Assignment 1

Due date

- cs542: Due by 11.59 PM EST on Tuesday, 8/30/2016.
- cs442: Due by 11.59 PM EST on Wednesday, 8/31/2016.

Submit your code as per the provided instructions.

Updates

- Format of output file has been changed
- The requirement for average preference score to be below 2.5 has been removed, as that was based on an earlier version of the assignment with different preference scores for course selection.
- 5% of the grade is dependent on the average preference_score that you generate.

Assignment Goal

A simple Java program.

Team Work

- No team work is allowed. Work individually. You cannot discuss the assignment with ANYONE other than the instructor and TA.

Programming Language

You are required to program using Java.

Compilation Method

- Compilation: Your code should compile on bingsuns or remote.cs.binghamton.edu with the following command:

  ```
  javac *.java
  ```

- Running the code: Your code should run on bingsuns or remote.cs.binghamton.edu with the following command: java Driver input.txt output.txt

Policy on sharing of code

- EVERY line of code that you submit in this assignment should be written by you. Do NOT show your code to any other student. Do not copy any code from any online source. Code for File I/O or String operations, if found online, should be clearly cited, and you cannot use more than 5 lines of such online code.
- Code downloaded in its entirety from an online repository of code (GitHub, BitBucket, etc.) and submitted as student's own work, even if citied, is considered plagiarism.
- Code snippets, for File I/O, if used from an online source should be cited by mentioning it in the README.txt and also in the documentation of every source file in which that code appears.
- Post to the listserv if you have any questions about the requirements. Do NOT post your code to the listserv asking for help with debugging.

Project Description

Assignment Goal: Develop a program, using Java, to assign courses to students based on their preferences.

- There are 4 courses (A, B, C, and D) being offered in the summer session. The capacity for each course is 10. The total number of students is 12. Each student is required to register for 3 courses. The student is asked to provide a preference for each of the courses. Top preference is specified as "1", while the lowest preference is specified as "4".
- In the file input.txt, you will be provided input in the following format:

```
Student_1 preference_for_A prefrence_for_B preference_for_C preference_for_D
Student_2 preference_for_A prefrence_for_B preference_for_C preference_for_D
...
Student_12 preference_for_A prefrence_for_B preference_for_C preference_for_D
```

- Use Java, to write a program to make the assignments, so that the output.txt file looks like the following:

```
Student_1 assigned_course_name assigned_course_name assigned_course_name total_preference_score
Student_2 assigned_course_name assigned_course_name assigned_course_name total_preference_score
...
Student_3 assigned_course_name assigned_course_name assigned_course_name total_preference_score

Average preference_score is: X.Y
```

- Replace X.Y in the output file with the actual average preference_score.
- If a student gets her first choice, the preference score is 1. If a student gets her second choice, the preference score is 2, and so on. Note that 20% of your grade is dependent on the quality of your course assignment (based on the combined preference scores).
- Class participation points will be given to the first 10 students who post interesting sample input files.
- A new requirement has been added to print the average_preference_score at the end of the output file.
- For this assignment's grading, we will collect the average preference_score for all submissions on blackboard and sort it to determine your rank in the class in terms of keeping the average preference_score low. 5% of your grade depends on the the average preference_score you generate.

Sample Input Files sent by students in this course

Please note that I have not verfied these input files.

- Input-1 Thanks, Lellapalli.
- Input-2 Thanks, Harsha Madhwani
- Input-3 Thanks, Christopher Fu.
- Input-4 Thanks, Tanmay Kale.
- Input-5 Thanks, Gaurav.
- Input-6 Thanks, Atharva
- Input-7 Thanks, Joshua Pratt
- Input-8 Thanks, Shashi Upadhyay
- Input-9 Thanks, Shiva Subramanian
- ~~Input-10 Thanks, Yash Divecha~~. [Reported to be identical to Input-1]
- Input-11 Thanks, Harshit Doshi.
- Input-12 Thanks, Omkar Nibandhe.
- Input-13 Thanks, Sandesh Nimhan.
- Input-14 Thanks, Triveni Banpela. [Same as input-15]
- Input-15 Thanks, Purva Mayakal. [Same as input-14]

Compiling and Running Java code

- Your README.txt file should have the following information:
  - instructions on how to compile the code
  - instructions on how to run the code
  - justification for the choice of data structures (in terms of time and/or space complexity).
- You should have the following directory structure (replace firstName_lastName with your name).

```
firstName_lastName_assign1/
firstName_lastName_assign1/Starting-with-java
firstName_lastName_assign1/Starting-with-java/src
firstName_lastName_assign1/Starting-with-java/src/Driver.java
[Other Java files you may need]
firstName_lastName_assign1/Starting-with-java/src/README.txt
```

Code Organization

- Your directory structure should be EXACTLY as given in the code template
    - Download the ANT based tarball here. Use the command on linux/unix: *tar -xvf firstName_lastName_assign2.tar.gz*.

Submission

- Read this file for general guidelines on how to prepare a README for your submission.
- Make sure all class files, object files (.o files), executables, and backup files are deleted before creating a zip or tarball. To create a tarball, you need to "tar" and then "gzip" your top level directory. Create a tarball of the directory firstName_lastName_assign1. We should be able to compile and execute your code using the commands listed above.
- Instructions to create a tarball
    - Make sure you are one level above the directory firstName_LastName_assign1.
    - tar -cvf firstName_LastName_assign1.tar firstName_LastName_assign1/
    - gzip firstName_LastName_assign1.tar
- Upload your assignment to Blackboard, assignment-1.

General Requirements

- Start early and avoid panic during the last couple of days.
- Separate out code appropriately into methods, one for each purpose.
- You should document your code. The comments should not exceed 72 coloums in width. Use javadoc style comments if you are coding in Java. Include javadoc style documentation. It is acceptable for this assignment to just have the return type described for each method's documentation.
- Do not use "import XYZ.*" in your code. Instead, import each required type individually.
- All objects, in Java, that may be needed for debugging purposes should have the "toString()" method defined. By default, just place a toString() in every class.
- Every class that has data members, should have corresponding accessors and mutators (unless the data member(s) is/are for use just within the method.).

Design Requirements

Late Submissions

- The policy for late submissions is that you will lose 10% of the grade for each day that your submission is delayed. There is NO difference in penalty for assignments that are submitted 1 second late or 23 hours late .

Grading Guidelines

Grading guidelines have been posted here.

mgovinda at binghamton dot edu
Back to Programming Design Patterns