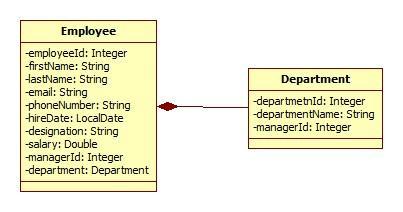
**Case Study for Steam API:**

Refer the classes given below to represent employees and their departments.



**Figure 20: Class Diagram used for Stream API**

Also create an EmployeeRepository class which is used to create and populate employee’s collection with sample data.

Create an EmployeeService class which queries on collections provided by EmployeeRepository class for following requirements. Create separate method for each requirement. (**Note:** Each requirement stated below must be attempted by using lambda expressions/stream API. It’s mandatory to solve all questions from following set. However, it is recommended to solve all questions to understand stream API thoroughly).

14.6: Find out the sum of salary of all employees.

14.7: List out department names and count of employees in each department.

14.8:Find out the senior most employee of an organization.

14.9: List employee name and duration of their service in months and days.

14.10: Find out employees without department.

14.11: Find out department without employees.--skip

14.12: Find departments with highest count of employees.

14.13: List employee name, hire date and day of week on which employee has started.

14.14: list employee name, hire date and day of week for employee started on Friday. (Hint: Accept the day name for e.g. FRIDAY and list all employees joined on Friday)

14.15: List employee’s names and name of manager to whom he/she reports. Create a report in format “employee name reports to manager name”.

14.16: List employee name, salary and salary increased by 15%.

14.17: Find employees who didn’t report to anyone (**Hint:** Employees without manager)

14.18: Create a method to accept first name and last name of manager to print name of all his/her subordinates.

14.19: Sort employees by their

* Employee id
* Department id
* First name