Design, Simulation and Programming Assignment

Core Java Programming Language

Trainer
Amarjit Singh

Design, Simulation and Programming Assignment

PROJECT OBJECTIVES

MAJOR OBJECTIVES

Identify Types and Define Types
Design Role and Responsibilities
Design Towards Single Responsibility
Design Towards Loose Coupling
Design For Changeability and Future Requirements
Evaluate Various Design Choices and Reason/Prove It
Define Design Objectives and Approaches

Appreciating Design Thinking
- Drive Design By Convention vs. Configurations
  - Design By Contracts
- Drive Design By Definitions
  - Design with Types
  - Design For Localisation/Encapsulation
  - Design For Invariance/Polymorphism
  - Design For Evolution/Extension and Change

Appreciating OOAD in Java
Representing Design in UML Diagrams
Writing Design Document
Exploring Design Strategies, Discussing and Reasoning Design

PROJECT DELIVERABLES

PROJECT DELIVERABLES

Solve Design Problems

Design Solution In Multi Stages Tracked By Commits

Submit Your Code and Documents Design in GitHub

Create Repository "FKPayrollDesign" with README.md File
Project Directory Structure Must Be As Follows

- FKPayrollDesign
  - SourceCode
    - *.java
  - Documentation
    - DesignDocument-v0.1.pdf
    - RoughSheets.pdf

README.md

Document Following Things In DesignDocument-v0.1.pdf

Required Diagrams

Sequence Diagram in UML
Type/Classes/Interfaces etc Design in UML

Relationship/Dependency Diagram
Annotate Relationship/Dependency Between Types
Coupling Between Objects
Other Dependencies

Required Documentation

Design Objectives and Approaches
Design Role and Responsibilities
Design Choices and Your Preferences with Reasoning
Future Design Improvements

Contradictions Discovered To Existing Design
Design Challenges, Experiments and Alternative Design
Design Improvements over Existing Design and Reason It!

PROJECT DEFINITION

PROJECT PAYROLL DESIGN

- Some employees work by the hour. They are paid an hourly rate that is one of the fields in their employee record. They submit daily time cards that record the date and the number of hours worked. If they work more than 8 hours per day, they are paid 1.5 times their normal rate for those extra hours. They are paid every Friday.

- Some employees are paid a flat salary. They are paid on the last working day of the month. Their monthly salary is one of the fields in their employee record.

- Some of the salaried employees are also paid a commission based on their sales. They submit sales receipts that record the date and the amount of the sale. Their commission rate is a field in their employee record. They are paid every other Friday.

- Employees can select their method of payment. They may have their paychecks mailed to the postal address of their choice; they may have their paychecks held for pickup by the paymaster; or they can request that their paychecks be directly deposited into the bank account of their choice.

- Some employees belong to the employee union. Their employee record has a field for the weekly dues rate for union. Their dues must be deducted from their pay. Also, the union may assess service charges e.g. membership fee, festival fees etc.

against individual union members from time to time. These service charges are submitted by the union on a weekly basis and must be deducted from the appropriate employee's next pay amount.

- The payroll application will run once each working day and pay the appropriate employees on that day. The system will be told to what date the employees are to be paid, so it will generate payments for records from the last time the employee was paid up to the specified date.

Following Use Cases Must To Be Implemented

1. Add a new employee
2. Delete an employee
3. Post a time card
4. Post a sales receipt
5. Post a union membership, service charge etc.
6. Change employee details (e.g., hourly rate, dues rate, membership fee etc)
7. Run the payroll for today

PROJECT GUIDELINES

Git and GitHub Guidelines

- *Use ONLY Personal Email ID and Your Full Name For Commits*
- *Repositories Must Be Public and Follow Naming Convention*
- *Follow Project Directory Structure Guidelines*
- *Please NOTE Repository Names Are Case-Sensitive*

There Must Be Three Branches In Each Repository

Master Branch

Nothing To Commit Here After First Commit

Dev Branch

All Development In Development Branch
All Time Must Be ***Working Code In Dev Branch***
Means Every Commit Must Be Working/Tested Code

Feature Branches

Name Feature Branch Clearly
Experimental and Final Feature Branches
Merge Final Feature Branch In Development Branch
Don't Delete Your Experimental/Final Feature Branches
After Merging To Dev Branch
Push Your Feature Branches Also To Repository

Document Each Commit Clearly and Completely

Commit Comments Must Contain

Feature Number and Feature Title
Comments Must Describe What You Solved
And Design Choices Implemented
and TODO List

Development and Feature Branches Naming Conventions

Project Requirements Might Be Given In Phases

Branches Naming Convention

Development Branch Name: *Development*

Feature Branch Names:

Feature Request Are Numbered

e.g.

Feature01-AddingNewEmployee

Feature02-DeletingEmployee etc.

Tag Versions Committed To Development Branch

Follow Coding Standards and Guidelines

Submit Your Rough Sheets

Number It and Add Your Name
