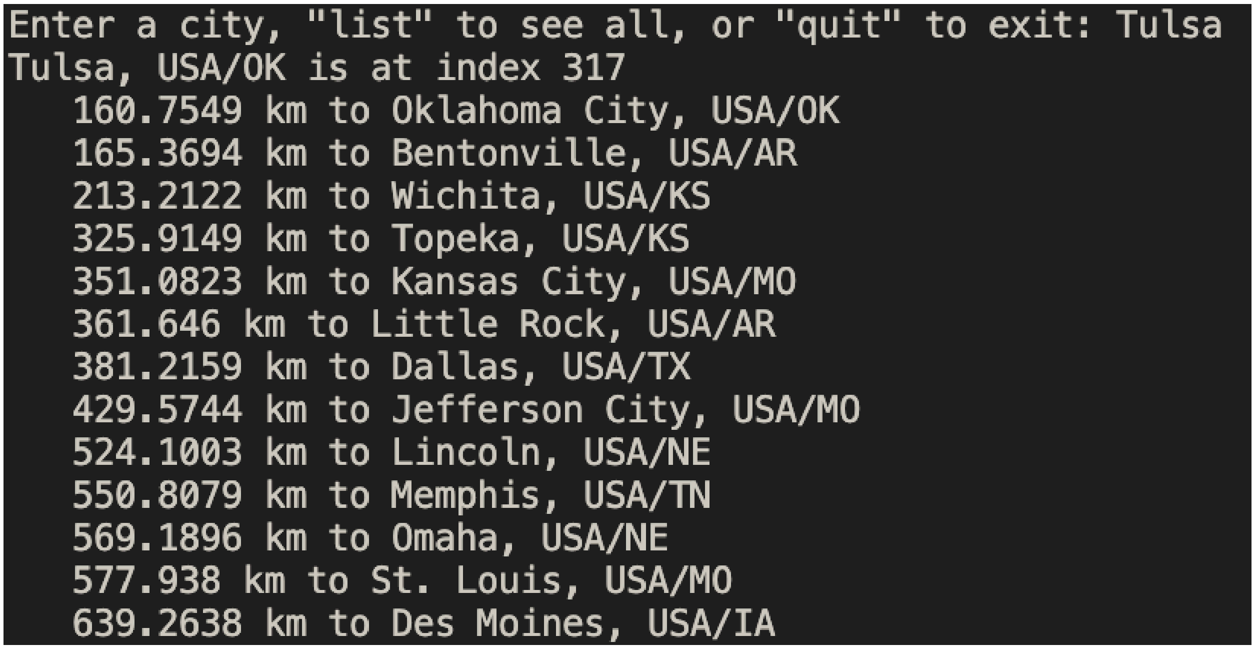
If rewritten, the code could be improved by splitting up into smaller functions or using dictionaries for faster searches. For this project, clarity was the main goal.

Testing (1-2 Paragraphs + screenshots of 3 test cases):

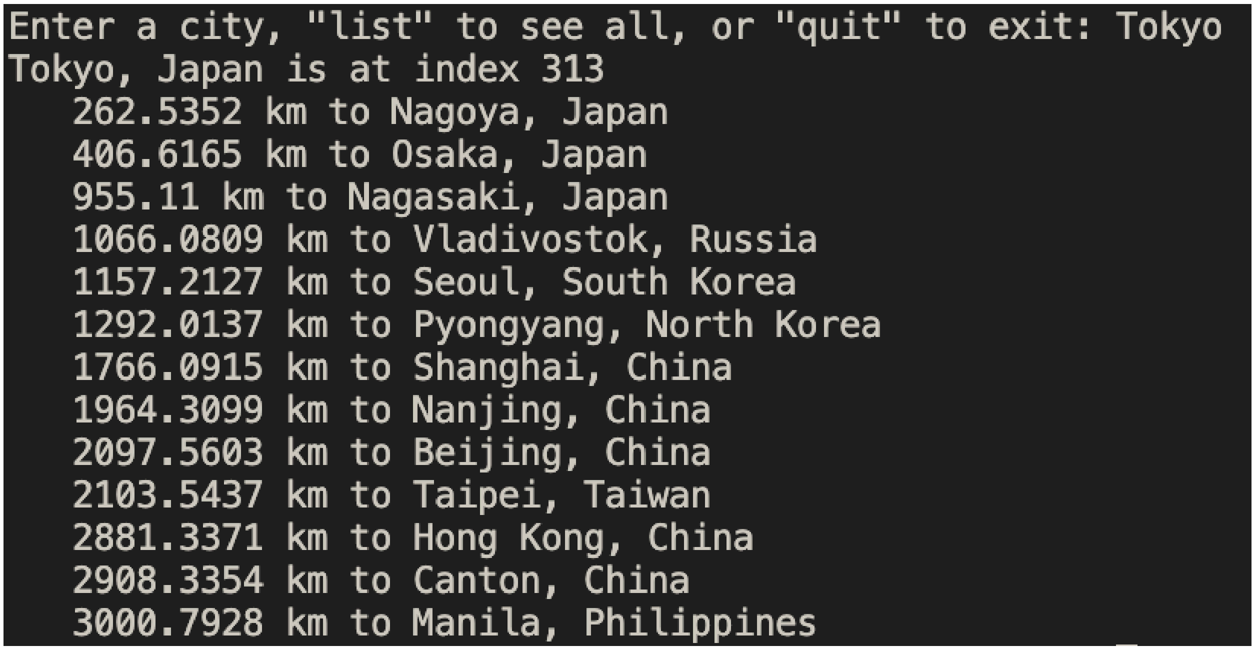
I tested the program by entering cities like "Bentonville", “Tulsa”, and “Tokyo” to make sure it worked for both local and international cities. The distances matched the values to the test examples. I also tested “list” and “quit” commands and it did work. I also tried invalid inputs like “Fake City” and the error handling worked. The results were accurate in all test cases. The program correctly listed the closest cities and their distances in kilometers with four decimal precision. The loop also repeated smoothly until “quit” was entered.



