Assignment for TCP2201 Object Oriented Analysis and Design

Trimester 1, 2019/2020

You may have a maximum of 4 people per team.

In your code, you must put comments documenting each method (function) as to who wrote that method. (If more than one person worked on a method, you may list all their names.)

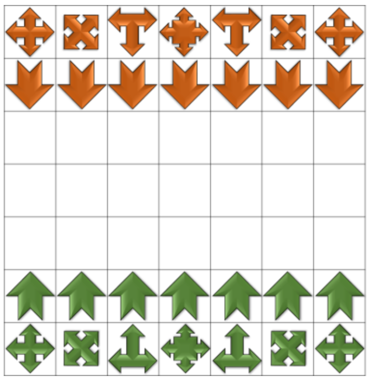
During evaluation, I may call any student to come and modify some code to do something differently in front of me to prove that they actually wrote the sections that their name appears on.

I also have a code plagiarism checker, which can identify code copied from others outside your team. If you copy code from anyone else, I will give you **zero**. In the past, I have given many zeros because I detected their plagiarism, so please avoid this heartache for yourself. **If you give your code to someone else to copy, it is also considered cheating and you can also get zero** so do not give your code to anyone else to copy. If you learn from an online source or past year projects, that's fine, but **write your own code**. If you are repeating this class, **you must start this project again from scratch.** Do not use any of your old team's code from the previous trimesters, or it will also be considered cheating.

The project: Perilanda Chess

You are to implement a Perilanda Chess game as a GUI-based Java Application. You do not have to implement a computer player – it can just be a two human player game.

Perilanda Chess is played on a 7x7 board, like this:



The above layout is the initial position of the game pieces. **None of the pieces are allowed to skip over other**.

The Chief can only move one step in any direction. The game ends when the Chief is captured by the other side.

The Trident can move any number of steps to the left or right, but only 1 step forward. When it reaches the opposing side of the board, it will turn around.

The Excel can move any number of steps diagonally.

The Tercel can move any number of steps up and down, or left and right.

The Advancer can only move 1 or 2 steps forward each time, but when it reaches the other edge of the board, it turns around and heads back in the opposite direction. (The icon should also turn around when that happens.)

After green has moved 3 times, all the green Excels will turn into Tercels and vice versa. Similarly, after brown has moved 3 times, all the brown Excels will turn into Tercels and vice versa. Then they will change again after the 6th move, 9th move, etc. (This makes Perilanda chess different from normal chess games, because the pieces will transform like that.)

You must use good object-oriented design concepts in designing your program. Subclassing, delegation, composition and aggregation should be used where appropriate.

You **must** use the MVC pattern, since this is a program with a GUI. This means you must not mix code that tells how the game works with the GUI code. This means that if you later put this game on another GUI system, e.g. Android, you will not need to change any model code – you will only have to replace the GUI code. So your view and controller classes must not have any of the game logic – all the game logic must be in the model classes only.

In **addition** to MVC, you must use design patterns in your code, and you must identify what design patterns you use and where. For example, the board might be a Singleton. The behaviour of the chess pieces might be implemented as a Strategy or State. (These are only ideas; you do not have to use these particular design patterns, and are encouraged to think about which design patterns might be suitable.) You may come to see the lecturers to discuss your design.

You should make your program user friendly, with suitable menus, save game, resizable windows, flipping the screen when it is the other player's turn, etc. For save and load game, the game should be saved into a text file so that it's human-readable.

You must document every class and method. You must practice proper indentation. **Marks will be deducted** if you do not do this.

Please see us early if you have problems – whether it is problems understanding what you need to do, or problems with team members who are not doing their work. The earlier you see us, the better chance you'll have of solving the problems. If you wait till the last week before the due date, it'll likely be too late to fix the problems.

Project Team Registration

You create your group on MMLS. If there is any discrepancy between the team members listed in the peer review Excel files and the MMLS team members, you will be liable for any loss of marks.

If you find as the project progresses that some people are not contributing or problematic in any other way, **please contact your lecturer immediately** so that remedial action can be taken before it gets too late. If you do not do so, you will be liable for any loss of marks.

