**0Unit 3 programming project**   
**Connect 4 with AI**   
**Christian Andrew Wellens-Miles**

Contents

[Analysis 3](#_Toc36514798)

[Problem identification 3](#_Toc36514799)

[Clients expectation 4](#_Toc36514800)

[Reasoning for computational solution 4](#_Toc36514801)

[Interviews with stake holders 5](#_Toc36514802)

[Response 8](#_Toc36514803)

[Problem research 9](#_Toc36514804)

[Existing solution research 9](#_Toc36514805)

[Features I might implement 10](#_Toc36514806)

[Features of the proposed solution 12](#_Toc36514807)

[Limitation of the solution 12](#_Toc36514808)

[Requirements 12](#_Toc36514809)

[Hardware requirements 12](#_Toc36514810)

[Software requirements 12](#_Toc36514811)

[Success criteria 12](#_Toc36514812)

[Design 13](#_Toc36514813)

[Task 13](#_Toc36514814)

[How it might look 14](#_Toc36514815)

[Login 14](#_Toc36514816)

[Register 15](#_Toc36514817)

[Connect 4 17](#_Toc36514818)

[Comments from the stakeholders 18](#_Toc36514819)

[Login system 18](#_Toc36514820)

[Register system 18](#_Toc36514821)

[Connect 4 18](#_Toc36514822)

[Diagram for the sub routines links 19](#_Toc36514823)

[Sub routine (need more detail) 19](#_Toc36514824)

[Login system 20](#_Toc36514825)

[Pseudocode algorithm 20](#_Toc36514826)

[Connect 4 20](#_Toc36514827)

[Input and output 22](#_Toc36514828)

[Key variables 22](#_Toc36514829)

[Validation 23](#_Toc36514830)

[Buttons 23](#_Toc36514831)

[Textboxes 23](#_Toc36514832)

[Testing method 23](#_Toc36514833)

[Waterfall model 24](#_Toc36514834)

[Spiral model 24](#_Toc36514835)

[RAD 24](#_Toc36514836)

[What I've chosen 24](#_Toc36514837)

[Functions to test 24](#_Toc36514838)

[Development 25](#_Toc36514839)

[Game creation - refer to SC 25](#_Toc36514840)

[Checking if there’s 4 in a row. 27](#_Toc36514841)

[Artificial intelligence 30](#_Toc36514842)

[Simplest Artificial intelligence 30](#_Toc36514843)

[Medium Artificial intelligence 31](#_Toc36514844)

[Prototype one 31](#_Toc36514845)

[Errors faced 37](#_Toc36514846)

[Prototype 2 38](#_Toc36514847)

[Errors with this code 43](#_Toc36514848)

[Finished Medium artificial intelligence 44](#_Toc36514849)

[Errors 53](#_Toc36514850)

[Hard artificial intelligence 55](#_Toc36514851)

[Difficulty form 58](#_Toc36514852)

[Design 58](#_Toc36514853)

[Creation 58](#_Toc36514854)

[SQL database 60](#_Toc36514855)

[My SQL database 60](#_Toc36514856)

[Connecting it with VB 62](#_Toc36514857)

[Login system 63](#_Toc36514858)

[Login form 63](#_Toc36514859)

[Register form 64](#_Toc36514860)

[Errors 66](#_Toc36514861)

[New Register form 66](#_Toc36514862)

[Problems with login system 67](#_Toc36514863)

[Final Product 68](#_Toc36514864)

[Connect 4 68](#_Toc36514865)

[Functional program 70](#_Toc36514866)

[Login form 70](#_Toc36514867)

[Register form 72](#_Toc36514868)

[Selection form 75](#_Toc36514869)

[Connect 4 game 77](#_Toc36514870)

[Vertical win 77](#_Toc36514871)

[Horizontal win 78](#_Toc36514872)

[Diagonal wins 78](#_Toc36514873)

[Easy artificial intelligence 79](#_Toc36514874)

[Medium artificial intelligence 81](#_Toc36514875)

[Hard artificial intelligence 90](#_Toc36514876)

[Success criteria 96](#_Toc36514877)

[Feedback 96](#_Toc36514878)

[Stakeholders 96](#_Toc36514879)

[Clients 96](#_Toc36514880)

[Future endeavours 97](#_Toc36514881)

## 

# Analysis

## Problem identification

I was approached by a teacher interested in showing the development of AI and how it can get progressively difficult as the user defeats the artificial intelligence (AI). The teacher, Joanne Jackson wants the game to be simple and easy to understand for young students. So, the problem is attempting to create an artificial intelligence which will develop as the user defeats the AI and easily display to children a game which they are familiar with, with simple graphics so they don’t get confused or flustered so it’s easy to watch. Designing an artificial intelligence and displaying different functionality to easily be understood and witnessed is the problem but doing it to a simple game with limited functionality for the AI.

## Clients expectation

The customer audience we are targeting would be teenagers learning about AI as it will be demonstrated in the simplistic game of connect 4 as the AI will increase in difficulties due to the number of wins you have.

The customer can use a fully fleshed out game with varying difficulties of AI to make it a challenge for connect 4 professionals or amateurs. If you would prefer to use it and play against your friend, then you can play locally with them. The customer will have to login or register if they want, then link to a database which will extract your data like high score or current progress to then set you off against the AI level you were on when you logged off. It will contain a win streak counting how many AI you have beaten before you ultimately fail. The game will start off easy for every user increasing in difficulty as the user plays more and more and starts winning. As the user wins the game the Ai increases in skill level according to the amount you’ve won and contains more adept programming to place counters more strategically than randomly placing them in available slots, increasing in difficulty until the user is unable to beat the AI like other games such as Civilisation V which contains varying difficulties of AI starting with simple then advancing to Deity which gives the AI buffs with then having more code to be more of challenge to beat.

This program could be distributed worldwide to teachers in computer science field to display the development of AI on a game and how it can match the skill of a human depending on the amount of games that they have beaten the AI at.

I will talk to the teacher on a regular basis to see if this is what she requires in order to present AI and what I can do to make it easier for the pupils to understand what’s happening.

## Reasoning for computational solution

The problem that I’m trying to solve is solely depends on a computational solution as I’m displaying the effect of simplistic AI compared to complex coding which you can’t really do without a computer program to show it. Problems that can occur is that the AI might be hard to differ from its other varying difficulties as there’s only so much you can do in connect 4

The displayed product will be the product the customer receives as the product represented will have to contain AI with difficulties and a simplistic UI for the students so they can easily determine the difficulty of the AI according to the complexity and skill they possess when they the game of connect 4. The AI “thinking” will be hidden from the user, the user will only see the move that the ai takes to counter the users’ moves.

In order to solve this problem I will have decompose the task into multiple small tasks such as coding the game to be functioning firstly then code the AI in the easiest level then add upon the code bit by bit to make the moves more calculated reacting to the previous move the user did rather than placing a counter in an open slot randomly, not calculated at all.

