1 Instructions

1.1 What to do

- Ensure Gradle is installed on your system
- Import code into your favorite IDE
- Implement all classes and functions. You can check todo sections for hints.
  - A default constructors should exist for all your domain classes
  - ***READ CAREFULLY*** the java doc before implementing any code
  - For list in the many side of associations always append "s" in your variable name e.g. List<Apple> apples;
- Run the test cases and make sure they pass.
- A failing test case implies you've a problem either at your logic or class definitions. Please review your code again and fix it till you pass all the cases.
- To achieve full mark: all your test cases should pass.

1.2 Sumbission

- Upload **ALL** the java source files inside (/src/main/java) to sakai.
- **DO NOT ZIP** any file

1.3 Warning

- Upon submission ***your code must be free of compilation errors***.
- Otherwise, no credit will be received as the automated build will fail to run the test cases on your submission.

At the very least, leave your partial solution commented so that the rest of your code can still compile.

1.4 Notes on grading

- A different set of test cases will be used during grading (they're not provided to you).
- With correct implementation, those extra cases are guaranteed to pass as well. They exist to prevent someone from hard-coding values just to pass test cases!

1.5 Pro tip *(Optional)*

- If you've docker installed, you can just run the command in the Makefile. This will build/test your code without even a need for IDE or java installed.
- If you're on Unix-like OS , just run ‘make run‘ from the project root dir.

1.6 You're the Instructor?

Please check InstructorOnly section.

2 UML Class diagram *[20pts]*

Implement the following in *Java*.

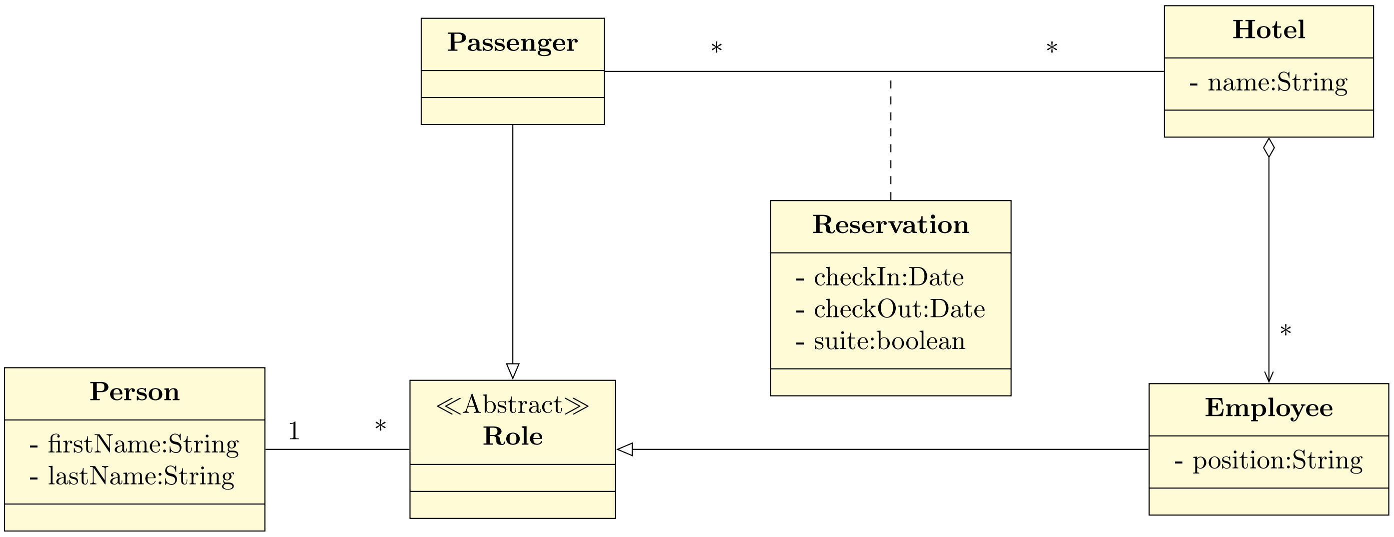

Figure 1: Hotel reservation system

3 Functional Programming

Check JavaDoc/todo at FunctionUtil.java for detailed description

3.1 Finding people with prefix *[10pts]*

$$peopleWithFirstNameWhoReservedAtHotel(persons, prefix, hotelName, year) \to list[String]$$

Returns the capitalized last names of persons whose first name begins with prefix and have stayed at hotelName during a certain year.

3.2 Find VIPs *[70pts]*

$$vipPersonsWhoStayedAtLeastKNights(persons, hotelName, year, k) \to List[String]$$

Returns the names of persons who stayed at a hotelName at least K nights(as VIP) during a certain year.