Dalhousie University
Faculty of Computer Science

CSCI 3132 **Object Oriented Software Development** **Summer 2019**

GROUP PROJECT **[100 marks]**
Submission Deadline: July 14, 2019

1. Introduction:

This group project allows you to showcase your knowledge of object-oriented design and development, as well as enhance your team working and independent learning skills. You will work in groups of 4 to complete this project.

2. What you need to create:

You will create a text-based console I/O game that can be based on:

- Any existing board game
- Your own created board game
- Any existing game show
- Your own created game show
- Any text-based arcade game

A sample I/O of a board game known as LUDO is shown below:

```
You are playing the Basic version of LUDO:
Rules: ... (List the basic rules here)
You are NORTH

Turn 1:
Your turn, roll a dice: 4
N: 4 ---N-----------------------------------------------------
S: 3 --S-----------------------------------------------------
E: 1 E-----------------------------------------------------
W: 6 -----W-----------------------------------------------------

Turn 2:
Your turn, roll a dice: 5
N: 5 --------N-----------------------------------------------------
S: 1 ---S-----------------------------------------------------
E: 6 ------E-----------------------------------------------------
W: 4 ---------W-----------------------------------------------------
.
.
.
Turn 19: You need to roll less than 2 or 2 to finish
Your turn, roll a dice: 5
Sorry, cannot move 5 places. Better luck next time
N: 5 ----------------------------------------------------------N--
S: 2 ----------------------------------------------------S------
E: 3 ----------------------------------------------------E---
W: X ----------------------------------------------------W 1st place
West won 1st place

Turn 20: You need to roll less than 2 or 2 to finish
Your turn, roll a dice: 2
N: 2 ----------------------------------------------------N 2nd place
S: 3 ----------------------------------------------------S--
E: 4 ----------------------------------------------------E-
W: X ----------------------------------------------------W 1st place

********CONGRATULATIONS!!! You finished in SECOND place*****************
```

3. **What you need to show**
   - The game design and implementation will be based on the object-oriented concepts you have learned. These include:
     - Abstraction
     - Encapsulation
     - Modularity
     - Inheritance (optional multiple inheritance)
     - Polymorphism
   - Your design must clearly showcase these concepts
   - Your implementation must also clearly show these concepts
     - It is not enough to have a few classes while parts of your code are written procedurally.

4. **What you need to submit**
   - A zip file containing the following:
     - A word or pdf document containing:
       - User level description of game rules (in simple narrative)
       - High level domain model of your game
       - Use case diagram capturing all actors and the essential use cases of the game
       - Detailed class diagram showing the various classes and their relationships
       - Screen shots of your game in action (i.e. output)
     - Source code
       - Put it in a folder named "source" before adding it to the zip file
       - Should contain all header, cpp, and make files (if required)
     - Readme.txt file which contains
       - Instructions to compile and run your game
       - Specific contribution of each member of your group*

   **Please note that a student who makes no contribution to the project may get a zero mark.*

5. **Important Dates and Submission Deadline:**
   - Project start date: Tuesday, June 04, 2019
   - Submission deadline: Sunday, July 14, 2019
   - Presentation (max 10 minutes per group)
     - You will present your project in class
       - Present your design
       - Show your implementation details
       - Demo your game
     - Presentation dates: 16, 18 and 22 July, 2019

6. **How you will be graded:**
   - ***Object oriented design*** ***(25 marks)***
     - Domain model (5)
     - Use case diagram (10)
     - Class diagram (10)
   - ***Object oriented implementation*** ***(30 marks)***
     - Abstraction (5)
     - Encapsulation (5)

  - Inheritance (10)
  - Polymorphism (10)
- ***Quality of work and submission (25 marks)***
  - Good use of pointers and references (5)
  - Overall quality of the code (5)
  - Playability and correct I/O (10)
  - Readme.txt file as described above (5)
- ***Presentation (20 marks)***

Grading Scale for Group Project

	Excellent	**Good**	**Satisfactory**	**Marginal**	**Inadequate**
Object Oriented Design (Word or pdf document)	All game elements and game rules are clearly captured in the domain model and use case diagram. Class diagram accurately reflects the domain model. Necessary attributes and operations are defined for each class. Relationships are clearly identified and correct.	Most game elements and game rules are captured in the domain model and use case diagram. Class diagram accurately reflects the domain model. Most of the necessary attributes and operations identified. Most relationships are correctly identified.	Most of the game elements and game rules are captured in the domain model and use case diagram. Class diagram largely reflects the domain model. Some key attributes in each class are identified. Some relationships are correctly identified.	Effort has been made to capture game elements and rules in the domain model and use case diagrams. Attributes have not been identified in the class diagram. Most of the relationships are missing or incorrect.	Game elements and game rules are minimal. Domain model and use case diagrams are either missing or incorrect. Classes and relationships between them are not identified.
Object Oriented implementation (Code)	All OO elements (abstraction, encapsulation, inheritance and polymorphism) are correctly implemented for multiple classes. Implementation is solely based on OOP. Class design is closely followed during implementation.	All OO elements are correctly implemented for more than one class. Implementation is largely based on OOP. Class design is closely followed during implementation.	Implementation of all OO elements is visible in the code and is correct. Effort has been made to follow the class design during implementation.	Encapsulation and inheritance are visible and correct in the implementation.	OO elements are not or incorrectly implemented.
Quality of work and submission	Code contains clear comments and is properly indented. Pointers and references are used where appropriate. Instructions for users are clear with correct user input accepted and messages/output displayed clearly. Assignment instructions are followed. Readme.txt file clearly outlines each members' contribution. Code compiles and runs according to instructions in readme.txt file.	Code contains appropriate comments and is properly indented. Pointers or references are used where appropriate. Instructions for users are mostly clear with correct user input accepted and messages/output displayed clearly. Assignment instructions are largely followed. Readme.txt file outlines each members' contribution. Code compiles and runs according to instructions in readme.txt file.	Code contains some comments and is properly indented. Pointers or references are used in some places. Program does not check for correct user input; however, output is displayed clearly. Attempt has been made to follow assignment instructions. Readme.txt file outlines each members' contribution. Code compiles and runs according to instructions in readme.txt file.	Code contains very few comments and indentation is inconsistent. Pointers or references are not used. Output is displayed but instructions for users are not clear. Assignment instructions are not followed, however, Readme.txt file vaguely indicates each members' contribution. Code does not compile according to instructions or instructions are missing in readme.txt file.	Comments are missing and code is not formatted. Pointers or references are not used. Output is not displayed or is incomprehensible. Assignment instructions are not followed. Readme.txt file does not include contributions or compilation string.