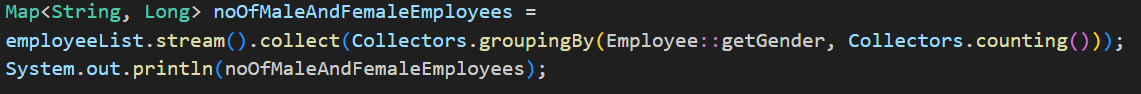
**How many male and female employees are there in the organization?**

**Print the name of all departments in the organization?**

****

**What is the average age of male and female employees?**

Map<String, Double> avgAgeOfMaleAndFemaleEmployees=

employeeList.stream().collect(Collectors.groupingBy(Employee::getGender, Collectors.averagingInt(Employee::getAge)));

System.out.println(avgAgeOfMaleAndFemaleEmployees);

**Query 3.4 : Get the details of highest paid employee in the organization?**

Optional<Employee> highestPaidEmployeeWrapper=

employeeList.stream().collect(Collectors.maxBy(Comparator.comparingDouble(Employee::getSalary)));

**Query 3.5 : Get the names of all employees who have joined after 2015?**

employeeList.stream()

            .filter(e -> e.getYearOfJoining() > 2015)

            .map(Employee::getName)

            .forEach(System.out::println);

**Query 3.6 : Count the number of employees in each department?**

Map<String, Long> employeeCountByDepartment=

employeeList.stream().collect(Collectors.groupingBy(Employee::getDepartment, Collectors.counting()));

**Query 3.7 : What is the average salary of each department?**

Map<String, Double> avgSalaryOfDepartments=

employeeList.stream().collect(Collectors.groupingBy(Employee::getDepartment, Collectors.averagingDouble(Employee::getSalary)));

**Query 3.8 : Get the details of youngest male employee in the product development department?**

Optional<Employee> youngestMaleEmployeeInProductDevelopmentWrapper=

employeeList.stream()

            .filter(e -> e.getGender()=="Male" && e.getDepartment()=="Product Development")

            .min(Comparator.comparingInt(Employee::getAge));

**Who has the most working experience in the organization?**

Optional<Employee> seniorMostEmployeeWrapper=

employeeList.stream().sorted(Comparator.comparingInt(Employee::getYearOfJoining)).findFirst();

**How many male and female employees are there in the sales and marketing team?**

Map<String, Long> countMaleFemaleEmployeesInSalesMarketing=

employeeList.stream()

            .filter(e -> e.getDepartment()=="Sales And Marketing")

            .collect(Collectors.groupingBy(Employee::getGender, Collectors.counting()));

