Cairo University, Faculty of Artificial Intelligence

# FACULTY OF COMPUTERS AND ARTIFICIAL INTELLIGENCE, CAIRO UNIVERSITY

## CS213: Programming II
## Year 2023-2024
## First Semester

## Assignment 3 – Version 2.0

## Course Instructors:
Dr. Mohammad El-Ramly

### Revision History

| | | | |
|---|---|---|---|
| **Version 1.0** | By Dr Mohammed El-Ramly | 26 Nov. 2022 | Main Doc |
| **Version 2.0** | By Dr Mohammed El-Ramly | 26 Nov. 2023 | Revised |

**CS213:** Object Oriented Programming
Assignment 3 (8 marks + 3) – Version 1.0

Cairo University, Faculty of Artificial Intelligence

## Objectives

This assignment aims to (1) learn OOP concepts in C++, (2) using OOP to build systems with intermediate complexity, (3) Templates and STL, (4) Recursion and Backtracking, (5) Code Reuse.

## Instructions

1. **Part1: Deadline is Sat 9th of Dec. 2023 @ 11:59 pm.**
2. **Part2: Deadline is Sat 16th of Dec. 2023 @ 11:59 pm.**
3. **Part3: Deadline is Mon 18th of Dec. 2023 @ 11:59 pm.**
4. **Weight is 9 marks** + **3 bonus** marks.
5. Students will forms teams of **three** students **from the same lab/section**.
6. Please submit **only work that you did yourself**. If you copy work from your friend or book or the net **you will fail the course**. تسليم حلول منقولة من أى مصدر يؤدى إلى الرسوب فى هذا المقرر
   لا تغش الحل أو تنقله من أى مصدر و اسألنى فى أى شئ لا تفهمه لكن لا تنقل الحلول من النت أو من زملائك أو أى مكان

## Task 0 (0 marks)

1. Review OOP / STL / Templates / Recursion / Backtracking.
2. **Review all code examples given in lecture and in classroom page.**
3. Create a **private NOT public GitHub** repo for the project and **use it for development.**

## Individual Task 1 (2 marks) – Individual Problems from Sheet 3

The **smallest ID** will solve problems 1 and 4. The **middle ID** will solve 2 and 5. The **higher ID** will solve problems 3 and 6. These problems will be in **Sheet 2** under Google Classroom.

**What to deliver?**

- **Working code in standard C++ and using STL not using third-party libraries.**
- Name your file A3_SheetPb**XX**_YourID.cpp (XX is pb num)
- **Each problem will be loaded ONLY as .cpp file in a separate place in Classroom.**

## Individual Task 2 (4 marks) – OOP Design and Development

You are given a generic implementation of a board game that consists of the following generic classes, separated into a header file and implementation files:

1- **`GameManager`** that owns a board and 2 players and runs the game and swap turns.

2- **`Player`** that represents a generic game player that has a name and can do x, y moves

3- An abstract game **`Board`** with functions to be defined in children for rules of the game.

4- A **`RandomPlayer`** that returns a random x, y move.

This framework is suitable for building multiple board games like X-O and similar ones. **It shows the power of OOP in reusing existing code to build new apps with less effort.** You are given an example X-O game implementation that includes the following classes:

5- **`X_O_Board`** represents the board of X-O and the rules for that game.

6- **`X_O_App`** is an application that creates a **`GameManager`** with **`X_O_Board`** and 2 players and runs the X-O game.

# **CS213:** Object Oriented Programming
# Assignment 3 (8 marks + 3) – Version 1.0

Cairo University, Faculty of Artificial Intelligence

Every team member will develop using the existing framework, a new board game similar to X-O. The smallest ID will develop game 1, the middle will develop game 2, and the largest will develop game 3.

For each game, you will need to inherit from the **`Board`** class and to create a special board for this game. You might also need to inherit from **`Player`** to create a special player for this game. You also need to create a random computer player for each game that chooses a **correct** random move.

