CSE21 : Project #2

You may work in pairs or by yourself. The expectation is that you work on the project outside of lab time but you may certainly use any extra time in lab this week. Your partner can be in any section, both of you will make your own submission while noting your collaborator/partner in the text box of the submission.

You have been hired by the UCMerced Dorm Life Council to prepare a program which measures compatibility between potential roommates. You are given a file with the list of students. For each student, you are given their name, gender and birthday. You will also have their preference for quiet time, music, reading and chatting. Using this information, you can then determine the compatibility between two people for rooming together, according to a formula developed by the Dorm Life Council. You will need to create four classes in order to do this project: Match.java, Student.java, Date.java and Preference.java. Below is how each class should function:

Student

Write a `Student` class that stores the following information about a person as instance variables:

- name: of type `String` for the name
- gender: of type char for the sex 'M' or 'F'
- birthDay: of type Date (see below) for the date of birth
- pref : of type Preference for their answers to four activities
- matched : of type Boolean to convey whether it's been matched

The Student class should have at least the following (you may add more as you deem fit):

- a constructor, which sets the instance variables to their appropriate input parameters (4 inputs)
- Accessor methods for each of the 5 instance variables
- Mutator method for matched to set it to true after a successful match with another student (note: should set on BOTH students)
- compare(Student st) method that returns the compatibility score between oneself and the Student input parameter, st.

- Compatibility Score Calculation (returned by compare)
    - Highest score is 100 and lowest is 0
    - Different genders make score 0 (only matching same gender students as roommates)
    - Formula : (40 – Preferences) + (60 – AgeDifference)
    - Preferences : Absolute differences in each of the 4 activities added together
    - AgeDifference : # of months between two birthdays with the maximum being 5 years (60 months)
    - Use Math.abs() method call to calculate absolute value e.g. Math.abs(age2 – age1)

Date

The class named Date should have the following instance variables:

- year: of type int in the range 1900-3000
- month: in the range 1 to 12 (January is 1)
- day: in the range 1 to 31 AND appropriate to the month. (can assume February is always 28 days)

The class Date should have at least the following methods:

- a constructor, which sets the instance variables to their appropriate input parameters (3 inputs)
- Accessor methods for each of the 3 instance variables
- compare(Date dt) method that returns the difference between oneself and the Date input parameter, dt.
    - Use dayOfYear() to calculate the difference between month/day then add 365*years to it. Finally divide the total number of days differential by 30 to approximate the # of months as a difference. If the difference is higher than 60 then should only return 60 as a max value.

```
public int dayOfYear() {
      int totalDays = 0;
      switch (month) {
            case 12: totalDays += 30;
            case 11: totalDays += 31;
            case 10: totalDays += 30;
            case 9 : totalDays += 31;
            case 8 : totalDays += 31;
            case 7 : totalDays += 30;
            case 6 : totalDays += 31;
            case 5 : totalDays += 30;
            case 4 : totalDays += 31;
            case 3 : totalDays += 28;
            case 2 : totalDays += 31;
        }
```

```
            totalDays += day;
            return totalDays;
        }
    }
```

Preference

Write a `Preference` class that records preferences for different types of activities:

- quiet Time
- music
- reading
- chatting

Each of these is of type `int` and must contain values in the range 0 to 10 (inclusive). 0 means the person hates the activity. 10 means the person loves the activity. The class Preference should have at least the following methods:

- a constructor, which sets the instance variables to their appropriate input parameters (4 inputs)
- Accessor methods for each of the 4 instance variables
- compare(Preference pref) method that returns the difference between oneself and the Preference input parameter, pref.
    - Sum of absolute value of the differences in the 4 activities.

Match

Write a `Match` class (with main()) that performs the following functions:

- Create an array of Students (max = 100)
- Read from a text file all the students information (tab delimited)
    - Name (String)
    - Gender (char)
    - Date (String: Month-Day-Year, - delimited)
    - Quiet Time (int 0-10)
    - Music (int 0-10)
    - Reading (int 0-10)
    - Chatting (int 0-10)
- For each line (in the text file)
    - Extract the information for each student
    - Create a Student pointed to be an entry in the array
    - Keep a count of actual number of students (less than 100)
- For each Student (after completely done reading)

- Check against every other student (assuming they are not matched already) their compatibility scores
- Find the Best Score and Best Match person
- Match the two roommates up and prints out the result

Note: Your algorithm for matching should work something like this:

```
Foreach student NOT currently matched
    Foreach rest of students NOT currently matched
        currentScore = studentA.compare(studentB)
        if the currentScore is better than MaxScore
            bestMatchStudent is student
            bestMatchScore is currentScore

    studentA is now Matched
    bestMatchStudent is now Matched
```

Also note that you have an array of students presumably. So studentA.compare(studentB) will actually look like this:

```
    if(!students[j].getMatched)    // student not matched already
        currentScore = students[i].compare(students[j]);
```

<u>Testing</u>

We have included two test files Students.txt and FullRoster.txt.

Expected Output: Students.txt

```
Abey matches with Melissa with the score 60
John matches with Jeff with the score 100
Craig has no matches.
```

Expected Output: FullRoster.txt

```
Abhay matches with Alexandra with the score 96
Adam matches with Alan with the score 100
Alexander matches with Alfred with the score 98
Analisa matches with Andrea with the score 99
Andrew matches with Colin with the score 90
Angelina matches with Dana Colleen with the score 93
Antonietta Vicky matches with Cynthia with the score 93
Arun Premnath matches with Kurt with the score 81
Chih matches with Dalila with the score 78
Daniel has no matches.
Kristal matches with Kristin Lunney with the score 91
Kristina matches with Lifen with the score 90
Lauren matches with Lien T. with the score 96
```

What to hand in

- Match.java, Student.java, Date.java and Preference.java
- Output of your program for Students.txt and FullRoster.txt (in output.log by copy and pasting from console)
- Partner’s name in the submission text box if any