Final Group Project Overview

What Will Be Accomplished

- Develop, test, and execute a graphic application for simulations using Java.
  - Create a Java application.
  - Given a set of events, choose the appropriate programming actions.
  - Work with Java in a way that demonstrates understanding of the principles behind the programming language.
- Demonstrate understanding of the principles of object-oriented programming including classes and inheritance.
- Integrate an understanding of XML and apply this understanding to Java.

Deliverables

- **.java** files as requested below
- XML files

Basic Requirements

The final project game: The scenario is based on the Pennsylvania State University system. A character (a student, a football player, a famous Penn State icon, etc.) will be walking around the state of Pennsylvania and will enter five campuses, one at a time. At each campus, the character will face challenges such as answering questions, solving a puzzle, or playing a game. The character will collect points in each location. After five visits, the game will end. Then a final screen will be shown, displaying the character's rankings according to his/her performance.

Detailed Requirements

The Map

The character will move on a map. It might be a sketch built with graphics (panels, buttons, etc.,) and a picture background.

The application has to have more than five campuses so the user has a choice of where to go. Find a creative way to also represent World Campus. It might be a special key or place for the character to enter World Campus. In order to enter the other campuses, the character has to be moved over the campus position on the screen.

The Campuses

As the character enters a campus, a new panel should show up. The panel will have one or more questions, or a challenge (a puzzle or a game). When the answer is given or the challenge is over, the character should be put back on the main map screen where he/she was last. You should keep score of the character's performance to use it in the final ranking.

The Character

The character has to be picked from a list of three choices. Each choice should make the character different; for instance, it might move faster, it might, more or less, be able to answer the questions, or have more or less speed or strength when he/she plays the games at each campus.

The Character's Movement

The character will be moved using the keyboard. When the character gets close or on top of the campus, the campus panel should appear and the campus game should be ready to be played.

The character should have an icon representing the user choice.

The Theme

The user might choose from three different options for a game theme. For instance, the theme might be Penn State football and the questions and games will be themed around that. Also, the theme might be Math and the questions and puzzles will regard Math. The theme options are your choice.

Game Over

When the character has entered and interacted with five campuses, the game is over and a final screen with the character's ranking should be displayed.

XML

The text of the questions and of the final ranking should be kept in a file and read when needed. You have to use what you learned in the XML lesson to read and write the file. XML is the required format, and it should be used as explained in the lesson.

The Timer

- The time (duration of game play) should be kept, displayed at all times (you might use a progress bar, for instance), and be part of the final ranking. You might choose to give a time limit and require players to enter five campuses before the time limit is reached. A final screen with the score, time, and ranking should be shown.

Class structure

All the projects:

- Have to start the basic class with the main method. It will create a JFrame, specified below.

- Will have their own JFrame extending from Java's JFrame. This JFrame will create a JPanel specified below.
- Will have their own JPanel extending from Java's JPanel.

Java Code

It has to be based on the examples given in class and in the textbook. If you intend to use advanced code, clear it with the instructor first.

Suggestions

Use inheritance where appropriate. Use parameters to reuse code. For instance, the Questions panel might use one panel class that receives different parameters (number of questions, name of the file with the questions, theme). Use the course examples and your own labs to get started. Start simple: start with something you might be more comfortable with. It might be the main screen, the character movement, or creating one questions/answers panel. Try to follow the lesson's examples and standards.

Grading Rubric

Criteria	**Ratings**		
Java Code It has to be based on the examples given in class. If you intend to use advanced code, clear with the instructor first.	Java code comes from the examples and all group members understand ALL of the code *10 Points*	One or more members of the group are not able to explain the code *5 Points*	SEE NOTE BELOW *-50 Points*
The Map The character will move on a map.	Has a functional map where the character can move and enter campuses *10 Points*	Has a map but it is not functional *5 Points*	Does not have a map *0 Points*
Campus Panels	The campus panels are complex and different enough *30 Points*	Campus panels are too simple *15 Points*	Campus screens are just like one of the Labs developed during the semester *5 Points*
Character choices (options that influence how the game is played); It has to have three options.	Three or more options *10 Points*	Two options *5 Points*	One or No options for setting up the game *0 Points*
Character movement The character will be moved using the keyboard	The character is able to move around and enter the campuses *10 Points*	The character movement is unstable and has problems *5 Points*	The character does not move as required *0 Points*

Theme Choices	Has theme choices and the chosen theme is fully integrated within the game *10 Points*	Has theme choices but they are not integrated in the game *5 Points*	No theme choices *0 Points*
Game Over	A panel with a message "Game Over" appears and shows the player's ranking *10 Points*	Has some errors. The panel does not appear or is missing requirements *5 Points*	No "Game Over" panel *0 Points*
XML Has to use the XML_240 java class. The game has to import or export xml files. Example: reading the questions or the text from the final ratings from an XML file.	Uses XML as required (text for questions, choices, and ratings or another creative use) and uses the XML_240 class *30 Points*	Uses XML in a very simple way *10 Points*	XML in the game does not work at all or does not use the XML_240 class *0 Points*
The Timer Game has to have a timer and it should be based on Timer lesson material and class examples	Has a Timer as used in the Timer lesson *10 Points*	Does not have a timer or has different kinds of timers *0 Points*	
Bugs	Works fine, no errors *10 Points*	Has some errors. Some parts of the game can be played, others can’t *5 Points*	Has so many errors that the game cannot even be played *0 Points*
Class Structure app, JFrame and JPanel	According to specs *5 Points*	Doesn't follow specs *0 Points*	
Coding Style 1 It has to start with the three java files used in all Graphics Labs • app.java • myJFrame.java • myJPanel.java	The application starts with app.java that creates a myJFrame that creates a myJPanel *10 Points*	The application does not have the three required files *0 Points*	
Coding Style 2 Static or final variables are not allowed	Does not use static or final variables *10 Points*	Uses static or final variables *0 Points*	

NOTE*: Unexplainable Java code will result in a deduction of 50 points. To reiterate, 'strange' java code that you do not likely understand, or if I find your code or parts of it on the web, you will loose 50 points and will likely be reported to the Associate Dean for cheating.*

The instructor will interact with groups and will answer direct questions. Students are expected to think, explore issues, and do research.

Groups do better work than individuals and must pull together. Every member of a group is responsible for the group product. Each member must be involved with all of the activities. Group activity will be closely monitored and necessary changes to groups will happen where appropriate.

Groups have their own unique group space right within this course environment. In groups (accessed through people), you have the ability see who is in your group, post announcements to all members at once, start discussions, add files, and initiate conference meetings with group members.