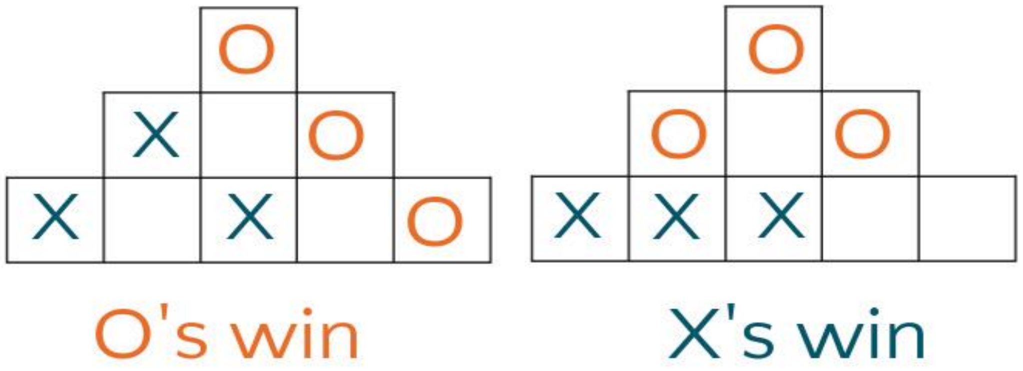
**Keep definition separate from implementation in a header file.**

**Game descriptions are taken from:** https://www.whatdowedoallday.com/tic-tac-toe-variations/

## 1- Pyramic Tic-Tac-Toe

The game board is shaped like a pyramid. Five squares make the base, then three, then one. Players take turns marking Xs and Os as in traditional tic-tac-toe.

**Winning:** The first player to get three-in-a-row vertically, horizontally, or diagonally wins. See two examples of winning positions, below.



## 2- Four-in-a-row

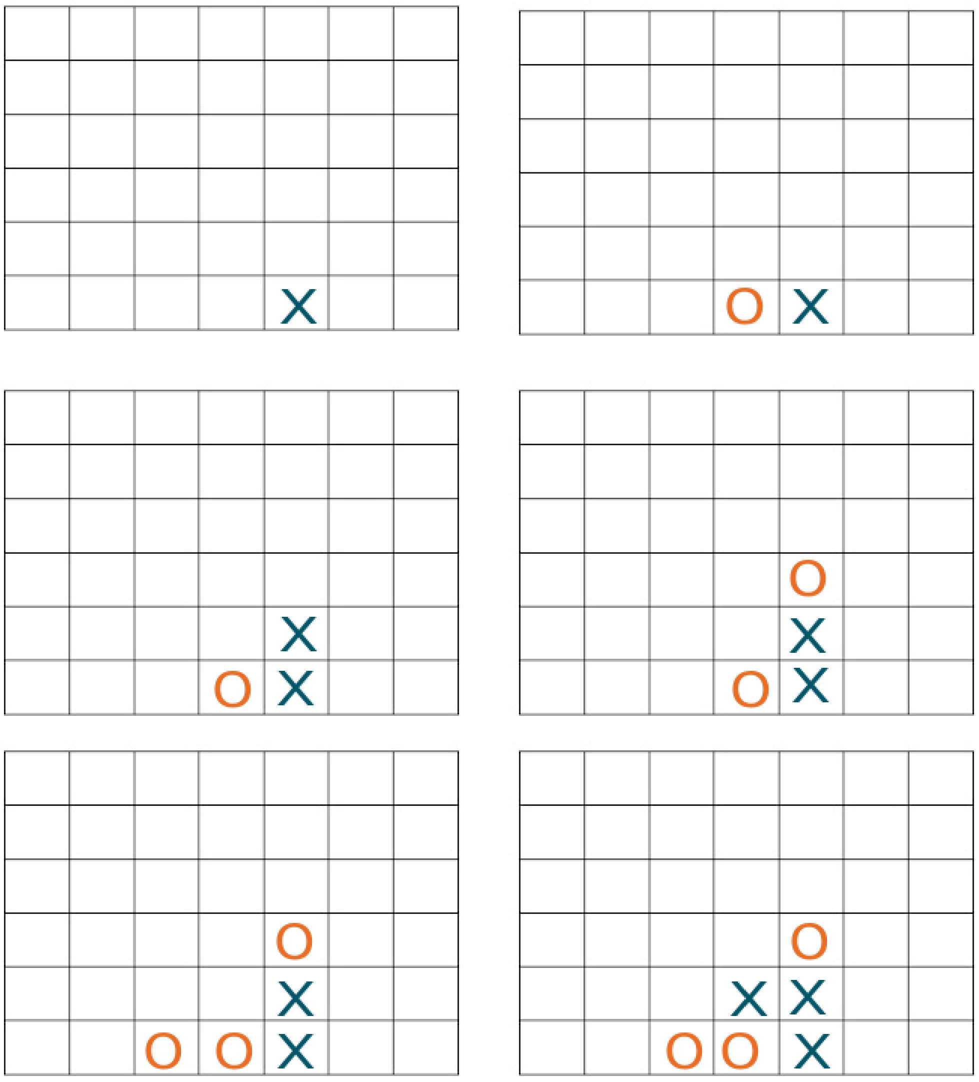
You will recognize four-in-a row as a two-dimensional version of the classic game, Connect Four. The game board consists of a 7 x 6 grid. Seven columns of six squares each. Instead of dropping counters as in Connect Four, players mark the grid with Xs and Os as in tic-tac-toe.

