AP Lesson 4.3: Integrate and Create

AP PROBLEM **4.3.1**

Integrate and Create

INTRODUCTION

For the final problem, you will integrate, or combine, various features from different projects to create something new. You will use the software development cycle you have been practicing throughout the course and will deliver a final presentation of your work. Finally, you get to choose which type of project to complete:

- Make an App as described in *Lesson 4.3 (Android Focus)* using your Android development skills. Please refer to **Problem 4.3.1 Make An App**. Note that you will not be implementing the LibGDX version of the app, rather you will implement the Android version.
- Continue to enhance the Elevens program. As described in the procedure steps below, you will create a "Thirteens" version of the game, and then integrate your game of Concentration from AP Lesson 4.1.

Materials

- Computer with BlueJ IDE
- Tools with which to create visual representations of your code
- Tools for creating and using a sprint task list and product backlog
- Presentation software

RESOURCES

AP Problem 4.3.1 Requirements
Resources available online

AP Problem 4.3.1 Rubric
Resources available online

Elevens Lab Student Guide (complete)
Resources available online

Image: blank.png
Resources available online

Procedure

Part I: Prepare

Before you begin implementing a new game with the Elevens framework, you need to become more familiar with its various classes and functionality. In doing so, you will be better prepared to make the changes and enhancements required of you.

1. Reference the resource **Elevens Lab Student Guide (complete)** and complete **Activity 10**: **ThirteensBoard**, including all exercises.

Part II: Plan

2. Read through the **AP Problem 4.3.1 Requirements** and then read this entire problem definition. Know all requirements before you begin any work.
3. Create a problem statement for your project.
4. Define a minimum set of requirements—what you want the game to do.
5. Based on the problem statement and the requirements, create a user persona. Write specific ideas and include motivations and personalities.
6. Identify a group of people that fits your user persona. They will become your stakeholder group. Plan for a meeting with them.
7. Document how you prepare for the meeting, specifically how you will introduce and discuss your project and game. Your teacher may provide sample prompts for talking with your stakeholder group.
8. Meet with your stakeholder group. Together you will review the problem statement.
   a. Define the "must-have" functions; what the app *needs* to do. Prioritize these functions. Be prepared to modify your requirements and feature set based on this input.
   b. Describe the major characteristics, such as the features of the game. Take care not to delve too deeply into details; this should be a high-level talk, not how buttons are shaped or comparing different shades of favorite colors.
   c. Create a "wish list" of functions that go beyond this first phase of development.
   d. At the end of the meeting, thank them for their time!
9. With the input from your meeting, adjust your user persona, if necessary.

Part II: Design

With the planning phase complete and with stakeholder feedback in mind, design the game's features.

10 Create the user stories for your project, create the sprint tasks and backlogs.

11 Design classes, methods, and/or instance fields you need for your game. UML diagrams may help with this task. Keep the following in mind:

- The width of the playing board must be 4 cards (not 5 as in the Elevens game) with as many rows as you want.
- Cards of the same rank and color are considered a match.
- When cards are matched, a blank image should appear (please refer to the resource file blank.png).
- Each card will have different states, such as face up, face down, and matched.
- You may not change the `Board` and `Deck` classes.

Part III: Develop and Test

You are now ready to begin implementation. All features should be seamlessly and logically arranged. Integrate them so they fit naturally and authentically into your design.

12 Create a backup copy of your solution from Part 1 so you can refer back to original and unedited code as you create the new features for Concentration.

13 Implement the new features of the Concentration game, modifying the user interface as necessary. Some helpful hints as you develop:

- Remove all unnecessary classes from your project and/or rename when appropriate.
- Learn the purpose of the `cards` folder.
- Do not delete any overridden method; modify them as necessary.
- Test with a very small board and a small deck of cards.
- Turn on the `I_AM_DEBUGGING` flag and add your own debugging statements.
- Use the debugging tool in BlueJ to step through code and learn functionality.

14 Provide/modify Javadocs and other in-line documentation.

15 Test your development stages regularly.

16 If directed by your teacher, share your project with another team during development and test another team's project. Provide constructive feedback and respond to feedback in a positive way.

Part IV: Deliver and Present

Prepare and deliver two presentations—one to your stakeholder group and one to the class. The stakeholder group presentation will be a subset of the classroom presentation.

17 Prepare and deliver a full presentation to the class:

a. Explain the team's decision to do this particular projectand share your problem statement.

b. Describe the user persona and how your game meets the needs stated in your problem statement.

c. Describe the current state of the product with respect to the problem statement, product backlog, and sprint task list.

d. Demonstrate the features of the game.

e. Explain the team's contributions of original code to the product solution, as well the integration of code from previous projects.

f. Reflect on how the game represents what you have learned in this course.

18 Prepare an "executive summary" presentation to deliver to your stakeholder group:

a. Describe the current state of the product with respect to the problem statement, product backlog, and sprint task list.

b. Demonstrate the features of the game that the team created.

c. Collect feedback from the audience as they use they play themselves.

AP Problem 4.3.1 Integrate and Create - Requirements

Problem 4.3.1 Integrate and Create requires documentation, specific functionality, and a team presentation.

Documentation Requirements

1. Problem statement
2. User persona including meeting preparation, stakeholder needs, ideas, and feedback
3. Product backlog and sprint task list
4. Javadocs for sources of new features
5. Product source code

Functionality Requirements

1. The user interface has all of the elements required for a game of Concentration.
2. The user interface contains *only* the elements required for a game of Concentration.
3. Code modifications and enhancements are made without code duplication, using efficient coding styles, and with the best object oriented principles in mind.
4. Implement the specific requirements described in **AP Problem 4.3.1 Integrate and Create**. Use the **AP Problem 4.3.1 Rubric** which describe the rubrics specific to this project.

Presentation Requirements

Develop one presentation and prepare two versions of it, one short version and one long version. For presentation details, see **AP Problem 4.3.1 Integrate and Create**.