Assignment for TCP2201 Object Oriented Analysis and Design

Trimester 1, 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.)

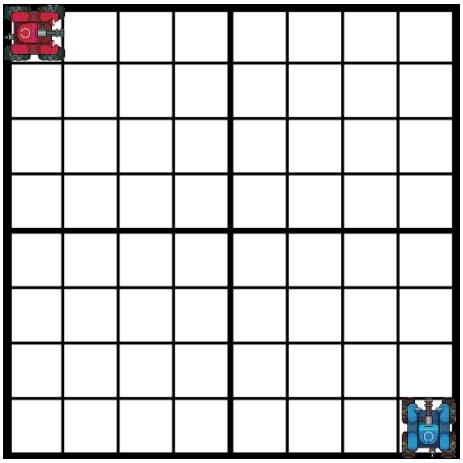
During evaluation, I may call any student to come and modify some code to do something differently in front of me to prove that they actually wrote the sections that their name appears on.

Every person must be involved in **both** programming and documentation. You cannot say "Bob does the programming and Mary does the documentation" because then Mary would not learn how to do the programming and the documentation might not reflect the truth about the program design. In fact, pair programming is encouraged – where more than one person works on the same piece of code at the same time – one person does the typing, but the other person is also engaged in the process and helping the person typing design and detect bugs in the code.

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: Robot War

The playing field is a 10X10 grid. The computer-programmed robot starts in the top left square, while the human-player's robot starts in the bottom right square.



The robots can only move up, down, left or right, and shoot up, down, left or right. The program sequence is 15 of these moves.

At the start of the game, the computer generates a new random sequence of commands for the computer's robot. The human then has to select 15 commands for the human robot. Then the round starts, and the robots run their respective sequences, animated on the screen.

If the human's robot manages to kill the computer's robot, then the human has won the game. If not, then the human has to write a new sequence to try to do that.

The aim of the game is for the human to take the least number of tries before their robot manages to kill the computer's robot. So, for example, if it only takes the human 3 tries, it's a better result than if it takes the human 10 tries.

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.

Saving the game will save the sequence the computer's robot uses, so that when you load the game, the computer's robot will use the same sequence as before. The human's last sequence should also be saved.

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 design patterns in your code, and you must identify what design patterns you use and where. For example, the sequence of commands could be implemented with the Interpreter pattern or the Iterator pattern. (These are merely suggestions – you may use any other design patterns you think suitable, as long as you document them clearly, showing me what classes correspond to which parts of the patterns.)

You may see your lecturer to discuss your design.

You should make your program user friendly, with suitable menus, save game, resizable windows, 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 your lecturer 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 29 July 2016. If you find that one or more team members isn't doing their part, please **report it to the lecturer early** so that action can be taken in time.

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. Name, phone number and email for each group member.
    b. Instructions how to compile & run your program, and user documentation on how to use your program.
    c. Use Case diagram – show the main functions
    d. UML Class diagram – also indicate which classes are participating in which design patterns and what their roles in the design patterns are.
    e. 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, 1 September, 2016. **Only the Group Leader is to submit the full project.** The other team members should only submit a text file showing the name, phone number and email for each group member and who the Group Leader is.

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

It is your responsibility to make sure that the correct file is uploaded. After uploading the Zip file, redownload and unzip it to verify that it was the correct file uploaded. If your file is corrupted, you will suffer the consequences.

Late policy: 10% will be deducted if it is submitted on 20 September. 20% will be deducted if it is submitted on 21 September. 30% will be deducted if it is submitted on 22 September. 40% will be deducted if it is submitted on 23 September. No submissions will be accepted after that.

Presentations are planned to be conducted during Study Week. All members must be present during presentation or marks will be deducted from missing members.

TCP2201 Project Evaluation Form (30%)

Project Due Date: Wednesday, 19 September, 2016. Late policy applies until Sunday, 23 September, 2016.

Lecture Section:	TT0
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 correctly, results are shown, 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	