**Rules**: The first player places an X in the bottom square of any column. Taking turns, players make their mark in any column, as long as it is in the lowest square possible. See image below for an example of possible first six moves.

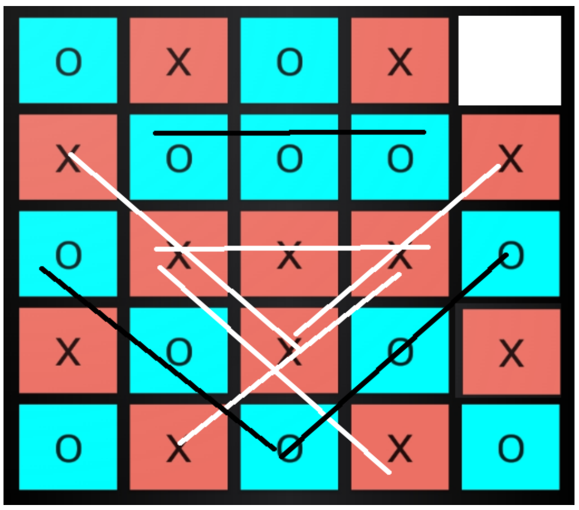
**Winning**: The first player to get four-in-a-row vertically, horizontally, or diagonally wins.

# **CS213:** Object Oriented Programming
Assignment 3 (8 marks + 3) – Version 1.0

Cairo University, Faculty of Artificial Intelligence



Example play for six moves of four-in-a-row



Example of finished 5 x 5 game and X wins with 5 times 3Xs and O loses with 3 times 3Os

### 3- 5 x 5 Tic Tac Toe

This tic-tac-toe variation is played on a 5 x 5 grid. As in the traditional game, players are Xs or Os.

**Rules:** Players take turns placing an X or an O in one of the squares until all the squares except one are filled. (Each player has 12 turns for a total of 24 squares.)

**Winning:** Count the number of three-in-a-rows each player has. Sequences can be vertically, horizontally, or diagonally. Whoever has the most, wins. Can one mark be counted in more than one three-in-a-row sequence? Decide ahead of time, yes or no. It is easier in implementation to allow counting more than once.

**CS213:** Object Oriented Programming
Assignment 3 (8 marks + 3) – Version 1.0

Cairo University, Faculty of Artificial Intelligence

## Group Task 3 (1 mark) – OOP System Integration

The team will integrate their work and deliver an app that gives a menu of choices with the games available and if the user chooses a game, it opens the board for him to play against another player or a random computer player.

