ENSF 409 - Project

Due Date: April 9th at 5:59 PM via D2L dropbox
Client: University of Calgary Supply Chain Management
Teams: Must be completed in your assigned ENSF 409 team of 3 or 4 students.

Client Problem Definition

The University of Calgary aims to be a best-practice leader in environmentally and socially responsible supply chain management. Did you know that Supply Chain Management (SCM) at the University diverts filing cabinets, bookcases, and some furnishings to prevent usable materials from ending up in the landfill? The University targets re-use of certain furniture items but would like to expand the program to more complete re-use of surplus items.

Ali is a representative for Supply Chain Management of UCalgary. He needs an application to manage the flow of office furniture on campus. His main objective is to prevent usable materials from being wasted when they could potentially be used again in another item of furniture.

“I realized that we can save lots of money and resources by re-using certain furniture. It would be great if we find a way to re-use all the surplus items,” Ali said. “It is also much more cost effective than purchasing brand new items.”

You are being asked to design an application to calculate the cheapest combination of available inventory items that can be used to fill a specific order. For example, if a faculty needs one executive chair, the application should be able to calculate the most cost-effective way of assembling the chair from other used furniture items. Perhaps one used chair only has usable legs and cushion foam, but another has a usable seat and arms. Together they can be used to create a new chair based on the costs of usable and available components.

Faculties should be able to order the suggested items. If the needed components are available, an order form should be generated and the furniture items should be removed from inventory. An example order form is shown in the design specifications.

In addition to the financial benefits for the university, there are environmental benefits to reusing furniture: reuse prevents perfectly functional goods from ending up in the landfill, while conserving resources and materials required to manufacture brand new furniture. Ultimately, achieving this goal will help the environment while saving money, time and energy!

Design Specifications

SCM has provided a sample database that contains information about used furniture items currently available in inventory. The provided file is `inventory.sql`

Each furniture item has an ID number, a type, a price, a manufacturer, and a boolean Y/N value indicating whether particular components are still in usable condition. There is also a table containing general information about each furniture manufacturer. This table is kept for records only and does not need to be updated using the application.

Your application must do the following:

1. Take in user input for 1) a furniture category, 2) its type, and 3) the number of items requested.
2. Calculate and output the cheapest option for creating the requested pieces of furniture or specify if the request is not possible to fill.
    - Furniture items must be purchased as a whole. Single components cannot be purchased separately. A future expansion phase will add a buy-back option for individual unused components, so your code should be well-documented for future developers to use.
    - Each furniture item can only be built from items of the same type (e.g. mesh chair, desk lamp, etc.). Components are not interchangeable across types (including light bulbs).
    - Even used furniture costs money! The price of an item will depend on its number of usable components and their condition.
    - If there are multiple combinations for the cheapest price, select any one combination.
    - You are welcome to add furniture data to the database for testing purposes.
3. If the request can be filled, produce a formatted order form in a .txt format. An example order form file is provided.
4. When an order form is produced, the database should also be updated to specify that the selected items are no longer available in inventory.
5. If a request cannot be filled, the names of suggested manufacturers should be included in the output message. All possible manufacturers are included in each furniture table. Note that some of the manufacturers do not sell all furniture items.

The generated order form should include blank spaces where specific name and date information could be filled in later. The order form should include the original request and the list of desired components, as well as the total price. An example order form is provided in `orderform.txt`

Example use case scenarios based on `inventory.sql`:

Scenario #1:
User request: mesh chair, 1
Output: Purchase C9890 and C0942 for $150.

Output file generated:

```
Furniture Order Form

Faculty Name:
Contact:
Date:

Original Request: mesh chair, 1

Items Ordered
ID: C9890
ID: C0942

Total Price: $150
```

Scenario #2:
User request: executive chair, 2
Output: Order cannot be fulfilled based on current inventory. Suggested manufacturers are Office Furnishings, Chairs R Us, Furniture Goods, and Fine Office Supplies.

You may choose your own order of input arguments, output wording, formatting, etc.

Deliverables

Your team will create the following deliverables to present your solution.

All of the deliverables should be inside a single .zip folder labelled with your ENSF 409 assigned group number. Only one team member needs to upload the solution to D2L. You must include all of your team members' names on each deliverable (i.e. in the code documentation, unit tests, etc.). Please also include all member first and last names in the provided text box when submitting on D2L.

UML diagram (15%)
Code implementation (40%)
Documentation (10%)
Unit tests (25%)
5-minute video demonstration (10%)

UML diagram: You must submit a UML diagram that accurately represents your final code design. The diagram must be legible and follow proper formatting conventions. You do not need to represent exception handling in your diagram.

Code implementation: All of your solution's .java files should be included. You should use the edu.ucalgary.ensf409 package. Each public class must be placed in its own file.

Your code should be developed using GitHub. Each team member should fork the repository and use pull requests, as this will confirm code contributions in case of team disputes. Suggested workflow:

- Have one person create the project on GitHub.
- Each person, including the one who created the project, should fork the repository and work on their fork, sending pull requests to the original project. While we will not enforce this model of working, do note that it is in your interests to follow it, as it ensures that (1) you have a copy of the code for future reference or inclusion in portfolios (in the event of the project owner removing you or deleting the project), and (2) there is a record of your actual contributions in case there is a subsequent dispute about the contributions made by each group member.
- If you pair program, take turns making commits so that there is a record of your contributions.
- Your final code will be submitted via D2L dropbox. In the case of any disputes about contributions, we will request read access to your GitHub project.

Documentation: All of your code should be fully documented and commented. Formatting and naming conventions should follow the standards used in ENSF 409. Your main should include instructions for running and using your code.

Unit tests: You must include unit tests that demonstrate how your solution meets the design specifications. Consider possible boundary cases and exception handling. Your code will be run against your own tests, though values may be changed to ensure that solutions are not hard-coded to specific scenarios.

Video demonstration: Use Zoom or another tool of your choosing to record a short demonstration of your application (5 minutes or less!). All team members should be part of the demonstration. Marks

may also be deducted for videos of excessive length. Videos should have clear audio and cameras should be on. This is your opportunity to demonstrate how your application works and why it fulfills the requirements.

Grading Rubric

Each deliverable will be graded using a rubric. You must satisfy all of the criteria in a particular category to earn that category's minimum grade. The grading criteria can be found in the accompanying rubric handout.

Team Assessment

Following the project submission, you will have the opportunity to reflect on the contributions of each team member. You will allocate a percentage of work to each member. For example, if all members contributed equally, you would allocate 25% to each person. If one student took on more responsibilities compared to the others, the allocation might be adjusted to 40%, 20%, 20%, 20%. These allocations may be used to adjust your individual grade on the project in case of uneven contributions. Any discrepancies between allocations will be flagged for the TAs and instructors to investigate further.

Hackathon Judging Presentation (optional)

This is not part of the ENSF 409 requirements but is optional for those participating in the hackathon. If you submit any of your draft deliverables by 11 am on Sunday, March 28th, you will receive feedback from the judges.

Before the hackathon submission deadline, each group should provide a brief demo video presentation as a pitch for your final design or summarizing your current progress. It should outline how your solution meets or exceeds the expectations and why it is the best choice. The judges will consider the criteria set out in the rubric. The top five teams will be picked to present to the judges as a 15-minute presentation. This will be worth 30% of your overall hackathon score and will be judged based on clarity, confidence, organization, visual aesthetic, and your ability to answer questions.