The data from the logins and high scores will be stored in a SQL database where the program will retrieve and send data in order to check the username is correct or to write and read the level of AI the user is on, doing SQL queries to extract the relevant data in accordance to the username and amount of AI beaten. This will be better than using a text file as it will allow for more advanced SQL querying allowing the program to more efficiently gather select data in order to output data

Procedures that I will have are:

* To check the grid whether the user or AI has got a 4 in a row to decide if they have won the game.
* To determine what level of AI will be used against the player with each major level increasing the depth of the coding for the AI so basically unlocking this code so that the AI can use this.
* It will decide whether the counter dropped is either red or yellow.

Functions that I will contain:

* To check if the board is currently filled up with counters and if they have not got a 4 in a row then the game will be a draw as they cannot place any more counters.
* Check whether the location they tried to place it in has got a counter or not return true or false.
* Which players turn it is only if the user has chosen to play locally with a friend. Either making the counter turn yellow or make the counter turn red

I will use object oriented for the counter locations on where they can be placed having attributes of colour and size with location with methods of blocking other counters from being able to be placed in the same location as this object

## Interviews with stake holders

Questions for stakeholder:   
Q1: How would you use AI in the modern day?   
Q2: How would you demonstrate AI?  
Q3: What would increase your ability to teach the varying complexity of AI?  
Q4: Do you have any features that you would like to include in order for a program to better suit younger students?  
Q5: Do you think using a simple board game would be suitable to show off Ai?  
Q6: How would you go and implement complexity of AI into game?

I ask question 1 and 2 to confirm that I have the right audience for my project and see if they have any insight in how they would teach AI.

I ask question 3 to see if they already have a solution in order make solving the problem even easier

Question 4 is down to personal preference to the teacher whether they want a more interactive game or more like a movie where they can slowly watch the complexity of AI increase over time.

Question 5 and 6 is to reinforce my idea of using a simple board game such as connect 4 to display the varying difficulties and advancements of ai as the game goes on.

#### Questions for user:

Q1:Would you say displaying AI in a board game is the easiest way?  
Q2:If no, How would you go about demonstrating AI?  
Q3: Could you recommend a simple game of similar principles to connect 4 to display the complexity of Ai?  
Q4: Is there a reason why artificial intelligence shouldn’t be represented on a board game?  
Q5:Would you find learning about AI from the proposed solution beneficial?  
Q6:What features would you want to have in a board game?

Question one is asked to see if the user could have any other thoughts on how to demonstrate AI and to see if they agree or disagree with the proposed solution.

Question 2 is asked to explore different ideas the user might be having and how to be integrate it into the solution and make it as usable as possible.

Question 3 and 4 is there to propose my solution to get feedback of the user and to see whether they agree with the idea and allow them to input any criticism and issues in it and what I can do to improve the solution.

Question 5 and 6 look at whether the proposed solution can be enhanced in any way and whether the solution proposed can be useful to solve the problem and surveying if the user has some personal preferences that will work well to better teach about AI.

Answers from stake holders

**Joanne Jackson:**  
  
**Q1: How would you use AI in the modern day?**   
Key stage 4 it is briefly mentioned for the larger essay star question for the wider world aspect of computing e.g use of Ai in medical fields and/or production. So, its essential to teach younger audiences

**Q2: How would you demonstrate AI?**  
I would demonstrate it in a main discussion-based way. Discussing different application and how they affect the real world. I have often thought about demonstrating the differences of using AI methods over randomized methods this may be a more engaging to show pupils the affects that even simple AI methods can have simple application.

**Q3: What would increase your ability to teach the varying complexity of AI?**  
A simple User interface would enable me to efficiently teach about the versatility of AI relevant to younger students new into computer science and eager to learn about artificial intelligence and what’s possible with coding.

**Q4: Do you have any features that you would like to include for a program to better suit younger students?**   
a simple user interface and simplistic controls allowing the user to easily observe the reactions of the artificial intelligence, responding accordingly. I would also like If the program to allow a teacher to look over the results of the users within their class.  
  
**Q5: Do you think using a simple board game would be suitable to show off Ai?**   
For newer students which are familiar with these games I believe this will be effective to display artificial intelligence on a simplistic level. For more advanced students it will be suitable to show how an artificial intelligence can be created to preform a simple role but wouldn’t be able to show them major complexities as the artificial intelligence is limited to a certain amount of moves.  
 **Q6: How would you go about implementing varying complexity of artificial intelligence to a game?**

Within a game the artificial intelligence will be programmed to combat a certain task, reacting to what the user does, so the best way I would go about implementing the complexity is by displaying different (noticeable) difficulties highlighting how artificial intelligence can be a variety of complexity.

#### Response

From the answers we have gathered from Joanne Jackson indicate that having a program that will contain a simple user interface would be more efficient to teaching the audience which this program will directed towards with simple controls, so their confusion of the program is limited. The program that will be implementing the artificial intelligence should be a well-versed game so that the users are already familiar with what to do. The artificial intelligence will be presented within a game so that users can understand more effectively on what and why the artificial intelligence is moving where it’s moving.   
  
Answers from users

***Jack:***  
**Q1: Would you say displaying AI in a known board game is the easiest way?**Yes, as if it’s a familiar board game that we’ve grown up playing, watching an artificial intelligence take a move can be easily understood on why and what it’s responding to enabling the people with experience the game more understanding and appreciation towards the artificial intelligence.

**Q2:** **If no, How would you go about demonstrating AI?**  
If I was to say no then it would be for the issue that the artificial intelligence might be restricted as a well-known board game that is viable to younger audiences might have limited amount of moves that the artificial intelligence will have really crippling its potential.  
  
**Q3:** **Could you recommend a simple game of similar principles to chess to display the complexity of Ai?**  
Yes, due to my point in the first question, the game is well known so it would be easier to understand what the artificial intelligence is doing so we can watch it adapt while we play more and more games. However, it might be difficult and time consuming to design a functional AI to be a challenge.

**Q4: Is there a reason why artificial intelligence shouldn’t be represented on a board game?**  
I believe if an artificial intelligence was implemented into a board you could easily display the complexity of AI (artificial intelligence) within the game and make it enjoyable for people to play and interact with. Learning the potential for AI.

**Q5:** **Would you find learning about AI from the board game beneficial?**  
There are many things you can learn from an AI in a board game like training to get better, having replay ability as the AI will change in difficulty as you similarly adapt better knowledge of the game.

**Q6: What features would you want to have in a board game?**  
I would like to see a comparison of scores between me and my friends. Having the board game be simple and something with replay-ability, challenging me the more that I win. Seeing the artificial intelligence become more and more adept as time goes on.

***Charlie:***

**Q1: Would you say displaying AI in a known board game is the easiest way?**It would be beneficial if the artificial intelligence is created in a board game well versed by people as it demonstrates artificial intelligence easier and helps you to understand what the artificial intelligence is doing more. However artificial intelligence is better displayed in real life instances like robotics.

**Q2:** **If no, how would you go about demonstrating AI?**I would recommend robotics as it can interact to your voice and hand signals however this might be too adept to program and perhaps difficult to get the expensive hardware.

