

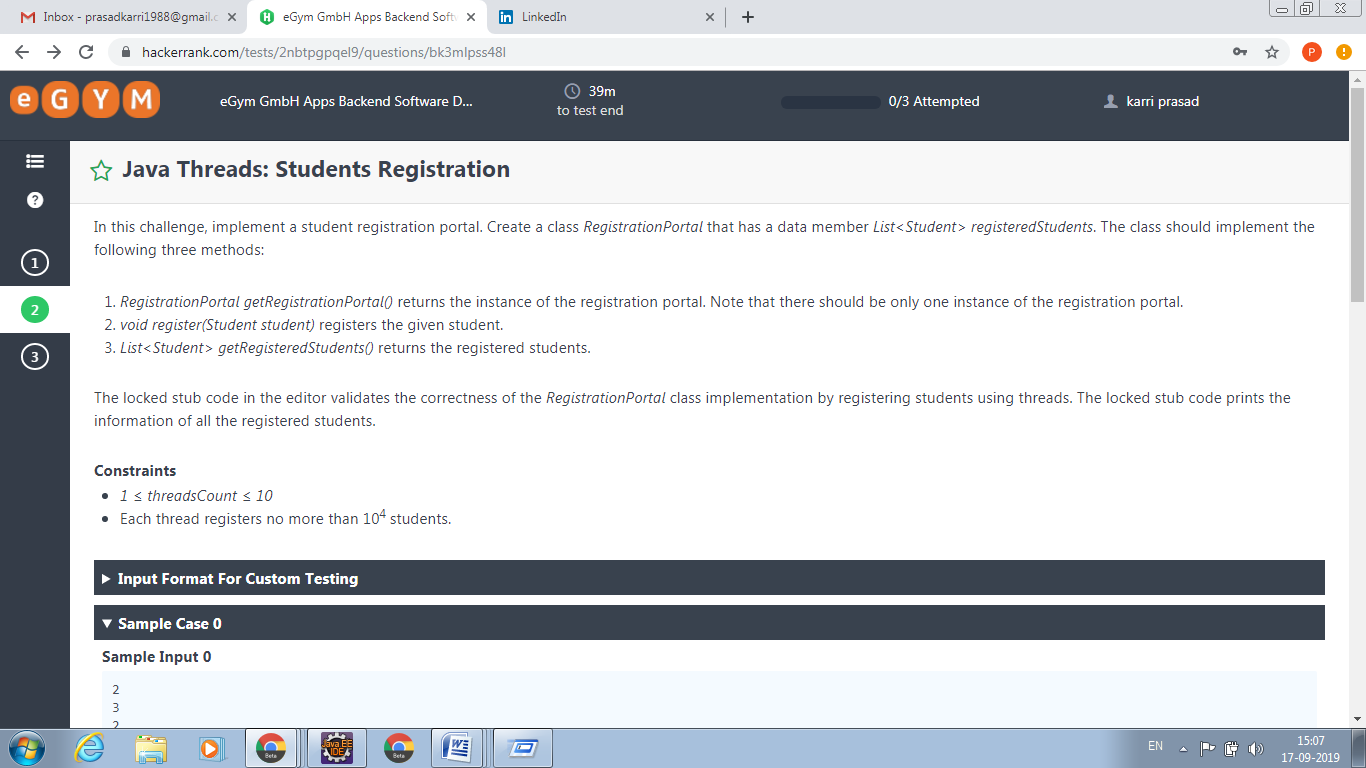
/\*

Enter your query here.