## Group Task 4 (1 mark) – Code Review and Code Quality

Code review is an essential part of software development to ensure its quality and readability and reduce the bugs. Each team member will read the code of the other members and produce a report about it. Develop a suitable code review process that involves

- Read more about code reviews and their tools also. See
  https://kinsta.com/blog/code-review-tools/
  https://github.com/joho/awesome-code-review
- Review the code using a this checklist **and write in your report what violations occur and which part file and line number.** https://www.codereviewchecklist.com/
- You can do review by hand record the results in a word file. Or you can use a tool to run and manage the review (online, or installed on your machine or integrated with GitHub),

## What to deliver for parts 2, 3 and 4?

1- **Written code in standard C++ not using third-party libraries** representing the whole gaming app that has the 4 games, including X-O supplied by professor**.**
2- Name your file **A3_Task2_3_4_YourGroup_YourIDs.zip** (with cpp, header and project files). **NO EXE. Do not upload exe or object files.**
3- **A pdf** report, including:
   a. A detailed program description **and the idea behind each function** in the inherited `Board` class you wrote for your game**.**
   b. A work breakdown table saying **who did what.**
   c. A report of the **quality of the code** of the other members and what mistakes you found there and how you reviewed it.

## Individual Bonus: Task 5 (1 mark) – AI Player

For your game, inherit a smart computer player class from the given player class instead of the random player. It should be able to choose the best movement given the current board status and play against human in order to win.

Read about min-max and backtracking algorithms and choose a suitable algorithm or try a suitable heuristic algorithm that chooses the best move for the computer. Try several game plays to ensure your algorithm works well.

Notice that the current X-O game has a totally random computer player that chooses random moves. You need a smarter one for your game.

**CS213:** Object Oriented Programming
Assignment 3 (8 marks + 3) – Version 1.0

Cairo University, Faculty of Artificial Intelligence

## Group Bonus: Task 6 (2 marks) – GUI

Develop a GUI for the entire integrated game app that starts with a GUI window with options and then opens every game in a GUI window and allows mouse / touch interactions for all the games.

**What to deliver:**

1- Submit the original game code.
2- Submit the modified game code with bonus options.
3- **Explain in the report what you did, what tools you used, how it works, and who did what.**

## Submission Instructions

Cairo University
Faculty of Computers and Information
CS251
Software Engineering I
Project Description
2015
Version 2.0
Project Team
Dr Mohammad El-Ramly
m.elramly@fci-cu.edu.eg

**Team will submit into classroom the following:**

1. **Original non-cheated solutions. Cheated solutions prevent you from learning. Basically, you cheat  yourself not me.**
2. **Part1**: For each problem, submit a file named A3_SheetPb**XX**_YourID.cpp
3. **Part2**: For each your gaming app, submit a file named **A3_Task2_3_YourGroup_YourIDs.zip** with the required items and report for **Tasks 2 & 3**.
4. **Part3**: For each your gaming app, submit a file named **A3_Final_YourGroup_YourIDs.zip** with the required items and report for all tasks and bonus.
5. Team will create a **private** project in **GitHub** to collaborate on code.
6. Each team member can work individually on his part. **But the team must provide ONE integrated and working program and report.**
7. Team members are expected to help each other but not do work of others.
8. Team members are responsible of testing all the programs and making sure they work.
9. **All team members must understand the details** of all programs and be able to explain it or even modify it if needed. TA can ask any team member about any of the programs developed**.**

## Marking Criteria

1. 1 x 2 For a correct working solution for each **individual problem** (individual)
2. 4 For a working original **game** that plays correctly according to the rules. (individual)
3. 1 For correctly **integrated program** that offers a menu of choices of games. (group)
4. 1 For **quality** report that shows evidence that ALL team reviews ALL code (group)
5. 1 For a working smart **AI player** for your game with a suitable algorithm (individual)
6. 2 For a working **GUI** for the whole app and every game (group)
7. -1 For not using GitHub.
8. -8 For cheating any part of the code, even 1 line.