Caua Melo

Search by Zip Code

In this program, the SearchZipCode class is designed to filter cities based on a specific zip code from user input. The class inherits the Visitor class, allowing it to traverse a Binary Search tree that stores city information that was built before this. When the user inputs a zip code, SearchZipCode stores it and compares it with each city's zip code as it traverses through the nodes of the Binary Search Tree. This is where the visit method comes in (inherited from the visitor class), it is used in every node of the tree, comparing the zip codes from the file to the user input. If a match is found, the city is added to a "matches" vector, which stores all cities matching the input. Once traversal is complete, the getZips method returns the vector containing the search results.

In the menu class, the searchZip function initiates the search process by prompting the user to enter a zip code and then creating an instance of the SearchZipCode class with that code. The Binary Search Tree containing the information extracted from the US Cities file (myTree) then performs an in-order traversal, calling the visit method on each City object in the tree. After the traversal, the results are stored in a results vector. If the results vector is empty, a message is displayed saying that there were no matches. If matches are found, a header is printed separately, and each city in the results vector is displayed. The results are displayed in a table format

Display minimum population

In this program, the MinPop class inherits the Visitor class to traverse each CIty object in the Binary Search Tree. The visit function is used to do any operations in each node of the Binary Seach Tree. While traversing the Tree, the visit function is used to compare each city's population to the current minimum population (minPopulation), and if the city has a lower population than the current minimum it becomes the new current minimum, updating minPopulation to that city's population, and minCity is updated to the city itself. After all the nodes of the tree are visited, the getMinPop method returns minCity, which contains the City object with the lowest population in the tree.

Once the city with the lowest population is found by the visit function, the minPopulation function in the Menu class gets the city's information and formats it. First, the function will convert the population of the city to an integer, then it will display a header, and finally, it will display the city's information. The population number will be formatted to be displayed with commas using the intWithCommas function. The information is displayed in a table format.