جامعة بيرزيت
BIRZEIT UNIVERSITY

Data Structures - Project 3

In this project, you will implement an AVL tree data structure using the Names files (attached). The tree nodes will represent a Name object that has the following attributes:

- Baby name
- Baby gender
- A linked list that contains the frequencies of this baby name in different years.

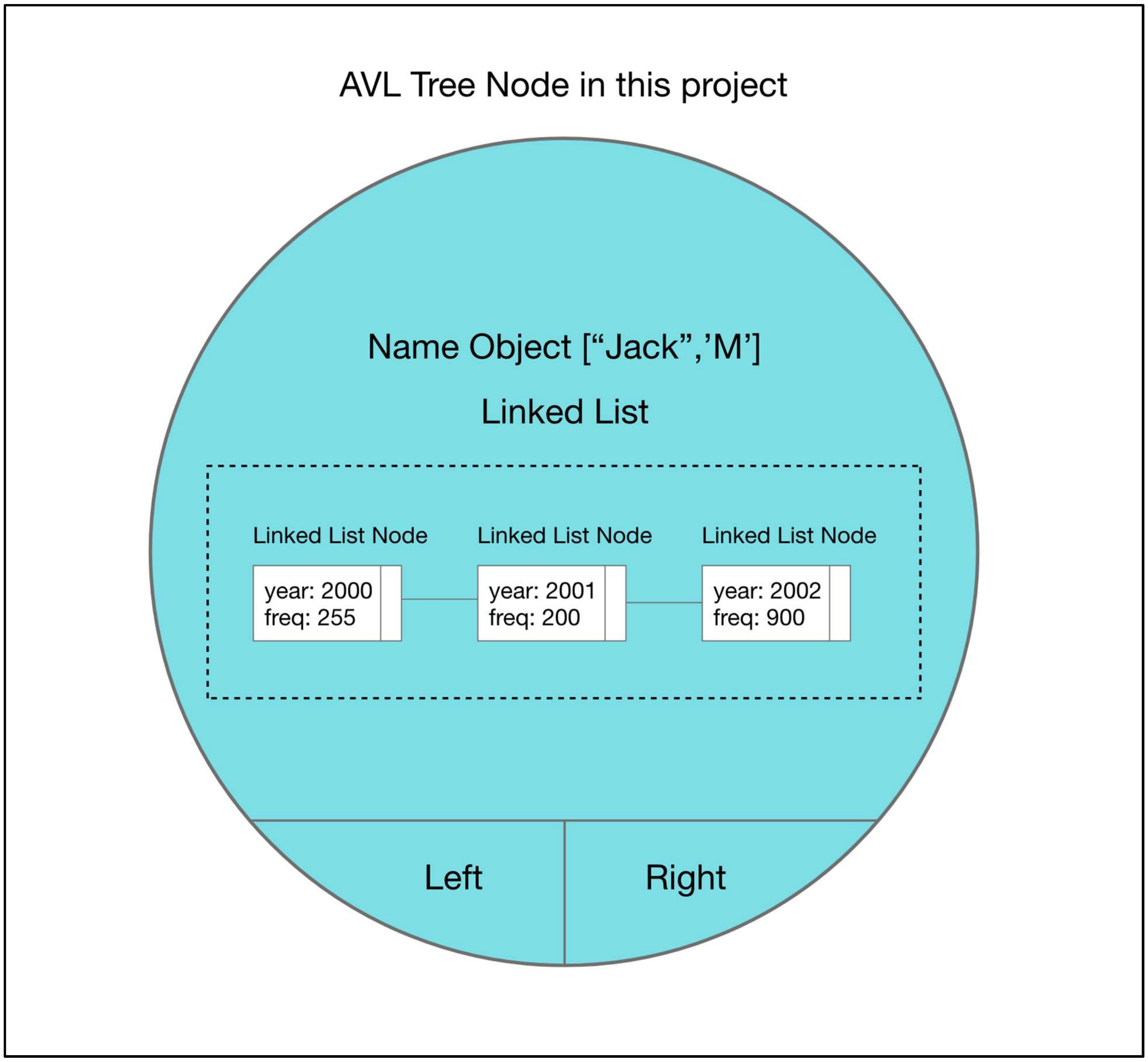
The name object is comparable using both the baby's name and gender together.

You will represent all the names in all the files. This means that you will **load all the files** with all the names in them, file by file and add them to the tree. There will be two cases for this:

1. A name-gender **does not exist** in the AVL tree (it is the first time that it appears in any of the files), in this case, the name will be added as a new node to the tree. Then you will insert a node in the linked list for the frequency of this year.
2. A name-gender **exists** as a node in the tree (the same name and gender were added by a file from a previous year); in this case, you will only insert an object into the **linked list** of this tree node.

The nodes inserted into the linked list of the frequencies will only have the **year** and the **frequency** (since the name object is in the tree node). So create a Frequency class that has only the year and frequency. Nodes inserted in the linked list will be sorted by the year meaning that you will need to compare the frequency objects by year.

Below is a visualization of how a node in the AVL tree will look like.



The node has the Name Object (Name , Gender) and the linked list of frequencies for this object for different years. It also has the left and right like any other AVL tree node.

The following functionality needs to be implemented

- ### Search for a Name

In this part, the user will be able to enter a name and chose a gender (**both**) and then you will need to list that object (the name and gender) along with the frequencies for all the years if exists. An example of how the output should like is below. Note that this is just an example you are free to implement the showing process as you wish.

Ali	**Male**	**2000**	**2001**	**2002**	**..**
		200	255	888	..

- ### Average Frequencies of a Name

In this part, the user will enter a name and chose a gender and the goal will be to display the average frequency for this name over the different years in which it appears.

- ### Name with max frequency

In this part, you will need to show the user the name that had the maximum frequency all over the years. Which means that it was the most popular name when adding the frequencies in the linked list of frequencies and years.

- ### Total number of babies in a selected year

In this part, you need to ask the user for a year, and the goal is to traverse the tree and calculate the total number of babies in the selected year.

- ### Export the AVL tree data to a file

In this part, you need to traverse the AVL tree **level by level** and write the information to an output file using the following format:

Baby_Name, Baby_Gender, Total Frequency

- ### Graphical User Interface (GUI)

A GUI is required to load the files and fill up the AVL and the linked lists in them. The GUI has a text field to type the directory from where the name files are located. It is also required to do all the functionality above. Use your imagination for how the GUI looks as long as it looks **PROPER** and does **ALL THE CORRECT FUNCTIONALITY**.

Good Luck!