**Q3:** **Could you recommend a simple game of similar principles to chess to display the complexity of Ai?** Chess would be a good example as the skill gap is easily distinguishable and ai has a lot of possibilities perhaps too many. This game has multiple artificial intelligence out there that you might be able to take inspiration from. This type of game is well known by the people affected by the program.

**Q4: Is there a reason why artificial intelligence shouldn’t be represented on a board game?**  
The fact that there could be limitations in the moves that the artificial intelligence degrades the possibilities of the artificial intelligence in this field but for the task at hand there’s no reason not to use a board game.

**Q5:** **Would you find learning about AI from the board game beneficial?**

Personally, I would find it useful as it will easily display on a game hopefully familiar to me and allow me to learn about the complexity of artificial intelligence while also enjoying my time.

**Q6: What features would you want to have in a board game?**I would enjoy having a leader board as I’m a competitive person and I enjoy trying to become first. I would also like the program to be easy to use and simple in graphics as I have a poor desktop

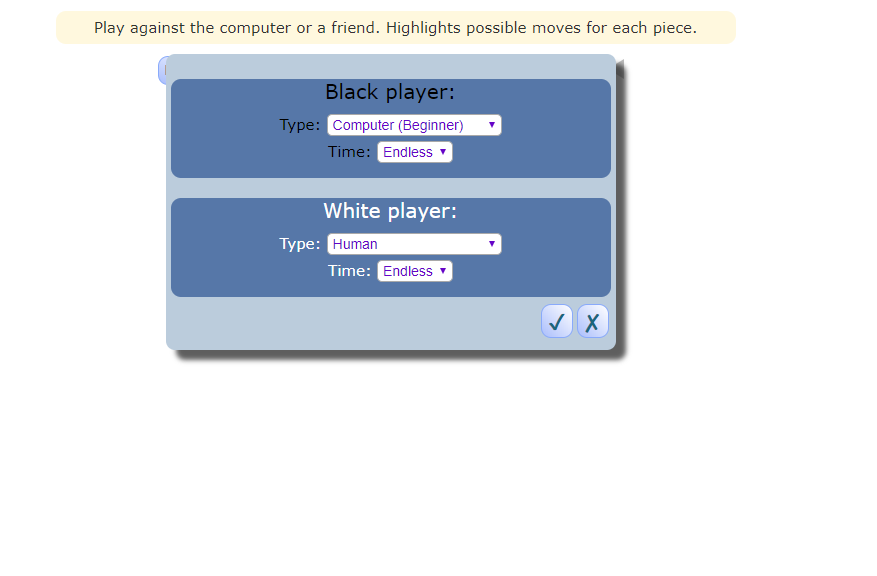
### Response

From the responses I have received from the clients of the program I have deduced that a board game will be the most suitable platform for solving the problem of displaying varying complexity in artificial intelligence as they find it more effective if it’s displayed this way. The Board game should remain simple as it will be time consuming to program the artificial intelligence to react to every situation and for it to not be too simple as the artificial intelligence won't have enough possibilities to solve the problem. The board game should also have some sort of way to compare results with other users playing the game as 2 of the clients both want this feature implemented.

## Problem research

### Existing solution research

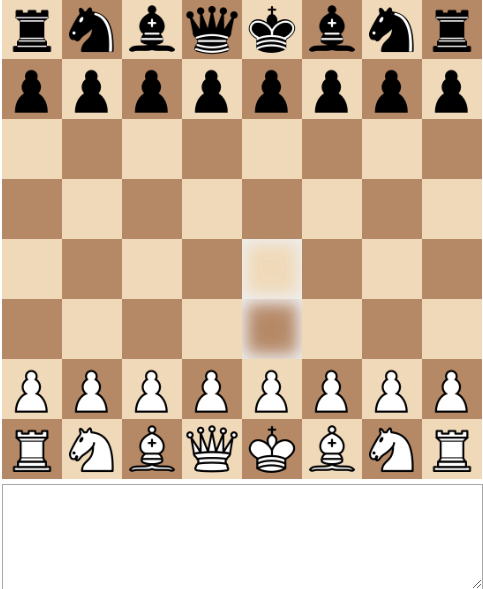
Chess is a well-known board game across the world. Due to this people have already created artificial intelligence to react to the near endless possibilities so researching how and what they have done will be useful.



This uses an interface before the actual game begins allowing the user to select whether the user is playing against an AI or computer. Indicating where the piece on the board will go can help people who are new to the game to understand what pieces do what. This feature is simple and can help anyone understand how to play which will be useful idea in any board game I decided to go with.



There is a variety of artificial intelligence each with its own set of algorithms and functions so that they react differently to the user. The user interface is also simple and easy to use having large buttons without any irrelevant details keeping it precise.



The board has very simple graphics with all the pieces being easily identifiable with their designated colour to make it easier for the user and more effective for someone to see where the artificial intelligence has reacted.As said before the highlighted squares are there when you hover over a piece with your mouse allowing you to see where on the board it can move.

### Features I might implement

* Simple user interface as recommended by both the client and the stake holder a simple user interface will allow the client to effectively understand the movement of the artificial intelligence.
* Indication of where the piece will go by highlighting to make it easier for people who are relatively new to the game to understand how the game works.

However, I believe with the response from the clients and stockholder this would be too complicated to understand the reaction of the artificial intelligence making it unsuitable to solve my problem.

I’ve decided that connect 4 was the best board game to suit my problem due to it being a simple board as it doesn’t have near infinite moves. It is well known within the age group it’s directed at. The comments I have received from stake holders and future users of this program indicated me towards this game. The game of connect 4 may have some other aspects implemented into to increase functionality and enable the artificial intelligence to have more versatility.



I will also include most of these features like difficulties except the username will have to be a certain login which will be stored in a data base where it will query the relevant data that it will store such as score/level of AI it is on with the database being in first normalised form to be the most efficient it can be. The UI will be much more simplified than this one to easily enable a class to understand what’s happening and easily display the function of the AI.

##### Features I would use:



I would use a different method on selecting the difficulty using the number of wins or in context of the picture the score being certain milestones increasing the level of AI according to each win. The various difficulties will determine how well the AI will be able to block the opponent from winning and how capable they are at seeing a few turns ahead. Calculating what they need to do to win.

The name for the human will be stored on the data base which contain all the data about the user like wins and password. Instead of you being able to click anywhere on the row to place a counter the program will only respond if you click the button at the top of the column.

There are examples of connect 4 games where you can always win a game using an AI to calculate every move and knows exactly what to do. Due to my projects game being a well renown and used game there has been many qualified people setting up strategies and methods allowing AI to be quite different each difficulty. Reacting to the human player.

### Features of the proposed solution

The features that will be implemented will be:

* Functioning AI with varying difficulties sets to the amount of wins the user has
* Database which will store the username, password, Score and the amount of wins the user is on
* A game of connect 4 with a winning condition

### Limitation of the solution

The limitation is the level of AI I can implement to connect 4 to it being a limited amount of moves they can do really limiting the reaction to the human player. The AI can only have limited features which will be how well it can block the player and how well it can win being good for an example but limited in its functionality.

