Introduction

The test exercise is based around building a graph of people, and their relationships with each other.

Instructions

Inside this zip file you will find a working gradle project structure with a build system, but no code in it. All of the exercises will be based around these files.

If you aren't familiar with gradle you only need to know one command, `./gradlew clean test` (unix) or `gradlew clean test` (windows), to build and test your code from the command line, but your chosen IDE should be able to import the project, perform the build and run the tests for you.

General guidance

Your solution should be delivered as Java or Groovy code, and a set of tests written in Java or Groovy that validate your solution meets the requirements of the exercises. These tests should automatically run when `./gradlew clean test` is run in the root of the project.

We are not looking for robust error checking in this exercise, we are looking for clean and concise code that reads the files exactly as they are presented, and performs the tasks specified. More specifically we are looking for

- An in-memory only solution, no use of databases or other persistent state stores.
- Neat, well structured and easily readable code
- Good use of Object Orientated modelling - we want to see entities represented properly as objects.
- Use of libraries and utility methods to save time and code
- Well thought through algorithms
- Comments to explain anything that you think needs further clarification
- No cheeky shortcuts, for instance hard coding values or writing code that does not work in the general case!

For the purposes of this test, to make your life easier we are **not** concerned about:

- Comprehensive javadoc
- Good/comprehensive exception handling

If you are unfamiliar with Gradle and you wish to use external libraries, please look in the file `build.gradle.kts`, where there are examples showing you how to include Guava and Apache Commons Lang in your project. You can easily use any library published in the JCenter repository by adding the details into the Gradle build file.

Completing the exercises

When all exercises are completed please zip up your whole solution, including all the gradle files. We should be able to unzip and build, and test your answers by running `gradle clean test`. For the avoidance of doubt, we will not consider your submission and it will not be evaluated unless this single command cleanly compiles your code, runs your tests and all of the tests pass.

Exercises

Exercise 1

Please implement code and data structures that read the files

- src/test/java/resources/people.csv
- src/test/java/resources/relationships.csv

and use them to build an in-memory data structure that represents the people in the file and their relationships with each other.

Exercise 2 - Validate correct people loaded

Write a test to validate that you have loaded the expected number of people.

Exercise 3 - Validate correct relationships loaded

Write a test to validate that the following people have the correct expected number of connections to other people

- Bob (4 relationships)
- Jenny (3 relationships)
- Nigel (2 relationships)
- Alan (0 relationships)

Exercise 4 - Write a method that calculates the size of the extended family

Write a method which, when passed the object representing a particular person, returns an int representing the size of their extended family including themselves. Their extended family includes anyone connected to them by a chain of family relationships of any length, so your solution will need to work for arbitrarily deep extended families. It should **not** count their friends. Write tests that validate this returns the correct result for the families of:

- Jenny (4 family members)
- Bob (4 family members)