SSW 215 Individual Software Engineering

Homework 6

Purpose: Practice using arrays and hashes in Ruby

This is an individual assignment. You may work with others to understand the problem and solution, but all programming must be your own. Do not copy anyone else's code.

1. Write a Ruby program **`family.rb`** that reads a simplified version of GEDCOM data about individuals and families, and prints out a summary of the information.

   GEDCOM is a standard format for genealogy data developed by The Church of Jesus Christ of Latter-day Saints. It is a line-oriented text file format where each line has 3 parts, separated by blank space:
   - level number (a non-negative integer)
   - tag (a string of 3 or 4 characters)
   - arguments (an optional character string)

   All lines (or records) begin with a level number. Records with level numbers greater than 0 are always in the form:
   ```
   <level-number> <tag> <arguments>
   ```
   Records with level number 0 are in the form:
   ```
   0 <xref-id> <tag>
   ```

   The field between the "0" and the tag is a unique identifier used to identify an individual or a family.

   Level numbers provide a mechanism for grouping records in hierarchical groups: all records with larger level numbers than the current record belong to a group headed by that record.

   For this assignment there are 8 different types of records:

Level	**Tag**	**Arguments**	**Belongs to**	**Meaning**
0	`INDI`	unique id between "@" delimiters	top level	Individual record
1	`NAME`	String	`INDI`	Name of individual
1	`BORN`	integer	`INDI`	Birth year of individual
0	`FAM`	unique id between "@" delimiters	top level	Family record
1	`MARR`	integer	`FAM`	Year of marriage
1	`HUSB`	pointer to individual	`FAM`	Husband in family
1	`WIFE`	pointer to individual	`FAM`	Wife in family
1	`CHIL`	pointer to individual	`FAM`	Child in family

Note that individual records and family records do not have to appear in any order. For example, the records about an individual may appear before or after the records for the families in which they participate as a spouse or child. Individual records and family records may be interspersed in the input file. Level-1 records may appear in any order after the level-0 record to which they belong (but will appear before the next level-0 record).

The output of your program should include the following:

- A list of all the individuals found in the input file, in order by unique identifier, showing for each:
  - Unique Identifier
  - Name
  - Year of birth
- A list of all the families found in the input file, in order by unique identifier, showing for each:
  - Unique Identifier
  - Names of the husband and wife
  - Year of marriage
  - List of children in the family in birth order (oldest first), showing for each:
    - Unique Identifier
    - Name
    - Year of birth

Example input:

```
0 @I01@ INDI
1 NAME Barack Obama
1 BORN 1961
0 @I03@ INDI
1 NAME Natasha Obama
1 BORN 2001
0 @I04@ INDI
1 NAME Malia Obama
1 BORN 1998
0 @F01@ FAM
1 CHIL @I04@
1 HUSB @I01@
1 CHIL @I03@
1 WIFE @I02@
1 MARR 1992
0 @I02@ INDI
1 BORN 1964
1 NAME Michelle Robinson
```

Example output:

```
Individuals found in file:
--------------------------
@I01@: Barack Obama [1961]
@I02@: Michelle Robinson [1964]
@I03@: Natasha Obama [2001]
@I04@: Malia Obama [1998]

Families found in file:
-----------------------
@F01@:(Barack Obama + Michelle Robinson) [1992]
--- Children:
------- @I04@: Malia Obama [1998]
------- @I03@: Natasha Obama [2001]
```

You may test your program with any test data you wish, but you must submit the results of running your program on the file **`stevens.txt`**

Upload files of:

- your program **`family.rb`**
- your test results