Final Project Design Questions

Design Check Signups: Until 12/2 at 11:59PM

Design Checks: December 3 –December 6

Help Session: November 20th, 2:30 PM (in class)

Before writing even one line of code, you must design your final project. This process will require you to break down and outline your program into classes, design your data structure(s), clarify the major functionality of your program, and pseudo-code important methods. After designing your program, you will find that writing the program is a much simpler process.

If you do not sign up for your design check on time or you do not show up for your design check then you will get a 0 for your design check grade. If you cannot make any of the available times you must let us know ahead of time. Exceptions will be made for illness and other extenuating circumstances. "I forgot" is not an excuse. Make sure to sign up early to ensure that you get the project and time that you want.

Note: If you are doing an Indy project, you must fulfill the requirements for the Design Check stated here as well as those stated on the Indy handout.

How To Attack This Assignment

You would be foolish not to attend the In-Depth Help Sessions for your final project. These will be held during **class time November 20th**.

Additionally, you should read and reread the final project handouts thoroughly. Be sure you understand and implement all of the functionality you are supposed to. You should be able to code directly from your answers to these questions.

IMPORTANT NOTE:
This design check will be graded interactively! We still expect good written work. You should prepare your answers as if they would be turned in and graded. But this time you will be explaining your thought process to us during a 25-minute meeting. This way you can get live feedback on your design and start coding your Final Project sooner!

During the meeting, we expect you to have answers prepared for all of the questions listed on the next sheet. We will ask you to explain in detail a subset of the questions below and provide feedback on these questions.

We are requiring you to physically bring in the following documents for the design check:

1. UML diagrams for containment / inheritance
    - Include ALL classes (including support classes and Swing components)
    - Write out signatures of major methods (as in the design lab) either in your diagram or in a separate list.
2. Pseudocode for the important algorithms in your project, as described by question 3

You should also be able to answer any other questions a TA may have about your planned design. You can take notes away from the design check. Your grade will be emailed to you after your check.

The design check will focus on the material you have prepared in advance**. Do not come to the design check without an initial plan of attack.** If you have questions, you should visit TA hours prior to your check.

Please note that failure to complete the design check during the period of **December 3rd to 6th** will result in 17% off your final project grade.

Design Check Questions:

1. Write out a sample run of your program, explaining what the user is doing and how the program reacts. This should be very specific. "The user clicks the button and the pieces flip over." is not acceptable. Include information for each step about your program's flow of control. Is it completely user-driven, completely automated, or a mix of the two? How will the flow of control be managed (be specific)?

2. What data structure(s) will you use in this assignment? What will these data structures hold? How will you use them (be specific)?

3. Provide pseudocode for important methods. Make sure you explain the methods specified below.

    Note: Do not just regurgitate the pseudocode we give you in the handout! Think critically and write detailed pseudocode.

- Sketchy: You should provide extremely detailed pseudocode for undo/redo, resizing a shape, and writing and reading a CurvedLine to and from a file. Each line of your pseudocode should have an almost 1:1 correspondence with a line of Java. Again, be specific!
- Othello: You should provide pseudocode for your MiniMax AI algorithm and for checking for a "sandwich" (ie. when there is a valid move).
- Pacman: You should provide pseudocode for the ghost's breadth-first-search target-finding algorithm. Your pseudocode should translate almost line by line to Java, so be sure to include all your variable initializations and correct types.
- Indy: You should provide pseudocode for the major algorithm in your project. If you are still developing the specifics, it is allowed to be at a higher level than typical for CS15 Pseudocode.

4. Create a complete inheritance diagram. Include any interfaces you will write or use. **At this point it goes without saying - use UMLet!**

5. Create a complete containment diagram. When appropriate include base types and arrays as instance variables using the convention in the Design Lab. If you prefer—these major methods and instance variables can instead be included in list form as you did in Tetris. **At this point it goes without saying - use UMLet!**

6. Make sure you have a good understanding of the major components of your project.