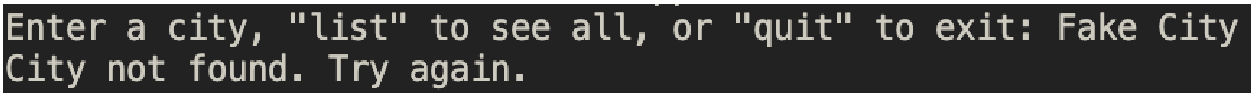
```
Enter a city, "list" to see all, or "quit" to exit: bentonville
Bentonville, USA/AR is at index 39
   165.3694 km to Tulsa, USA/OK
   272.4324 km to Kansas City, USA/MO
   273.7905 km to Little Rock, USA/AR
   278.1523 km to Jefferson City, USA/MO
   295.5707 km to Topeka, USA/KS
   299.0846 km to Wichita, USA/KS
   325.7864 km to Oklahoma City, USA/OK
   415.288 km to St. Louis, USA/MO
   419.1217 km to Memphis, USA/TN
   493.0243 km to Dallas, USA/TX
   507.7564 km to Lincoln, USA/NE
   532.5069 km to Omaha, USA/NE
   549.9179 km to Des Moines, USA/IA
```



```
Enter a city, "list" to see all, or "quit" to exit: Tulsa
Tulsa, USA/OK is at index 317
   160.7549 km to Oklahoma City, USA/OK
   165.3694 km to Bentonville, USA/AR
   213.2122 km to Wichita, USA/KS
   325.9149 km to Topeka, USA/KS
   351.0823 km to Kansas City, USA/MO
   361.646 km to Little Rock, USA/AR
   381.2159 km to Dallas, USA/TX
   429.5744 km to Jefferson City, USA/MO
   524.1003 km to Lincoln, USA/NE
   550.8079 km to Memphis, USA/TN
   569.1896 km to Omaha, USA/NE
   577.938 km to St. Louis, USA/MO
   639.2638 km to Des Moines, USA/IA
```



```
Enter a city, "list" to see all, or "quit" to exit: Tokyo
Tokyo, Japan is at index 313
   262.5352 km to Nagoya, Japan
   406.6165 km to Osaka, Japan
   955.11 km to Nagasaki, Japan
   1066.0809 km to Vladivostok, Russia
   1157.2127 km to Seoul, South Korea
   1292.0137 km to Pyongyang, North Korea
   1766.0915 km to Shanghai, China
   1964.3099 km to Nanjing, China
   2097.5603 km to Beijing, China
   2103.5437 km to Taipei, Taiwan
   2881.3371 km to Hong Kong, China
   2908.3354 km to Canton, China
   3000.7928 km to Manila, Philippines
```



```
Enter a city, "list" to see all, or "quit" to exit: Fake City
City not found. Try again.
```

Conclusion (1 paragraph)

This project met the goal. I learned how to use the Great Circle Distance formula to find real-world distances between coordinates. The code is simple, easy to read, and handles user input well. In the future, I will add more user options, such as displaying more cities, or saving the results to a file, but the program itself runs exactly like expected.