Simple Persistence

Answer the following questions after completing Phase 3.

Submit both your final zipped up code folder and this completed document to the LMS before the due date.

1. What is the performance time, in milliseconds, when calling PrintPeopleDetails from Phase 1?

2182

1. What is the performance time, in milliseconds, when calling PrintSerializedDetails from Phase 3?

2073

* 1. What Advantages and Disadvantages does serializing the Employee objects have on the following?
     1. Your Code?

Makes my code harder to write, also more complicated

* + 1. On Performance?

Performance is better, it was just a wee bit faster

* + 1. On Maintenance of your application?

It forces us to use the same deserialization process, which means we are stuck doing things the same way, unless we go through the effort of converting our files

1. What advantages does loading all Employee records in a hashmap/dictionary have?

It made things real fast

1. What would happen to memory utilization in your application if you loaded a hashmap/dictionary with employees objects at application startup?

It uses so much more memory

1. Assume your application is starting cold, and you'll need to print the data for employee id 1001. Which approach to storing and accessing data would you select, and why? (searching individual files vs loading all records into a hashmap first, then searching)

I’d just search individual files as long as I’m doing operations on only one file

1. What if we wanted to print the Employee Id for the employee with the last name of “Fox”. Does your implementation of FindEmployeeByLastName seem inefficient at all?

Not really, it took like 14 milliseconds to find fox

1. If we searched by last name often, what could we do to make searches by last name faster?

Make a hash map with keys that are the lastnames

1. Each small Employee file takes up 24 bytes of disk space. How much total space on disk does our original 10,000 small employee records take up?

276,688 bytes (270KB)

1. Each large Employee file takes up 40KB of disk space. How much total space on disk does our original 10,000 large employee records take up?

409,600,000 bytes (390MB)

1. Does the increase in individual employee record size change your response to question 5.) above (Storing records in individual text files vs loading them into a HashMap in memory?) Why or why not?

Yes, as it would be significantly faster and more consistent to use a hash map rather than search through so many large files, to the point that it is worth the memory space.