One of the limitations will be that the code is written in visual basic requiring a specific interpreter in order to even boot up the program decreasing the versatility of the program.

if the user is visually impaired it will be difficult for them to play the game as the two colours which are meant to differentiate the player might look similar if not the same. Limiting the functionality of the program.

As this program is designed to only work on a computer so the limitation is that it requires that you have visual studio restricting the audience's capability to use it.

## Requirements

### Hardware requirements

**Mouse and Keyboard**-The mouse would be there to input into the connect 4 board dropping a counter into a column. The keyboard would be needed to login to your account on the data base written in SQL.

**a monitor -** as this program will be solely dependent on visualisation of the outcomes of your input in the game of connect 4 to drop the counters in the columns and to be able to see where to click.

### Software requirements

Since the programming is written in visual basic it will require the same software to interpret and compile the code in order to use it.

It will also require a software that will be capable of executing visual basic code.

## Success criteria

|  |  |
| --- | --- |
| Criteria | What to show |
| Connect 4 board | The board must be a 7x6 grid, allowing both players to alter it and affect it. |
| AI with varying difficulties | The code of the structured difficulties of AI showing them to have varying capabilities with some being able to successfully stop the player from getting connect 4 but won't be able to see when they have a clear victory to fully demonstrate varying complexity of AI and how they can be made to be unbeatable |
| Login system | Screen shots of the data base holding all the data and the design view of the form. |
| Inputs data into database | Data from the user must be able to enter their details into a program and for that data to be copied over to a database. |
| Retrieves data from database to check it | Data can be retrieved from the database to be used to compare the users inputted data and the data held within the database |
| Allow the user to re-enter their details | All the user to do a query within the database to compare their inputted data to the ones under the same username (Primary key) |
| Allow it to represent the user | That the users details are displayed on the program from when they entered it |

# Design

## Task

The task at hand will be split into two main parts due to it being simpler to decompose the task and make it into smaller tasks which are simpler to solve rather than doing the whole thing at once. The program will be split between two parts one of them will be:

* **Login system:**   
  This part will consist of storing the user's data into a database and allowing them to access the data whenever they sign into their account as shown above and allow them to input their own personal details into the database. I’ve chosen to decompose the program, splitting login system because this part differs greatly compared to the rest of it as this is managing and storing data while the rest is about the actual game.This will be again broken down to:
* Login
* Registering
* Storing in a database

It will be broken down like this so it's easier to manage and to solve as there will be more smaller tasks making it more comprehensible for me.

* **Connect 4 game**

As said previously the coding of the connect 4 game and the artificial intelligence differs from what I have to do for the login system, so splitting these up is logical making it easier for me to program them separately then merge them together when they are designed  
Within the connect 4 game I will further decompose it, so the tasks are nothing more than just algorithms being:

* Creating AI
* Creating the game
* Linking it with the login system

## How it might look

### Login

This form will enable the user to login to there account retrieving the information from the SQL Database to check if the data matches. If however the user doesn’t have an account then the user can register.  
  
The Textbox’s will allow the user to enter a username and a password which will be directly linked to the SQL database preforming a search query to find if the username then matches the password entered.  
  
If the user hasn’t registered into the SQL database then this will redirect the user to another form where this Is possible.  
  
The login button once clicked is the enabler to the SQL database allowing it to query for the data to see if the data inputted into the text boxes are linked to one another.

### Register

Confirm password  
(textbox)

Password  
(textbox)

Surname  
(Textbox)

First name  
(Textbox)

Back  
(button)

Register  
(button)

The function of the form is to add the details of the user into the database by having them enter it via the text boxes  
  
The text boxes are used to input the users data, creating a username using the first name and surname. Making sure the password match up y forcing the user to enter it twice.  
  
The back button is merely there to return to the login form if they accidently pressed it

The Register button once activated enters the data within the SQL data base searching firstly if the user has inputted the same First name and surname with the exact same password. If this occurs then the user will be forced back to the form to edit some of the information.

The criteria for login system will be completed as this checks everything needed for one.

### Connect 4

The main function of this form will be to play a game of connect 4 with either a friend or an AI effectively showing how the complexity of the ai.

Drop counter button- This button is the function of a person dropping a counter on the board. The AI will also use this function to react to the player and in turn counters/ randomly dropping their counter. This will meet dropping counter, the criteria for the program.

Setting button – This button will allow you to alter things about the game

The grid – The grid will be very simple to effectively teach what’s happening within the game and how the AI reacting. Once the button is activated, the box on the first row will either go Red or it will go yellow depending on whose turn it is. This will complete the ‘connect4 Board’ criteria .

## Comments from the stakeholders

After presenting my model to the people intrigued with my idea allowing them to voice their opinions if anyone had any concerns as I’m certain this will be the basis of my final program, I’ve received these comments:  
  
Joanne Jackson – This exactly how I foreseen the program to look like, a simple User interface so its easily usable by the students and teachers to represent the functionality of the artificial intelligence implemented into the game. Overall, I’m happy to see the design of the forms simple and easy to use which is key when teaching students or when younger people are using the program allowing them to easily go through it.  
  
  
Algorithms

### Login system

Once the program is booted the user will be met with a login screen. If the user has a username generated and a password set then they can enter there details into a textbox then click the login button where the database (where all the data is stored) will request the usernames deatails, if the username doesn’t exist then it will cause and error and return a false value requesting that the user enter the correct username prompting them that it is incorrect. However if the username is valid then it will look if the password entered matches the password set in the database depending if they match, it will either progress the user to the connect 4 game if password matches else it will output that the user has entered an incorrect password.

If they do not have a login then they will have to register using the register button found upon the login user interface

### Register system

If the user is new to the program there will be a register button where the user can enter there details into textbox’s where the program will input the data if they meet these requirements:

* Unique name and surname
* Password matches
* Textbox’s are fully filled out
* Password is strong enough

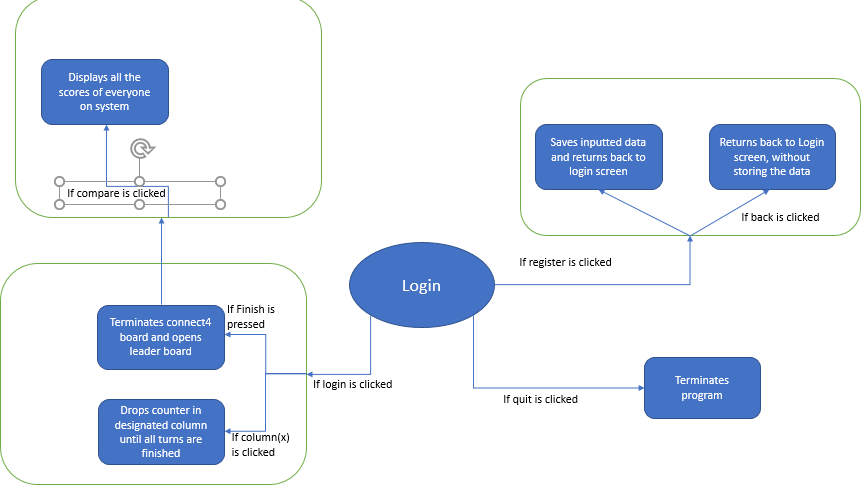
After registering is complete and valid then the program will save the data upon the data base and then direct the user to the connect 4 game.

