SODV 1202

Introduction to Object Oriented Programming

Project

Connect Four

Project: Connect Four

Overview

Connect Four is a two-player connection game in which the players first choose a symbol/color and then take turns dropping one Symbol/colored disc from the top into a seven-column, six-row vertically suspended grid. The pieces fall straight down, occupying the lowest available space within the column. The objective of the game is to be the first to form a horizontal, vertical, or diagonal line of four of one's own symbols/colored discs–Wikipedia.

Directions

The best way to tackle a large project is to break it down into smaller tasks. In this project, you will work with your teammate(s) towards the goal of completing the project by completing a series of milestones listed below. The directions for each milestone are listed below. Each milestone must be presented or demonstrated in your final project report.

Milestones

1.	Team Name and General Strategies	Groups are formed, so pick a team name and use Google/Wikipedia to get an idea of how the connect-4 game works. Brainstorm some general strategies your team will use to design both a 2-player mode and a player vs. computer mode.
2.	UML diagram	Create a UML diagram outlining the classes you plan to create for your project.
3.	Program Skeleton Development	Start developing an object oriented program with your teammate(s) to play the Connect Four game.
4.	Setup Git Repository	Prepare your Git repository to be used by the team. You should find the issue tracking system to be helpful in keeping your project organized and on-schedule. Create issues for all the development tasks that you have brainstormed and divide the workload among your team.
5.	Send a GitHub link to your instructor	In the d2l, project page, submit the GitHub link, and always make frequent commits so that your instructor can inspect your activity and progress.
6.	Prepare for the future	Throughout the project you will need to file additional issues as bugs or new ideas are discovered.

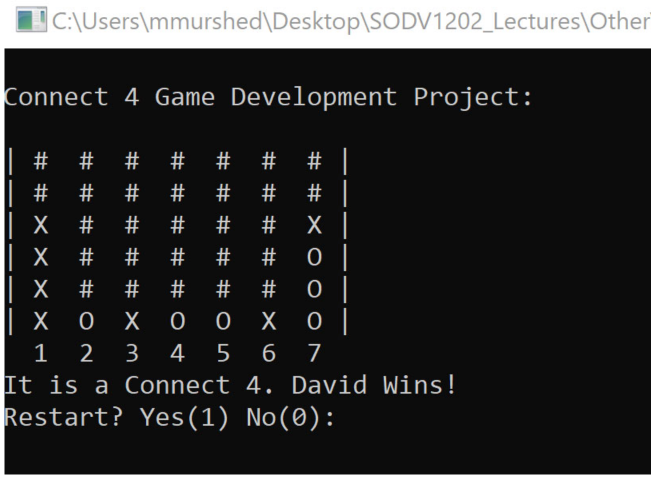
7.	Complete Core Classes	You should design at least four classes, along with any other classes needed to implement your project: Hints: You may have a controller class to play the game, a model class that implements intermediate steps and holds information about the game, two classes that extend the Player abstract class. One should be a human player. The other should be a computer player that can use any rule-based algorithms or AI algorithms. Both of these classes should communicate with the world via an object that is created in your controller class. You may also have a class that interacts and provides communication via text input from the keyboard and output on the Console.
8.	Complete Basic 2 User Game	Your primary focus is to complete the game for two human users. Once complete, you can start developing advanced logic for the computer.
9.	Make sure, OOP principles are followed	Most of the points will be allocated for understanding and implementing OOP principles, Therefore, before advancing your project further, make sure you have taken advantage of all OOP principles discussed in this course.
10.	Development of an AI algorithm	If time permits, develop an AI algorithm/ rule-based algorithm to make your program compatible with human Vs. computer mode.

Rules

Connect 4 is a classic board game where two players (typically two different colors or two different symbols) take turns dropping disks (symbols) in a grid of 7 columns by 6 rows. The discs attract the gravity and fall to the lowest empty space in their column (also rests on top of other pieces, if any). The winner of the game is the first to have 4 discs of their symbol/color in a row. This can be horizontally, vertically, or diagonally.

