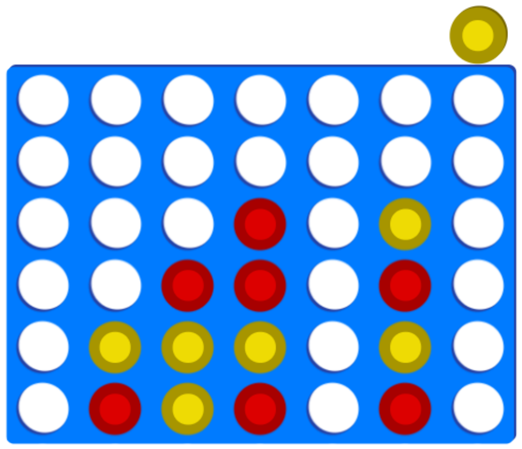
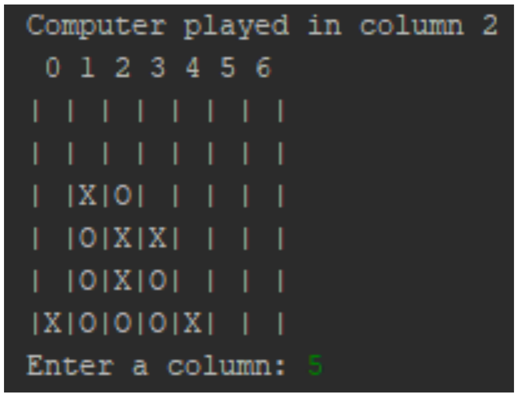
Fall 2021 **COSC 3P71 Artificial Intelligence: Assignment 1**

Instructor: Beatrice Ombuki-Berman
Teaching Assistants: Nicholas Aksamit, Tyler Crane, Liam McDevitt, Alanna McNulty

Due Date: Monday, October 18th@9 AM. (**NOT LATES**)
Goal: Implement the game Connect 4 with an AI that runs on Minimax
Languages: Any programming language available in our labs

Connect 4: A two player game in which each player takes turns placing coloured pieces into a 6 row by 7 column grid. The turn player chooses a column to place their piece, and the piece falls to the lowest position in the column not already occupied by another piece. The goal for each player is to be the first player to create a horizontal, vertical, or diagonal line of 4 or more of their pieces in an adjacent line.





```
Computer played in column 2
 0 1 2 3 4 5 6
| | | | | | | |
| | | | | | | |
| |X|O| | | | |
| |O|X|X| | | |
| |O|X|O| | | |
|X|O|O|O|X| | |
Enter a column: 5
```

A graphical user interface is not required, a simple text-based interface such as the one shown above is fine. The program should interact with a human player and provide a simple interface for entering moves and viewing the board. The program should also filter any invalid inputs that user may make appropriately, such as trying to play in a column that is full.

The computer player will be implemented using the Minimax algorithm described in class to determine what moves to make. A max depth will be used to limit how deep the search tree will grow, so that the AI decides what move to make in a reasonable amount of time. In order to intelligently decide what branch in the tree the AI should follow, a heuristic evaluation function will need to be implemented to evaluate the boards in the leaf nodes.

Take care when designing your evaluation function; a strong heuristic will significantly impact the performance of the AI when playing against a human. Consider a random board and ask how you would determine which player is currently performing better. As well, once a heuristic is implemented, try to find ways that you can exploit the heuristic to win. Finding such strategies will provide a good idea of any weaknesses in your ai and mitigating these flaws will allow you to greatly strengthen your heuristic function. You are encouraged to experiment and try out different approaches.

Hints:

- Put some thought into how you will implement the game before you begin programming. How will you represent the board? Will this representation allow you to easily generate new boards when building the search tree? How will the program interact with the user? How will the AI store possible game states? Thinking about such problems before you begin coding will minimize errors and simplify the entire implementation process.
- Implement your program one step at a time, don't try to do everything at once. Start with your data structures, then set up each of your functions one at a time. You should not be implementing the AI if you can't play with two human players.
- Test each function as you implement to ensure they are all working correctly. Simple functions can often be overlooked, and a small bug can escalate into your entire AI not working properly.

Hand in:

i. Submission will be done electronically through Sakai. Submission will include all your commented source code, as well as detailed instructions for compiling and running your program. Failure to provide adequate explanation and documentation may result in a submission not being graded. **Do not assume the marker uses the same IDE as you.**

ii. Please note that the virtual COMMONS is available to all students at Brock. You are not required, but if you prefer to (or need to) you can use the virtual COMMONS instead of a personal computer. Machines in the virtual COMMONS have IDEs for Java, Python, and C#.

iii. Any questions/concerns regarding the grading of any assignment MUST be raised within 7 days of graded assignment hand-back. In this case, please send your concerns/questions to Course Coordinator: Tyler Crane at: "tcrane2@brocku.ca". To better serve you, please don't send multiple queries on the same topic to Tyler, Professor, and other TAs. Tyler will be the point of contact for any such queries and the Professor will receive them all at once from Tyler.

iv. We will be creating a FAQ for this class and adding some of your questions there (while maintaining your privacy). In this case, please send all your email questions to Tyler at "tcrane2@brocku.ca". Tyler will try his best to respond to all your questions as soon as possible, so please wait 48 hours before you contact course instructor if

Tyler has not responded. Note: Tyler will be working closely with the course instructor, and she will have access to all your questions

v. Feel free to use any language (with reason) as long as it can be opened and executed on the lab computers. Examples include Java, C#, C++, and Python. No matter your choice of language, ensure you have provided sufficient comments such that your program can be understood. At minimum, include a comment describing each function/method and class/module.

vi. This assignment is to be completed individually. Plagiarism detection software will be applied in this course for all submitted work. Additionally, a number of assignments will be randomly selected, and the authors will be asked to explain their code and submitted documents.