### Connect 4

They will hold the programming for the artificial intelligence reacting to the users input.   
Allows the user to drop counters onto the board. Displaying their username pulling it from the database as they logged in saving there score and displaying it onto the user interface

Once the user is finished with the connect 4 game then the user will be able to view a leader board where he can then compare results extracting the results from the database.

Diagram for the sub routines links



## Sub routine (need more detail)

As the artificial intelligence is the main focus of the program the user interface will be lacking as more focus will be to develop the artificial intelligence. The artificial intelligence will be designed to operate on a connect 4 board specifically programmed to counter and win the game.

### Login system

This is used to enable the user to personally store their detail referring to the program allowing them access to view by pressing the login button.

Searching function

### Pseudocode algorithm

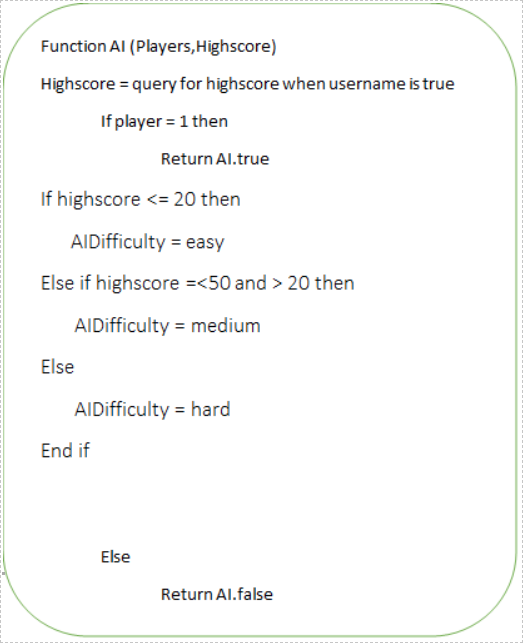
Function login (username,Password)  
 Searchname = username   
 searchpassword = Password   
 If searchname and searchpassword = Databasequery(Username,Password) then   
 Return True   
 else   
 return false  
 Print(“The username or password is incorrect)

End if   
end function

### Connect 4

The connect 4 game will decide if the user will be playing with artificial intelligence or playing with two people, allowing this so they can play the game if they want to.

This function will set the difficulty of the artificial intelligence and either enable it or disable Artificial intelligence and depending on the high score of the user the artificial intelligence will react differently as more or less of the programming is available to it



Function Difficulty(AIDifficulty)  
 If Ai = True then   
 If AIDifficulty is easy then  
 AIEasy.enable

Else if AIDifficulty is medium then   
 Return AIMedium.enable  
 Else   
 Return AIHard.enable ‘

End if

End if

Gives AI access to code

Input and output

|  |  |  |
| --- | --- | --- |
| Inputs | Response | Output |
| Login | Searches the data bas to see if data entered is matching | You will either be logged into your account or you will be asked to re-enter the password or username |
| Register Button | Re directs the user to a register form where the user can enter there details which will be entering data into the database | The username will be created with the data being stored and able to be pulled from the data base to allow the user to login and keep track of score |
| Back button | Allows transition from forms | Reverts the user to the previous form in this case the login form. |
| Drop counter button | The program responds by dropping a counter on the column that the user has pressed | It will drop the counter down that column deciding whether it’s red or if it’s yellow. |

Key variables

|  |  |  |
| --- | --- | --- |
| Name | Data type | Use |
| Username | String | A name comprised from the users name and surname. Its used to allow them there unique login |
| Password | String | To secure the account so the only person (Disregarding the teacher) to know it is the person who set it. |
| Login | List |  |
| Column1-7 | Integer that will store which row it should be placed on | This will increment every time the assigned column is pressed |
| Player | Holds value of counter | This will decide what colour is dropped on the grid altering each time a counter is dropped |
| Multiarray(6,7) | This creates a virtual grid of the connect 4 game in a 2D array. | This will be used to check if a player has gotten 4 in a row to end the game. Setting a certain value in the array to add up to a unique product to decide whether red or yellow player wins |
| AI | It will be Boolean to either enable AI or have two players | This will be true if AI is to be used in the game enabling a piece of programming. |
| Rng | This will be random to get a integer that is random | This value will determine where the simple level artificial intelligence will drop there counter. |
| Difficulty | This will store the level of difficulty the artificial intelligence | This variable will be used to decide which programming for the artificial intelligence will be used within the game of connect 4 |
|  |  |  |

Validation  
All data the user enters must be analysed to ensure that the data is relevant and is correct to avoid any errors occurring due to syntax errors.

### Buttons

There are no vulnerabilities with buttons as they are assigned a function and contains its individual variable to ensure that there are no errors within it this is mainly applicable with the ‘drop counter’ buttons on the connect 4 board.

### Textboxes

The text boxes for the first name and surname will be strictly string so only letters will be entered into the textboxes while the password textboxes will enforce that there will be characters and at least a number so that the password is secure. The information when registering will not be inputted into the database if these requirements are not met. There will be an output to indicate that there is an error that the user has done so that they can correct it.

Testing method   
Each function will be tested using the debugger and through using it as the user would so that all the bugs or errors will be presented forward ensuring that the features are correctly functioning. These tests will be thorough with each installation of every feature of the program.

I will go through every button and text box to ensure that the success criteria are met and that there are no bugs which will occur when using it. This will accomplish the algorithms within the program is responding as intended once interacted with. Using both black box testing and white box testing to ensure that all the bugs a rooted out.

### Waterfall model

Waterfall model is useful when you don’t have a full team of so suitable for smaller projects where the problem is clearly identified. It contains minimal client intervention so it could lead to the client not actually liking the finished product. If any changes are required, you will have to start from the beginning to alter it.

### Spiral model

This method is more suitable when a problem is large and you have a large amount of people actually working on the features can be added in a systematic way solving the decomposed problem piece by piece constantly having a client checking over the solution to ensure that you’re meeting there expectation. This method is only suitable for large projects like full-fledged games and not smaller games as this method will cost too much.

For my project this isn’t suitable as the client has set clear instructions and has given me a problem with only one solution.

### RAD

This model is suitable when you’re trying to create software in a short amount of time say a few months. This has less technical risk and adaptable if the clients view changes. Features may be reduced due to this model requiring a small amount of time not suitable for small problems as there isn’t enough there.

Overall, this model isn’t suitable for my problem as it’s not compatible with small projects which I would say that my project can be deemed as small.

### What I've chosen

I will use the waterfall model to create my program as its most suitable for a small prject like my own and rather straight forward not requiring any prototyping. This would be the most time efficient one and will allow me to solely focus on the solution of the problem.

