# Assignment for TCP2201 Object Oriented Analysis and Design

Trimester 2, 2016/2017

You may have a maximum of 5 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.

## The project: Barsoomian Chess

You are to implement a Barsoomian 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.

Baroomian Chess is played on a 5x8 board, like this:

The above layout is the initial position of the game pieces.

|  |  |
| --- | --- |
|  | The Arrow Box can only move forward, 1 or 2 steps. If it reaches the end of the board, it turns around and starts heading back the other way. |
|  | The Star can move1 or two steps in any direction. |
|  | The Cross can only move diagonally, but can go any distance. |
|  | The Heart can move only 1 step in any direction. The game ends when the heart is captured by the other side. |

After 4 turns, a Star will turn into a Cross, and a Cross will turn into a Star. (This makes Barsoomian chess different from all other 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 write a GUI-based, user friendly implementation of this game. There should be a proper separation of the model from the view (Model-View-Controller [MVC] pattern) – do not put the logic into the JPanel or widgets. You must provide an easy way for the human to enter the command sequence for their robot, and to modify their command sequences.

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. You might use an Abstract Factory, a Factory method, or a Prototype to create the pieces.(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 me 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.

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

**Please see me 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 me, 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

See the Excel file under the Assignment tab. You must register your team by 25 December 2016. You must also create your group on MMLS with the same team members. If there is any discrepancy between the Excel team members 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
   1. 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.
   2. Every method must be commented to show who wrote that method, and if it is not obvious, the purpose of that method.
   3. All the code must be properly indented.
2. Report containing
   1. A header like this:

|  |
| --- |
| TCP2201 Barsoomian Chess Project Trimester 2, 2016/2017 by <<TEAM NAME>> Team Leader: Name, phone number, email  Team members:  Name, phone number, email  Name, phone number, email  Name, phone number, email  Name, phone number, email |

* 1. Instructions how to compile & run your program **from the command line**, and user documentation on how to use your program.
  2. Use Case diagram – show the main functions
  3. UML Class diagram – also indicate which classes are participating in which design patterns and what their roles in the design patterns are.
  4. Sequence diagrams – for each of the use cases from the Use Case diagram.

Zip up all the source code files together with the report and submit to the MMLS Assignment submission system by 6pm, 6 February, 2017. **Each group submits one project, according to the MMLS group.**

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

**Late policy**: 10% will be deducted if it is submitted on 7 February. 20% will be deducted if it is submitted on 8 February. 30% will be deducted if it is submitted on 9 February. 40% will be deducted if it is submitted on 10 February. No submissions will be accepted after that.

# TCP2201 Project Evaluation Form (30%)

Project Due Date: Wednesday, 6 February, 2017. Late policy applies until Sunday, 9 February 2017.

|  |  |
| --- | --- |
| Tutorial Section: |  |
| Team Name: |  |
| Group Leader: |  |
| Member |  |
| 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 Design Patterns | 3 |  |
| User friendliness and appropriate GUI components used, windows resize properly and the board scales properly, menus still work during game play, etc. | 4 |  |
| Functional requirements fulfilled, e.g. 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** |
| Use Case Diagram done and is coherent with the implementation | 2 |  |
| UML Class Diagram done and is coherent with the implementation | 3 |  |
| Sequence Diagrams done and is coherent with the implementation | 3 |  |
| User Documentation done and is coherent with the implementation | 2 |  |
| Total: | 10 |  |