CSCE 2211 Fall 2023 Applied Data Structures
Assignment #3

Dr. Amr Goneid ***Date: Thu October 12, Due: Tuesday October 24, 2023***

The problem: Building a Dictionary for Major Cities in the World

One of the many applications of Binary Search Trees is building a dictionary structure in which a key is used to search for a given item and to provide relevant information if the key is found.

Objective

Implement a dictionary for the Major Cities in the world in the form of a Binary Search Tree (BST). The data for major population cities on Earth is provided in the Excel file **Major City List**

An entry for a city in the file gives the city name, its latitude, its longitude and the country in which the city exists. The objective of the dictionary is to be able to retrieve the city information given the city name as the key.
Your dictionary should be a ***growing*** one, i.e., it should be able to add major cities in separate runs.

Required Implementation:

1. **The BST ADT**
   A Full implementation of the BST class is found at:
   http://www1.aucegypt.edu/faculty/cse/goneid/csce2211/codes.rar

   The ***BST*** ADT can be implemented with the following member functions:
   - ***Constructor:*** Construct an empty tree
   - ***Destructor:*** Destroy tree
   - ***insert***: Insert an element into the tree
   - ***empty:*** Return True if tree is empty
   - ***search:*** Search for a key
   - ***retrieve:*** Retrieve data for a given key
   - ***traverse:*** In-Order Traversal of the tree
   - ***preorder:*** Pre-order traversal of the tree
   - ***levelorder:*** level-order traversal of the tree
   - ***remove:*** Remove an element from the tree

   Design and implement a template class ***BST*** with a **minimum** of the above member functions. Use a template node class that consists of a **key** part and a **data** part. The node should also have a pointer to the left sub-tree and a pointer to the right-sub-tree.

2. Implement a program to build, maintain and use a ***BST*** dictionary with City Name representing the key part while the data part includes the latitude, the longitude, and the country.
   The dictionary should provide the following functions:
   - Generate the dictionary in memory.
   - Save the dictionary to disk as a text file.
   - Download a dictionary from disk into memory (as a ***BST***)
   - Update a dictionary with added cities if needed.

- Search the dictionary given the City Name and obtain the latitude, the longitude, and the country.
- Produce a listing of all dictionary entries.
- Find the distance between two cities in the dictionary **(see below how to compute the distance between two cities given their latitudes and longitudes).**

Notes on Implementation:

1. **Saving the dictionary on disk:** Given the ***BST*** in memory, save the nodes in a text file on disk using **traversal**. Be careful in choosing the traversal method in order to obtain the same BST when you download the dictionary back from disk.
2. Your program should distinguish between the following runs:
    - **Initial Run:** Assumes that there are no previous dictionaries stored on disk. Constructs the initial dictionary from initial City List then saves the resulting dictionary to disk.
    - **Cumulative Run:** Assumes the presence of a previous dictionary on disk. It will download that dictionary from disk into memory (as ***BST***) and update it if necessary. The updated dictionary is then stored back on disk.
    - **Query Run:** Assumes the presence of a previous dictionary on disk or already downloaded into memory (as ***BST***). It will be used in searches, listings or for computing the distance between two cities.
3. **How to compute the distance between two cities:**
   For a given city, Latitude and longitude are given in DMS units (Degree, Minutes, Seconds). Latitude is North (N) (0 to +90°), or South (S) (0 to -90°), 0 is the equator. Longitude is East (E) (0 to +180°) or West (W) (0 to -180°), 0 is the prime meridian (Greenwich, England).
   The great-circle distance between two locations on earth is the shortest distance over the earth's surface (giving an 'as-the-crow-flies' distance between the points, ignoring any irregularities). Given the latitude φ (radians) and longitude λ (radians), the distance between two locations 1 and 2 can be calculated using the "Haversine" folrmula:

$$a = \sin^2(\Delta\varphi / 2) + \cos\varphi_1 \,.\, \cos\varphi_2 \,.\, \sin^2(\Delta\lambda / 2)$$
$$c = 2\, \mathrm{atan}2(\sqrt{a}, \sqrt{1-a})$$
$$d = R\,.\,c\ (Km)$$

   where ***R = 6371. 137 km*** is the mean Earth's radius, $\Delta\varphi = \varphi_2 - \varphi_1$, $\Delta\lambda = \lambda_2 - \lambda_1$ and ***atan2(..)*** is the 4 quadrant arctangent.
   To convert to radians, we first convert to decimal:
   α (Decimal Degrees) = degrees + (minutes/60) + (seconds/3600)
   with a (-) sign for (S) latitudes and (W) longitudes.
   Then we convert to radians using α(radians) = α (Decimal Degrees) × π / 180°