## Functions to test

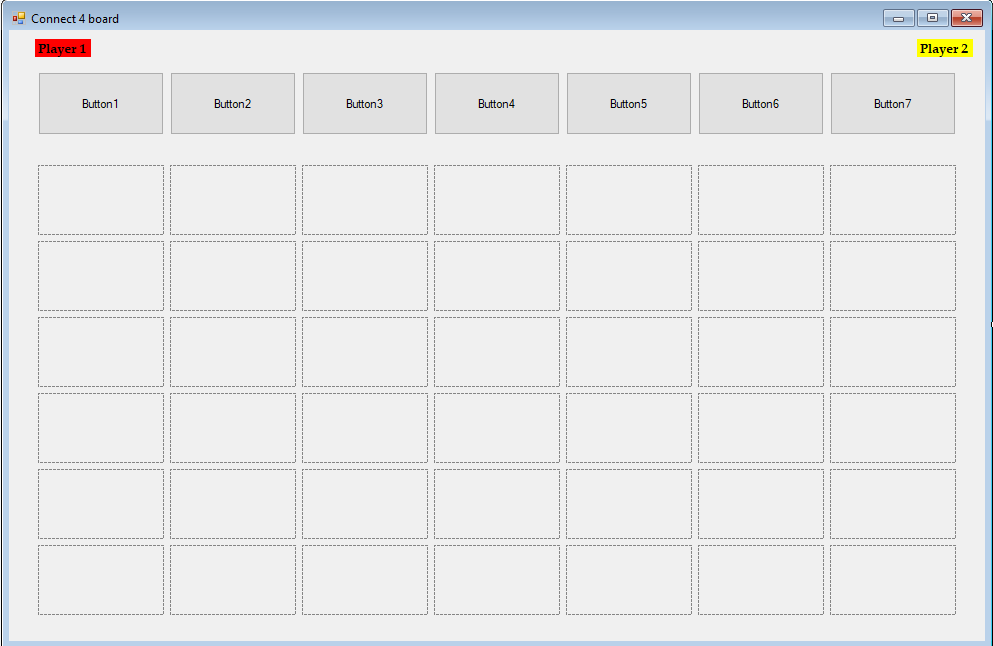
Like the success criteria I will keep linking back to this table to ensure everything within the system is working as intended.

|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game |  |
| Inputted data is in correct data type e.g only characters for names |  |
| Allows user to input into database |  |
| Database can retrieve relevant information and pass it to program |  |
| ‘Register now’ button redirects user to register form |  |
| ‘Back’ button returns user to login screen from the register form |  |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) |  |
| ‘Register’ button inputs data within the database and displays it |  |
| User can select Artificial intelligences difficulty |  |
| ‘Drop counter’ button allows the user to drop the counter |  |
| Winning conditions can be met and is viable |  |
| Artificial intelligence is present within the game |  |
| Artificial intelligence with simple coding works |  |
| Medium difficult artificial intelligence is functioning |  |
| Hard difficulty artificial intelligence is functioning |  |
| The game terminates after the user is finished with program |  |

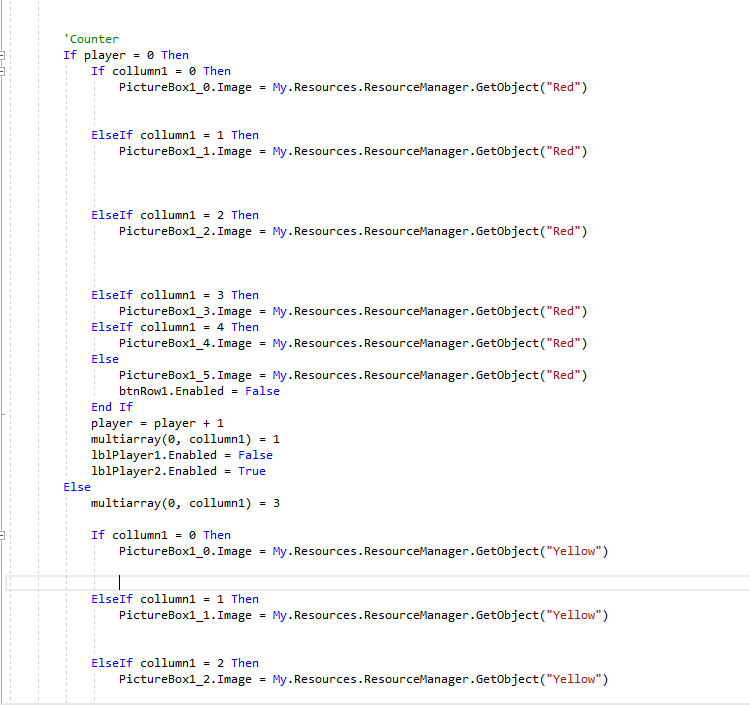
I will test these functions by doing alpha tests sending out a copy of a prototype to selected clients so that they can full test the game finding any errors or bugs while also sending a report on what they like and dislike about the approach taken in creating the solution.

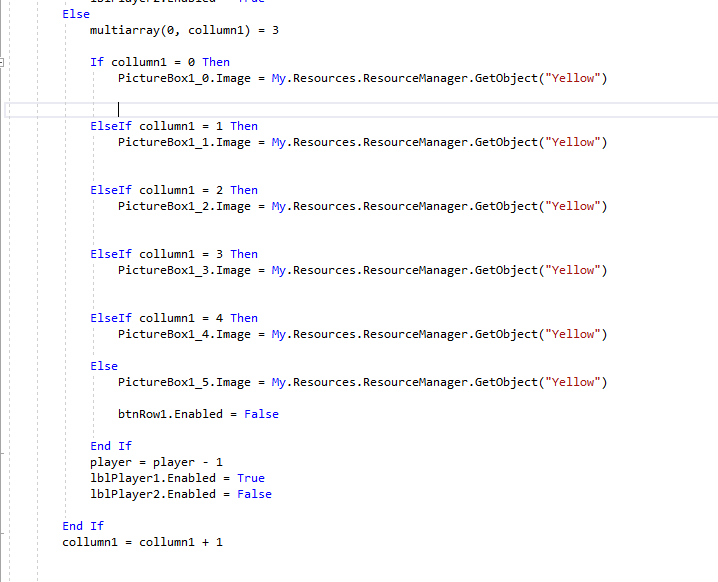
# Development

## Game creation - refer to SC



Simple design to easily demonstrate the AI and to ensure smoothness with simple functions to allow each move being played sticking to the similar design created beforehand.