Deliverables

1. Java Source code for the entire project
    a. Every class must be commented to show what the purpose of that class is. If the class is part of a design pattern, document what part it plays.
    b. Every method must be commented to show who wrote that method, and if it is not obvious, the purpose of that method.
    c. All the code must be properly indented.
2. Report containing
    a. A header like this:

TCP2201 Project
Trimester 1, 2019/2020
by <<TEAM NAME>>

Team Leader: Name, phone number, email
Team members:
Name, phone number, email
Name, phone number, email
Name, phone number, email

b. Instructions how to compile & run your program **from the command line**, and user documentation on how to use your program. **This is especially important if you developed your code using an IDE.** The lecturer marking might not have your IDE, or a different version of your IDE. **You are responsible for any loss of marks if the lecturer has trouble compiling and running your code**. (Especially be careful of capitalization of file names – Windows ignores the capitalization of file names, but LINUX and Mac do not. Some of the lecturers might be marking on LINUX or Mac.)
c. UML Class diagram – also indicate which classes are participating in which design patterns and what their roles in the design patterns are.
d. Use Case diagram – show the main functions
e. Sequence diagrams – for **each** of the use cases from the Use Case diagram.

The documentation, UML Class Diagram, Use Case Diagram and Sequence diagrams **must** reflect the version of the code submitted or marks will be deducted.

Zip up all the source code files together with the report and submit to the MMLS Assignment submission system by 6pm of the due date. **Each group submits one project, according to the MMLS group.**

In addition to the main submission above, each group member should submit the "Assignment Peer Review" Google Form *individually* to the "Assignment peer review" project submissions. There will be **three** peer reviews at intervals during the project. This peer review submission is secret – your team members will not see it, so you can be completely honest about how they performed.

- If the person is a sleeping partner or contributed almost nothing, you can evaluate them 0-1
- If they contribute very little effort, can rate them 2-3.
- If they are seriously involved and actively contribute, you may rate them 4-5. These marks are private and confidential. Kindly submit to us on MMLS in the "Assignment peer review" submission.

For the first review, you should rate your teammates from the start of the project till the first review. For the second review, you should rate your teammates from the first review until the second review. For the last review, you should rate your teammates from the second review till the project submission.

Note: **All** students must be involved in the programming. You cannot say "This student just did the documentation" or "this student just did the UML diagram."

Do not email your lecturer your project unless MMLS Assignment Submission is not working.

Late policy: 10% will be deducted if the project is submitted on 1 day late. 20% will be deducted if it is 2 days late. 30% will be deducted if it is submitted 3 days late. 40% will be deducted if it is submitted 4 days late. No submissions will be accepted after that.

Due Dates

- Peer review #1 due Friday 23 August
- Peer review #2 due Friday 13 September
- Project Due Monday, 23 September.
- Late policy applies until Friday, 27 September.
- Peer review #3 should be submitted immediately after you submit your project.

If you have submitted a version of the project, but then change it, you can re-submit until the cut-off date, and the new version will replace the old version.

TCP2201 Project Evaluation Form (30%)

Tutorial Section:	
Team Name:	
Group Leader:	
Member	
Member	
Member	

Prototype and Presentation (20%)

Item	Maximum marks	Actual Marks
Comments, indentation, following proper Java naming conventions, other Java style issues.	2	
Object-oriented concepts like subclassing, delegation, composition, aggregation, polymorphism, etc.	3	
Appropriate use of Model-View-Controller, and at least one additional Design Pattern.	3	
User friendliness and appropriate GUI components used, windows resize properly and the board scales properly, menus still work during game play, the board flips for each player, etc.	4	
Functional requirements fulfilled, e.g. the board is set up correctly, players can play through a game properly, all the pieces move and transform correctly, winner is declared, save game, load saved game, etc.	8	
Total:	20	

Report (10%)

Item	Maximum marks	Actual Marks
UML Class Diagram done and is coherent with the implementation	3	
Use Case Diagram done and is coherent with the implementation	2	
Sequence Diagrams for each use case done and is coherent with the implementation	3	
User Documentation done and is coherent with the implementation	2	
Total:	10	

Note: Individual marks will be adjusted after the lecturer interviews you, based on how much work each person did.