University of South-Eastern Norway

OBJ2100 (2025 Vår) Obligatorisk Oppgave 1

Version 1.0

Release Date: 6 February 2025

1. Overview

This is an individual assignment where you will apply key programming concepts learned in class til now. There are **five available tasks** (see Annex 1), and each student must **choose one to complete**.

1.2 Assignment Expectations

Your final product does not need to be fancy or fully polished, but it must be functional and demonstrate the key concepts covered in class. The focus is on applying what you have learned rather than creating a perfect product.

- There are no detailed requirements or specifications for each task—you are the product owner. Use your imagination to define the features and user experience. Think critically about what makes your product valuable and practical while working within the given time constraints.
- Strive to implement the core lessons (Animation, Exception Handling, Text I/O, Binary I/O, and Object Serialization) in meaningful ways.
- If you choose not to apply certain concepts, you must clearly justify your decision in the documentation (for example, time constraints or the product limitations)

The goal is not perfection but demonstrating thoughtful application of course concepts.

1.2 Workload Expectation

This assignment is designed to take approximately **15-20 hours**, including planning, coding, debugging, and documentation. Allocate your time wisely to ensure both code and documentation are completed effectively.

2. Key Guidelines

2.1 Think Before You Code

- Do not jump directly into coding—use the documentation template (See Annex 2) to plan your approach first.
- Outline your program structure, class design, and key features before implementation.
- Describe (“THINK”) how you will apply animation, exception handling, and Java I/O stream in your chosen task.

2.2 Write Clean and Readable Code

- Use consistent indentation, meaningful variable names, and comments where necessary.
- Follow best practices for error handling, modularization, and object-oriented design.

- Include Javadoc comments for all classes and methods to improve code readability.
- Ensure your code is easy to read and maintain.

2.3 Collaboration Policy

- Discussions with classmates are encouraged, but sharing or exchanging code is strictly prohibited.
- You may help each other understand concepts, but your implementation must be your own.

2.4 Time Management

- Documentation is a significant part of this assignment—allocate enough time for writing.
- Plan your work schedule to ensure both coding and documentation are completed on time.

2.5 Responsible Use of AI

- AI tools (e.g., ChatGPT, DeepSeek, Copilot) may be used, but must be declared.
- In your AI Usage & Code Authenticity Statement, explain: What AI-generated suggestions you used. How you modified the AI-generated code. Why you made those modifications.

4. Submission

Each student must submit **two files** in Canvas:

1. **A ZIP file** containing all source code and any necessary assets.
    - ZIP File Name: `StudentID_TaskName.zip` (e.g., `23455_CarRacingGame.zip`)
    - Your **ZIP must include:**
        1. All Java source files
        2. Any required assets (images, configuration files, storage files, etc.)
        3. A `README.md` file with instructions on how to run the program.
2. **A PDF document** following the provided documentation template (Annex 2), which explains your thought process, design decisions, and implementation details.
    - PDF File Name: As as the ZIP file name.

4. Submission Deadline: 23 February (Sun) kl. 23:59

If you are unable to finish the work by the deadline, please notify me before it expires.

Annex 1. Task Options

- **Option 1: Car Racing Game**: Develop a simple 2D car racing game where players control a car to avoid obstacles. Track and record the player's game time. Display a leaderboard showcasing the top five longest survival times at the start of the game.
- **Option 2: Library Management System**: Design a Library Management System that allows users to add, delete, and search for books. Books can be borrowed and returned with a graphical interface.
- **Option 3: Digital Diary**: Create a simple Digital Diary application where users can write daily entries, save them to a file, and view previous entries with a graphical interface.
- **Option 4: Simple Quiz Application**: Create a Quiz Application where (1) administrator can enter Q&A in advance, and (2) users answer multiple-choice questions via a graphical interface. The quiz progress and results can be saved and retrieved later.
- **Option 5: Personal Finance Tracker**: Develop a personal finance tracker that allows users to record expenses and incomes with categories (e.g., Food, Transportation, Entertainment). Use visuals to present the daily summary.

Annex 2. Documentation Template

Task Name:

Student ID:

Student Name:

1. **Production Description**

1.1 Overview

(Provide a short description of your product. What does it do? Who is the target user? What problem does it solve?)

1.2 Features

(List and briefly explain the main features of your product.)

1.3 Application of Key Lessons

(Describe how you applied the following concepts in your program. Provide specific details and explain why you chose certain implementations.)

1.3.1 Animation (JavaFX)

(How did you implement animation? Which classes (e.g., Timeline, Transition) were used?)

1.3.2 Exception Handling

(What errors could occur, and how did you handle them? Did you create custom exceptions?)

1.3.3 Text I/O

(How does your program read/write text files? Where is this used in your program?)

1.3.4 Binary I/O (including Object Serialization)

(What objects/data are stored in binary form? Why did you choose binary I/O instead of text I/O?)

2. Challenges & Problem-Solving Approach

(Describe at least two major challenges you encountered during development and how you solved them. Be specific!)

3. Design and Implementation Details

3.1 Program Structure

3.1.1 Class Diagram / Flowchart

(*Attach a system diagram, flowchart, or simple sketches of how the program is structured.)*

3.1.2 Explanation of the overall program structure

(Explain how your program is structured. What are the main Java classes? What are their responsibilities? How do they interact?)

3.2 Key Design Decisions

(List the major design choices you made and justify them.)

Component	Design Choice	Justification
Game status	Binary Serialization	
System Log	Text File I/O with buffered mechanism	
Exception Handling	Custom Exception	

3.3 Implementation of Key Features

(Describe how you implemented the core functionalities in your program.)

3.3.1 Animation

3.3.2 Exception Handling

3.3.3 Text I/O

3.3.4 Binary I/O

4. Reflection & Improvements

4.1 Lessons Learned

(What did you learn from this assignment? What mistakes did you make? What would you do differently next time?)

4.2 Future Improvements

(What are the limitations of your program? What additional features would you add if given more time?)

5. AI Usage & Code Authenticity Statement

Be transparent about AI usage and modifications.

5.1 Did you use any AI tools (e.g., ChatGPT, DeepSeek, Copilot)?

___ No AI tools were used.

___ Yes, AI was used (explain how below).

5.2 AI Code Modifications

(If AI was used, describe how you modified and improved the AI-generated content.)

AI-Generated Code	Modifications Made	Reason for Changes
Suggested an animation loop	Adjusted speed and timing	Prevent animation lag
Provided basic exception handling	Added custom error messages	Improved debugging

6. Screen Shots for Program Execution

(Provide screenshots that demonstrate key features and functionality of your program.)