جامعة بيرزيت
BIRZEIT UNIVERSITY

COMP242 -Data Structures - Project 4

In this project, you will implement a **Hash Table** data structure using the Names files (Same files as in the 3rd project). The Hash Table nodes will represent a **HashName** object that has the following attributes:

- Baby name
- Baby gender
- A **max heap** data structure that contains the frequencies of this baby name in different years.

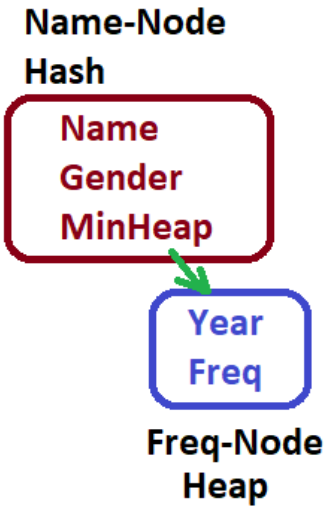
The nodes in the Hash Table are managed by a unique key consists of baby's name + baby's gender.

You will represent all the names in all the provided name files in a given folder. This means that you will **load all the files** with all the names in them, file by file and add them to the hash table. There will be two cases for this:

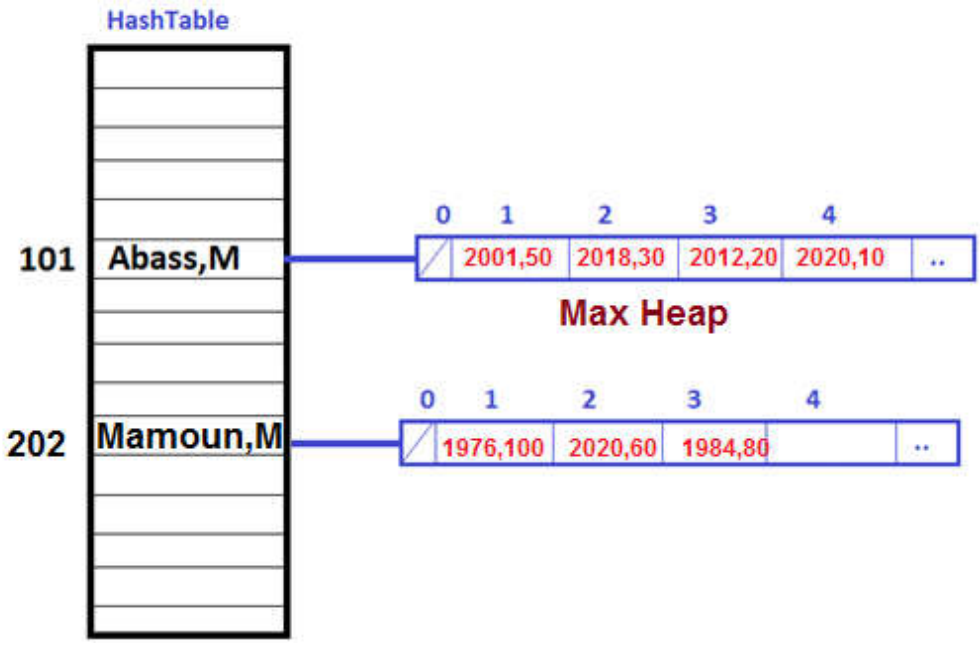
1. A name **does not exist** in the hash table (it is the first time that it appears in any of the files), in this case, the name will be added as a new **HashName** record to the hash table. Then you will insert a heap node into the **max heap** for the frequency of that year.
2. A name **exists** as a node in the hash table (the same name and gender were added by a file from a previous year); in this case, you will only insert a new heap node into the **max heap** of this hash table node.

The heap nodes inserted into the max heap of the frequencies will only have the **year** and the **frequency** (since the name object is in the hash table node). So create a **Frequency** class that has only the year and frequency. Nodes inserted in the heap will be compared by the frequency.

Below is a visualization of how a node in the Hash table will look like.



The **HashName** node has the baby's name and gender and the max heap data structure of frequencies for this object for different years.



Notes:

- We will use **quadratic probing** for solving the collisions when hashing. Use the string hashcode method of (baby's name + baby's gender).

The following functionality needs to be implemented

- Load names files from a specific folder and an estimation of data size (i.e. average number of name records in files).
- Add new name/gender record.
- Delete a name record.
- Add a new name's year/freq record.
- Update a name's year/freq record.
- Search for 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 (sorted by frequency). 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.

Abas	**M**	**2001**	**2018**	**2012**	**2020**
		50	30	20	10

- Name with max frequency

In this part, you will need to show the user the name that had the maximum frequency in any year. (E.g. from the previous figure, Mamoun has a maximum of 100 in 1976)

- Graphical User Interface (GUI)

A GUI is required to load the files and fill up the Hash table and the heaps 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!