UNIVERSITY OF PARMA

Department of Engineering and Architecture
Degree course in Computer, Electronic and Communications Engineering
PRACTICAL TEST IN COMPUTER SCIENCE & PROGRAMMING LABORATORY
January 16th , 2025

Name: _____ Surname:_____ Matr:_____ Workstation: _____

Write a program using the C language (name the project with your student **<ID>**) that behaves as described below. The available time is 120 minutes. At the end of the time, the saved files on **U:** are going to be automatically collected. Additional documents, files... are available in **T:\Bertozzi**, it is recommended to use WordPad to read text files.

The file "europa.csv" contains information about a number of European cities. Specifically, it is in CSV format and each line, excluding the header, contains: city name (variable length, maximum 56 characters), country code (2 letters), latitude, and longitude in decimal format, all separated by ";". For example:

```
Parma;IT;44.8044;10.3325
```

Develop a program that:

1. (12-15 points) Reads and stores the contents of the file into an appropriate data structure.
2. (3-5 points) Asks the user for the name of a city and searches for it in the data structure defined in the previous step.
   3. (3-4 points) Prints its geographical coordinates in sexagesimal format.
   4. (2-6 points) Generates a file named "dist.csv" in CSV format which, line by line, contains: the names of the previously stored cities and the straight-line distance from the city entered in step #2, separated by ";".
5. (3 points) Modify the program to sort the data read in step #1 based on the city name and to enable binary search using bsearch() for the search in step #2.

Dynamic memory allocation should be used where appropriate. Defining and using functions for the first 4 points allows for additional points (the numbers in parentheses indicate the maximum points achievable without and with the use of functions, respectively).

Note:
Given the coordinates of two cities, their straight-line distance in kilometers can be calculated using the well-known formula: distance = acos(sin(lat1) × sin(lat2) + cos(lat1) × cos(lat2) × cos(lon2-lon1)) × 6371.
It is important to remember that the trigonometric formulas provided with the math.h library expect angles to be in radians and that M_PI is the constant for π.

The code should be developed following the proposed order (if you want you can anyway anticipate #5). The correction stops at the first incorrectly implemented step.