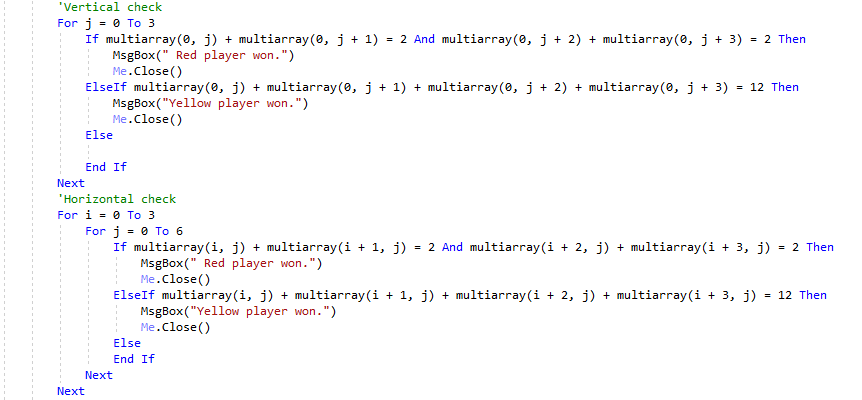


Each button will have similar coding used to simply add a counter to the board with the variable Column x (x being an integer depending on which column it's on, on the board). This piece of code is how the player would go ahead and drop a counter into a column displaying it either red or yellow due to the player turn which will be displayed with the players name being faded out when it’s the oppositions turn.

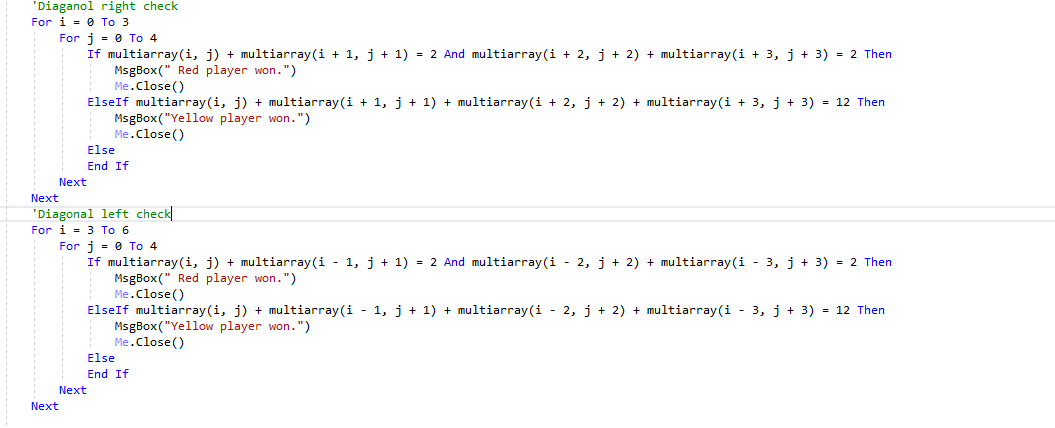
The code ***“Multiarray (0, collumn1) = 3“***is setting a prime number in an array to display that it is Red. Otherwise the array will the set the number where the counter should be as 1. After each button is pressed the variable ***Column(X)*** is incremented to abide to the rules of connect4.

Currently it will be event driven rather than Object oriented as this is just a prototype for the game as its easier to test and maintain as everything is being set up.

### Checking if there’s 4 in a row.



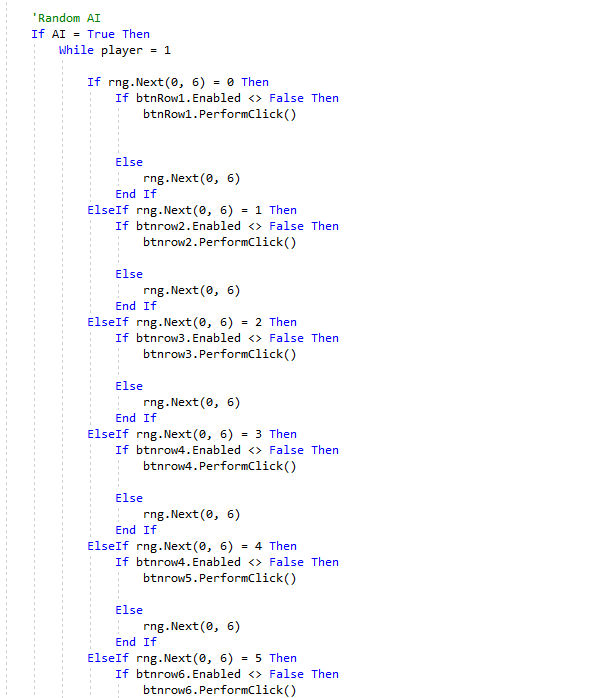
As spoken of in the last screenshot a value is assigned in an array indicating whether it is a yellow counter there or a red counter assigning different values. This algorithm sorts through this array vertically and horizontally to see if the values assigned to the 2D array add up to a specific value indicating that there are 4 of the same coloured counters in a row. This applies to both vertically and horizontally. If there is a victor, it will display a message saying which colour won.

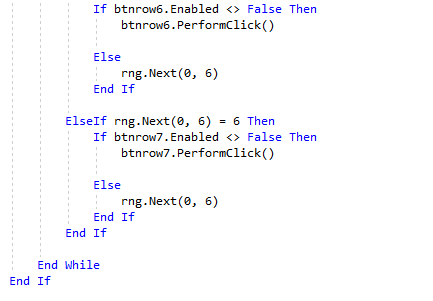


This uses the same method as the horizontal and vertical checker instead of adding 1,2 and 3 to the X or Y axis it does both as the right diagonal check will increment by one the I and j values which represents the x and y axis of the connect4 grid. The diagonal left will start at column 4 as it is the first column that can win using a left diagonal going through each column.

## Artificial intelligence

### Simplest Artificial intelligence

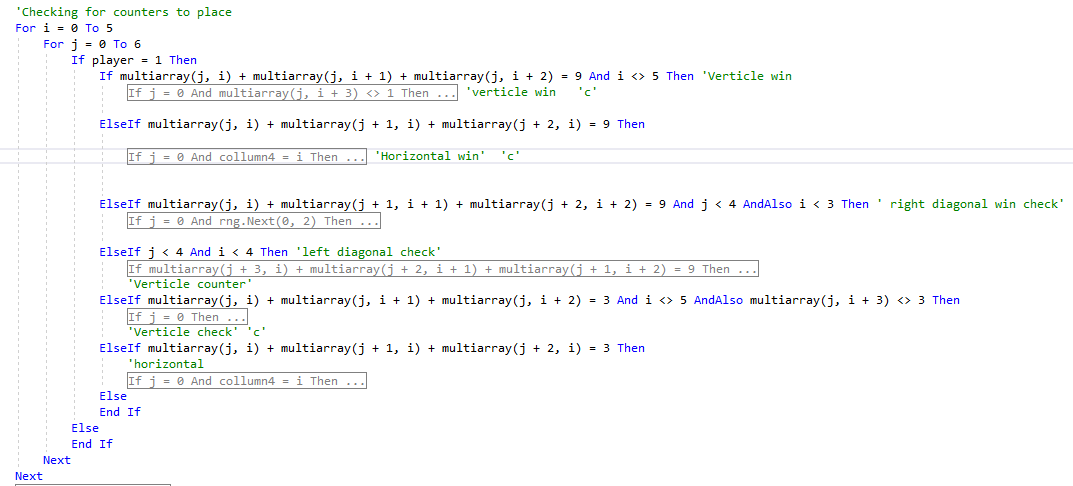




This is how the AI operates by randomly preforming the click function which allows it to drop a counter in the randomly generated column. This AI will be the first AI the user encounters showing it as a simple AI that doesn’t strategically place its counters using the line ***‘rng.next(0,6)’*** to provide a random integer between 1-7 (0-6 includes 0) representing the columns on the connect 4 board ,checking also if the button on the board is disabled if so then there are no other free slots where a counter can be dropped.