Figure: Connect Four.

If you are new with the game, this video (https://www.youtube.com/watch?v=utXzIFEVPjA) should help out. It talks about how to play Connect Four. Connect Four is a solved game, meaning, if the first player can make the proper move, he always wins. Therefore, if you can design good algorithms, and the computer is allowed to make the first move, ideally, the computer should always win. If you want to know the strategies to win connect four, you can check out this link (https://www.youtube.com/watch?v=7DnCAgHeFOk) In your project, you will program the Connect 4 game using the numbers as input that has both a 2-player mode and a 1-player mode. When your program starts, the game should have some start screen where the player can select either 2-player mode or 1-player mode using a console command.


```
C:\Users\mmurshed\Desktop\SODV1202_Lectures\Other

Connect 4 Game Development Project:

| #  #  #  #  #  #  # |
| #  #  #  #  #  #  # |
| X  #  #  #  #  #  X |
| X  #  #  #  #  #  O |
| X  #  #  #  #  #  O |
| X  O  X  O  O  X  O |
  1  2  3  4  5  6  7
It is a Connect 4. David Wins!
Restart? Yes(1) No(0):
```

Snapshot of a console Connect Four Board.

For simplicity, assume the first player is always ‘X’ and the second player is always ‘O’. You can also use text color if you like. Players will interact in the game using numbers. If the first player enters 1, the leftmost column should drop his symbol (‘X’) as deep as possible. However, if the column is already full, the player cannot drop a piece in that column, nothing should happen (your program must not crash here). Once a player has successfully played a piece, their turn is over. The game should check if the move created four-in-a-row of the same symbol. If it is the case, your game should print the winner. If not, it should simply pass the turn to the other player. If all 42 blocks in the grid are played, it is a draw, and no one wins.

In the **2 player mode**, players will use numbers to select which column they wish to drop their piece. The game should then prompt that it is the other player turn. The game should clearly indicate which players’ turn it is using their Disc Symbol/Name/Color (for example, you could have text saying "It is David’s turn:"). The turns should go back and forth until the game ends.

In **1 player mode**, the player will play against a computer program. This does not have to be a "good" AI, just some process by which moves are taken automatically without player input. The game should randomly decide to have either the human or the computer go first (since moving first is a big advantage). From there, the game proceeds until it ends.

In either case, when the game ends, the game should show some text indicating either who won or that the game was a draw. The game must then return to the "start" screen, where a player can once again choose either 1-player mode or 2-player mode.

Tips

Do not spend a lot of time working on the AI until all basic components are properly functioning using the OOP concept. You can achieve more points by following good programming practices and OOP principles as much as possible. For example, interacting multiple class objects, uses of interfaces, method overloading, overriding, virtual functions, and delegates will help you attain more marks than the AI based design. Moreover, you will not get extra credit if your AI can beat the instructor, though it can be fun to try. Focus on completing the game working first with 2 players.

Always think about OOP concepts! Consider what objects to use, how to use properties, values and attributes to come up with a design with more private methods/functions/properties/fields and a few public methods/functions/properties/fields.

Evaluation

The project will be evaluated for design, code quality, and functionality. A full break-down of how marks will be assigned is listed in the table below.

Task	Marks
Design ○ **Chosen design makes development and maintenance easier** ○ **Classes and methods divided logically** ○ **Classes demonstrate with abstraction, property, encapsulation, Inheritance, polymorphism and good object oriented design**	5
Code Quality ○ **Completing written milestones** ○ **Writing easy to read codes** ○ **Consistent style** ○ **Documented with comments where necessary including XML comments**	10
Functionality ○ **Demonstrates that a strategy was planned and implemented** ○ **Functionality must be non-trivial / significant** ○ **Program does not crash**	15
Total	30

At the end of the project peer evaluations will also be requested, feedback and viva-voce will be factored 50% in final grades.