FORMATIVE ASSESSMENT 3

Project

Beginner Java

JD521

2024

Creative & Technology Universitas

training solutions

Table of Contents

Please complete the declaration of authenticity below for all assignments:

DECLARATION OF AUTHENTICITY

I ____

(FULL NAME)

hereby declare that the contents of this assignment are entirely my own work with the exception of the following elements: (List the elements of work in this project that were not self-generated as well as who the originator of the element is)

Element	Originator

Signature: ______________________________

Date: _____________

Software Design & Development

Module: Beginner Java
Module Code: JD521
NQF: 5
Credits: 22
Assessment type: Formative Assessment
Mark allocation: 100
Hand out date: 26 April 2024
Hand in date: 10 May 2024

Instructions

Read each question carefully and consider the mark allocation prior to answering.

- Ensure you answer all questions.
- The use of AI tools is strictly prohibited in completing this assessment.
- For final submission**,** submit a compressed file that includes: **The Declaration of Authenticity, Source Code, and ReadMe file.**

Unit Standards:

US Type	US ID	Description	Level	Credits
Core	115359	Demonstrate an understanding of the handling of error in a computer programming environment	NQF-4	2
Core	115365	Apply the principles of designing computer system inputs and outputs	NQF-5	7
Core	115362	Manage software development source files using appropriate tools	NQF-5	5
Core	115367	Demonstrate logical problem solving and error detection techniques	NQF-5	8

Assessment outcomes:

- Using date, math and time utilities
- Understanding threads
- Creating threads
- Understanding streams.

Project-based Questions (100)

Instructions: Read the scenario and answer the following questions:

Advanced Library Management System

The library manager loved the implementation of the previous project and has requested you to improve on this project to accommodate additional functions and features.

Enhance the existing Library Management System to include features that utilize date, math, and time utilities, create and manage threads for background tasks, and use streams for data input/output. This upgraded system will introduce functionalities such as due dates for borrowed books, overdue fines calculation, parallel processing for tasks like notifications, and persistence of library data through file operations.

New Objectives:

- Using Date, Math, and Time Utilities: Implement functionality to track due dates for borrowed books and calculate overdue fines using date and time utilities, incorporating mathematical calculations for fines.
- Creating Threads: Utilize multi-threading to perform background tasks such as sending reminders for due books and processing overdue fines in parallel with the main application flow.
- Using Streams: Implement data persistence by saving and loading library data (books, members, transactions) using file streams, ensuring data is maintained between sessions.

Upon completing this task, the library manager promised to sign off your WIL logbook.

Source: Mashile T., (2024)

Project Tasks

Task 1: Implement Due Dates and Overdue Fines

- Extend the Book and Member classes to include tracking of due dates for borrowed books. Each book borrowed should have a due date set based on the current date plus a predetermined loan period (e.g., 14 days).
- Implement a method to calculate overdue fines for each day a book is overdue. The calculation could be a simple daily rate or more complex, involving varying rates over time.

Task 2: Background Notifications and Fine Processing

- Create a background task using threads that periodically examines for due or overdue books and automatically updates fines for overdue books.
- Implement another thread that sends out notifications to members with due or overdue books. This could simulate sending emails or generating a report.

Task 3: Persistence Using File Streams

- Save library data (including books, members, and transaction logs) to files using streams, ensuring data is not lost when the application closes.
- Load library data from files when the application starts, allowing for seamless continuation from the last state.

Task 4: User Interface Improvements

- Enhance the text-based menu system to include options for checking due dates, viewing fines, and managing notifications.
- Ensure that the system provides clear feedback on background operations, such as updating fines and sending notifications.

Additional Requirements:

- The application should handle concurrent operations safely, ensuring data integrity when multiple threads access and modify shared resources.
- Proper error handling for file operations, including handling scenarios where data files are missing, corrupted, or inaccessible.
- Implement thread-safe collections or use synchronization mechanisms for shared data access between threads.

Submission Guidelines:

- Source code files, including any new classes or modified functionalities.
- A ReadMe text file detailing the implementation of the new features, focusing on the use of date, math, and time utilities, threading, and streams.
- A user manual that Includes screenshots demonstrating the new functionalities in action, particularly showing how the system handles concurrency and persistence.

End of project **[Total = 100 Marks]**

End of paper

Formative Assessment 3 Marking Rubric

Penalties: Points may be deducted for significant shortcomings in any evaluation area, such as frequent program crashes, major bugs, or failure to compile.

Assessment Criteria			
Student name/number:			
Criteria	**Mark allocation**		
	15 – 11	**10 – 6**	**5 – 0**
1. Using Date, Math, and Time Utilities (25 Marks)			
Implementation of Due Dates (10 Marks): Assess the correct implementation and tracking of due dates for borrowed books, including the accurate calculation of due dates based on the loan period.			
Overdue Fines Calculation (15 Marks): Evaluate the logic and accuracy of the overdue fines calculation, ensuring that mathematical operations are correctly applied and that fines increase appropriately over time.			
2. Creating Threads (25 Marks)			
Background Task Implementation (15 Marks): Assess the effective use of threads for background tasks, such as notifications for due books and fine calculations. This includes evaluating how threads are created, managed, and terminated.			
Concurrency and Synchronization (10 Marks): Evaluate how the project handles concurrent access to shared resources, ensuring data integrity through appropriate synchronization mechanisms or thread-safe collections.			
3. Using Streams (25 Marks)			
Data Persistence (15 Marks): Assess the implementation of file streams for saving and loading library data, including books, members, and transactions. This includes evaluating the data format used, handling of file exceptions, and ensuring data integrity.			
Efficiency and Error Handling (10 Marks): Evaluate the efficiency of the stream operations and the robustness of error handling, ensuring the system can gracefully recover from errors such as missing or corrupted files.			
4. Code Quality and Documentation (15 Marks)			
Readability and Structure (5 Marks): The code should be well-organized, with meaningful variable names, consistent indentation, and clear separation of functionalities into methods/classes.			
Comments and Documentation (5 Marks): Inline comments and documentation should clearly explain the purpose and logic of the code, making it easy for another developer to understand and maintain.			
Design and Scalability (5 Marks): Evaluate the overall design of the system, including the use of object-oriented principles and how easily new features (like additional background tasks) could be added.			
5. User Interface and Experience (10 Marks)			
Usability (5 Marks): The text-based menu system should be intuitive, with clear instructions on how to perform each action, especially the new functionalities related to due dates, fines, and notifications.			
Feedback on Background Operations (5 Marks): The application should provide clear feedback to the user about the outcome of background operations, such as successful fine calculations and notifications sent.			
TOTAL			**/100**