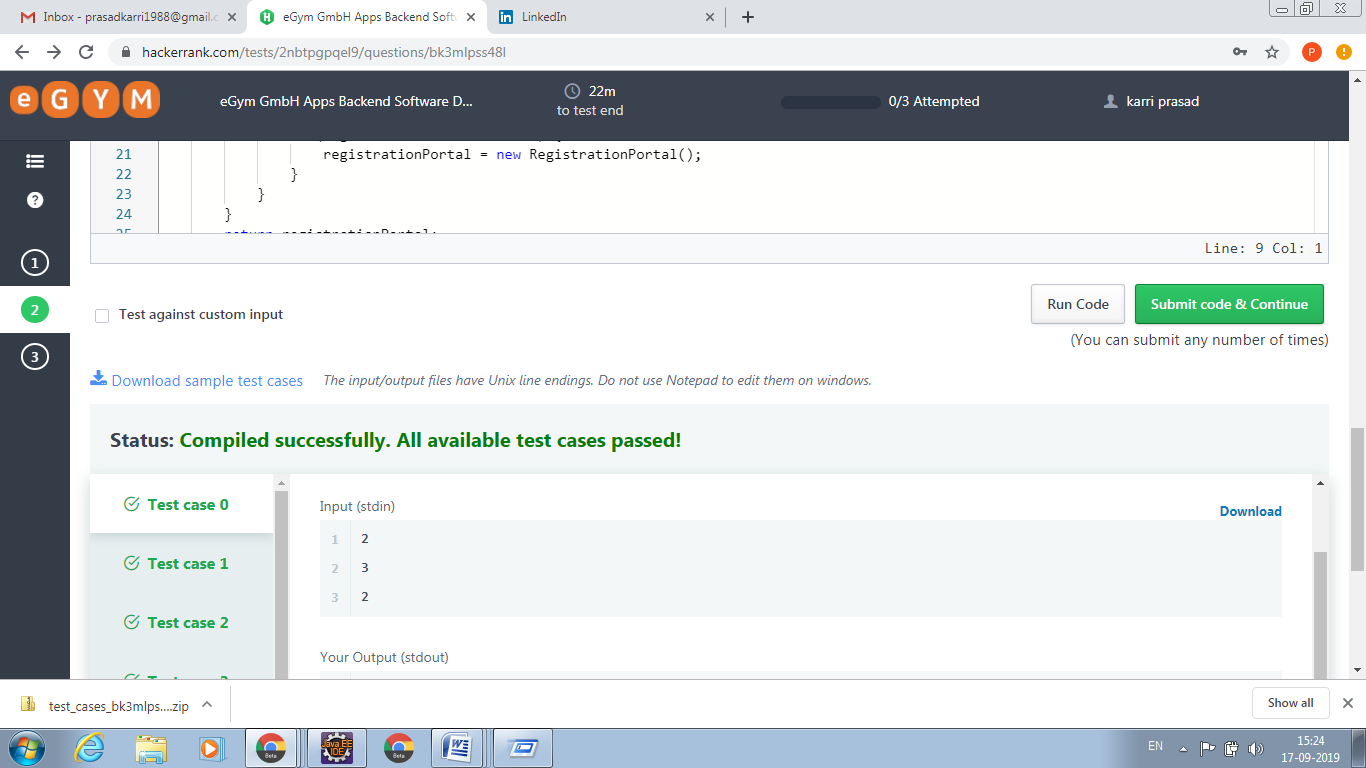
Please append a semicolon ";" at the end of the query

\*/

select c.name from company c, salary s where c.id=s.company\_id group by (company\_id) having avg(s.salary)>=40000







**Design Student Registration system based on certain conditions they are give.**

**import** java.util.ArrayList;

**import** java.util.List;

**class** RegistrationPortal {

**private** **static** RegistrationPortal *registrationPortal*;

List<Student> studentList=**new** ArrayList<Student>();

**private** RegistrationPortal() {

}

**public** **static** RegistrationPortal getRegistrationPortal() {

**if** (*registrationPortal* == **null**) {

**synchronized** (RegistrationPortal.**class**) {

**if** (*registrationPortal* == **null**) {

*registrationPortal* = **new** RegistrationPortal();

}

}

}

**return** *registrationPortal*;

}

**public** **void** register(Student student) {

**synchronized** (RegistrationPortal.**class**) {

studentList.add(student);

