Project 1 : Datalonya Student Houses Simulator

CmpE 250, Data Structures and Algorithms, Fall 2021

Instructor: [redacted]
TA: [redacted]
SA: [redacted]

Due: 9/11/2021, 23:55 Sharp

1 Introduction

You are going to be implementing a Java program for simulating the distribution of student houses. You are going to be running this simulation by using the Java Collections Framework.

2 Details of the Project

The University of Datalonya offers houses for the students. All houses are for one student only. But they can be in various conditions: old, new, needs renovation, broken oven, broken shower, new kitchen, new bed etc. According to these conditions each house has a rating point between 0 and 10.

Students study at the university up to 8 semesters. If they are located at a house, they stay until graduation. But each student has his/her own criterion on house selection. This criterion is a threshold on the house's rating. For example, if the student wants a house with minimum rating 4, then he/she is not located at houses with rating 2 or 3 (any rating below 4) even if they are the only available houses.

New allocations are made at the beginning of each semester. House and student lists are checked for new allocations if possible.

Now you are enrolled at the University of Datalonya Dormitory Office. The current list of houses and students are transferred to you. From now on, you are responsible for the allocation of the houses.

You are supposed to arrange the lists of houses and students in collections of your choice and simulate the allocations until all students in the list graduate. In your simulation, check for matching houses and students at every new semester. The output of your simulation is the list of students who cannot stay at any house.

3 Input/Output Format

3.1 Input Format

The input will be given as a file argument. There are two types of input lines: house lines and student lines. The input lines will be in mixed order. You can assume that the input file is error free. You don't have check for data.

A typical line for a house is like this:

```
h <id> <duration> <rating>
```

There are 4 parts which are single-space separated in a line for a house. The first part, the letter h, is the house indicator. The second part, <id>, is the id of the house. The third part, <duration>, is the duration as the number of semesters that the house is full. If the duration is 3, then this house will be available after 3 semesters. The last part, <rating>, is the rating which shows the good or bad condition of the house.

A typical line for a student is like this:

```
s <id> <name> <duration> <rating>
```

There are 5 parts which are single-space separated in a line for a student. The first part, the letter s, is the student indicator. The second part, <id>, is the student id. The third part, <name>, is the name of the student. The fourth part, <duration>, is the duration as the number of semesters that the student will study at the University of Datalonya. For example, if the duration is 3, then this student will graduate after 3 semesters. Of course, max duration can be 8 for a student. The last part, <rating>, is the rating which shows the minimum rating criterion of the student to accept a house. For example, if the rating parameter for a student is 3.2, then this student can only stay at houses with rating equal to or greater than 3.2.

Your simulation program should read the input file and create appropriate collections for students and houses. Then process these collections until all students graduate. Your program should process houses and students in their id order. For example if there are two houses with ids 1 and 3 that are available at the moment, first the house with id 1 is allocated if possible. For students, if there are two students with ids 123 and 126 whose rating criterion holds for a house, the student with id 123 gets the house.

3.2 Output Format

Your program should end when the waiting student list is empty. The output of your program is the list of students that couldn't stay at any of the houses. You should create a .txt file and print one student name at a line. The output student list must be in ascending order of student id but only names will be printed in output file.

3.3 Java Project Outline

Your java project will be named **Project1**. Your entry class for the project will be named **project1main**. All your .java files will be under folder **Project1\src**. Your project should be compatible with Java 17. Your program will be compiled with below command:

```
javac Project1/src/*.java -d Project1/bin -target 17
```

The input and output files can be at any folder. Design your code in order to accept full path for file arguments. Your program will be run with below command:

```
java project1main <inputfile> <outputfile>
```

For example:
```
java project1main input.txt output.txt
java project1main somepath/input.txt someotherpath/output.txt
```

4 Examples

For the input data:

```
h 1 0 8
s 10 Ali 3 9
h 2 0 4
h 3 0 7
s 11 Melis 3 9
s 12 Ayse 5 4
h 4 6 9
s 13 Selim 5 3
h 5 5 5
```

The correct result is:

```
Ali
Melis
```

5 Grading

Grading of this project is based on the automatic compilation and run and the success of your code in test cases. If your code compiles and runs with no error, then you will get 10/100 of the project grade. The rest of your grade will be the sum of collected points from each test case. Each test case will have equal weight. Maximum project grade is 100.

6 Warnings

1. This is an individual project.
2. All source codes are checked automatically for similarity with other submissions and exercises from previous years. Make sure you write and submit your own code. Any sign of cheating will be penalized and you will get -50 points for the project and you will get F grade in case of recurrence in any other project.
3. There will be time limit on test cases, so it is important to write your code in an efficient manner.

4. You can add as many files as you can as long as they are in the src folder and your project can be compiled as above. But the entry point of your program should be project1main.java.

5. Make sure you document your code with necessary inline comments and use meaningful variable names. Do not over-comment, or make your variable names unnecessarily long. This is very important for partial grading.

7 Submission Details

You will zip up your project folder and submit on Moodle as a single .zip file. The name of the zip file is Cmpe250_Project1_<studentid>.zip No other type of submission will be accepted.