}

}

**public** List<Student> getRegisteredStudents() {

**return** studentList;

}

}

**class** RegistrationHelper {

**private** **final** RegistrationPortal registrationPortal;

**public** RegistrationHelper(RegistrationPortal registrationPortal) {

**this**.registrationPortal = registrationPortal;

}

**public** **void** register(Student student) {

**if** (**this**.registrationPortal != **null**) {

**this**.registrationPortal.register(student);

}

}

}

**class** Student {

**private** **final** String id;

**private** **final** String name;

**public** Student(String id, String name) {

**this**.id = id;

**this**.name = name;

}

**public** String getId() {

**return** **this**.id;

}

**public** String getName() {

**return** **this**.name;

}

}

**class** RegistrationRunnable **implements** Runnable {

**private** **final** RegistrationPortal registration;

**private** **final** **int** studentsCount;

**private** **final** String studentsIdPrefix;

**public** RegistrationRunnable(RegistrationPortal registration, **int** studentsCount, String studentsIdPrefix) {

**this**.registration = registration;

**this**.studentsCount = studentsCount;

**this**.studentsIdPrefix = studentsIdPrefix;

}

@Override

**public** **void** run() {

RegistrationHelper registrationHelper = **new** RegistrationHelper(registration);

**for** (**int** i = 1; i <= studentsCount; i++) {

String studentId = "id-" + studentsIdPrefix + "-" + i;

String studentName = "name-" + i;

Student student = **new** Student(studentId, studentName);

registrationHelper.register(student);

}

}

}

**import** java.util.Collections;

**import** java.util.Comparator;

**import** java.util.List;

**import** java.util.Scanner;

**public** **class** Solution {

**private** **static** **final** Scanner ***SCANNER*** = **new** Scanner(System.***in***);

**public** **static** **void** main(String[] args) **throws** InterruptedException {

**int** threadsCount = Integer.*parseInt*(***SCANNER***.nextLine());

Thread[] threads = **new** Thread[threadsCount];

**int** expectedRegisteredStudentsCount = 0;

**for** (**int** i = 0; i < threadsCount; i++) {

RegistrationPortal registrationAccess = RegistrationPortal.*getRegistrationPortal*();

**int** studentsCount = Integer.*parseInt*(***SCANNER***.nextLine());

expectedRegisteredStudentsCount += studentsCount;

threads[i] = **new** Thread(**new** RegistrationRunnable(registrationAccess, studentsCount, String.*valueOf*(i + 1)));

}

**for** (**int** i = 0; i < threadsCount; i++) {

threads[i].start();

}

**for** (**int** i = 0; i < threadsCount; i++) {

threads[i].join();

}

RegistrationPortal registrationAccess = RegistrationPortal.*getRegistrationPortal*();

List<Student> registeredStudents = registrationAccess.getRegisteredStudents();

**if** (registeredStudents.size() != expectedRegisteredStudentsCount) {

System.***out***.println("Wrong Answer");

} **else** {

**boolean** correct = **true**;

**for** (Student student: registeredStudents) {

**if** (student == **null**) {

correct = **false**;

System.***out***.println("Wrong Answer");

} **else** {

String studentId = student.getId();

String studentName = student.getName();

**if** (studentId == **null** || studentName == **null**) {

correct = **false**;

System.***out***.println("Wrong Answer");

}

}

**if** (!correct) {

**break**;

}

}

**if** (correct) {

**class** StudentComparator **implements** Comparator<Student> {

@Override

**public** **int** compare(Student first, Student second) {

**int** firstStudentNumericId = Integer.*parseInt*(first.getId().split("-")[2]);

**int** secondStudentNumericId = Integer.*parseInt*(second.getId().split("-")[2]);

**if** (firstStudentNumericId == secondStudentNumericId) {

**int** firstStudentNumericIdPrefix = Integer.*parseInt*(first.getId().split("-")[1]);

**int** secondStudentNumericIdPrefix = Integer.*parseInt*(second.getId().split("-")[1]);

**return** firstStudentNumericIdPrefix - secondStudentNumericIdPrefix;

}

**return** firstStudentNumericId - secondStudentNumericId;

}

}

Collections.*sort*(registeredStudents, **new** StudentComparator());

System.***out***.println(registeredStudents.size());

**for** (Student student: registeredStudents) {

String studentId = student.getId();

String studentName = student.getName();

System.***out***.println(studentId + " " + studentName);

}

}

}

}

}

Attached test case results. All test case scenarios are passed.