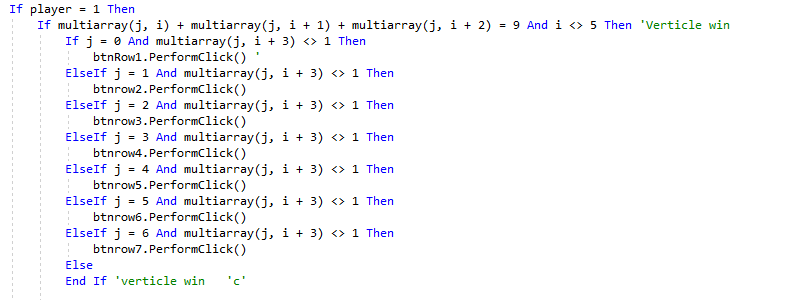
### Medium Artificial intelligence

### Prototype one

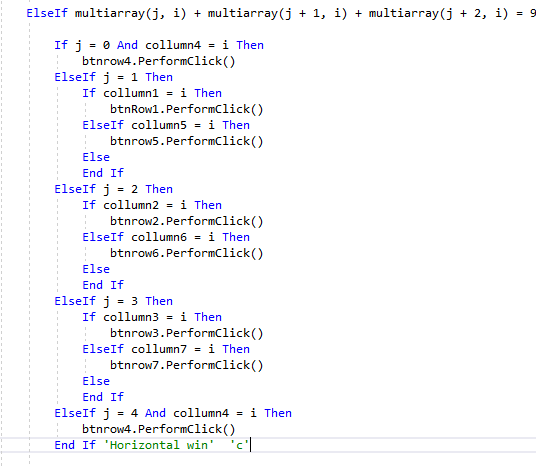


In this version the artificial intelligence checks were all under one for loop which made it difficult to isolate any issues I had with the algorithm. I tried to label each of the if statements to be more effectively white box testing the artificial intelligence while simulating it on the game testing if each of the functions work. This algorithm is supposed to be an upgrade from the simple artificial intelligence being able to do more complex moves and pose more a challenge to the player.

Vertical win check

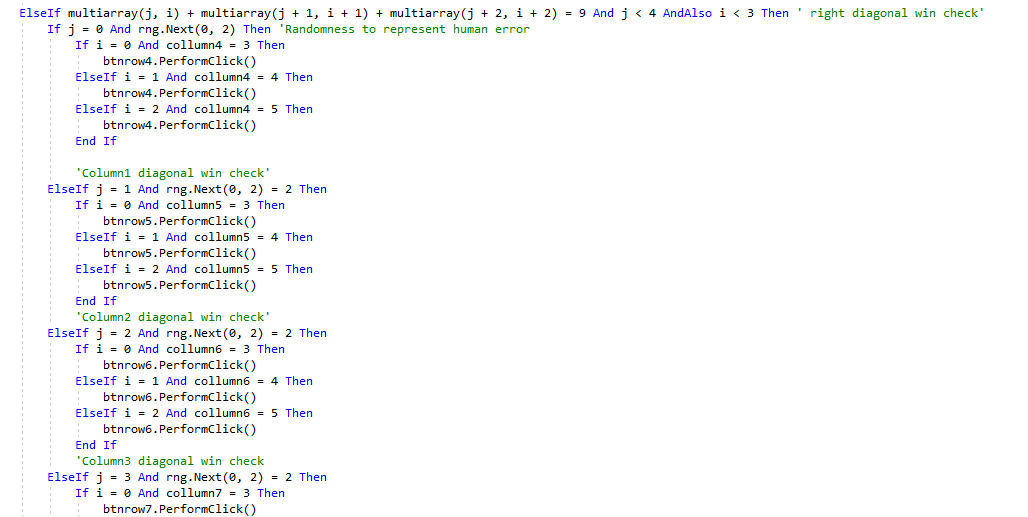


This algorithm is checking the vertical win with by using a for loop shown in the image above to cycle through the entire table and manipulating so that it looks at only the values of the original counter (at mulltiarray(i,j)) and the two above it so and calculated what it added up too and if it reached the value set on the yellow counter(Artificial intelligences colour of counter) multiplied by 3 to indicate that one more above it will constitute a win as long as the slot above the 3rd yellow counter is blank checking this by using ‘***multiarray(j,i+3) <> 1’*** i representing the y axis on the connect 4 board using the ***btnrow(...).PerformClick()*** to actually interact with the connect4 board



This algorithm is searching for a horizontal 4 in a row of the same colour using the same method altering the x axis (j value) which is equivalent at a human looking on a row and seeing if the opposition is one away from winning and can next turn. The line ***‘if j = 0 and collmn4 = I then’*** Firstly identifies which column it starts and has 2 yellow counters next to it, using the ***‘collumn4 = I'*** to see if there is a space available where the artificial can achieve a 4 in a row. The variable collumn4 keeps track on the location of the y coordinate of the counter specifically on the 4th column ranging from 0 to 5.

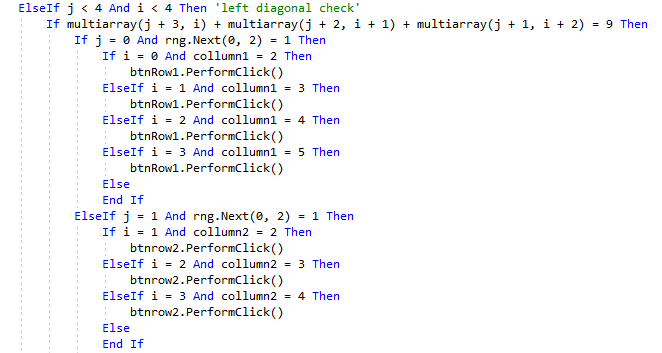
Right diagonal win check

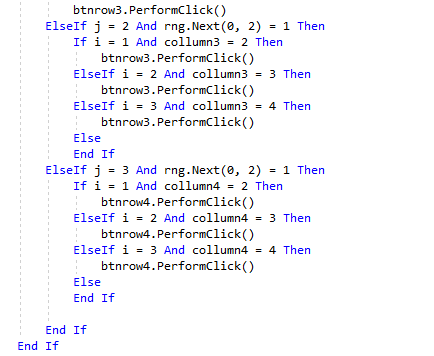




Again, this is for the artificial intelligence to win using the same method used to go through the table instead going up one and right one to select the two counters from the original counter in a right diagonal while making sure that the counter to achieve a 4 in a row is open and the counter won't drop past it. The ***‘ j< 4 andalso I < 3’*** is there to prevent the database going over its applied size. The ***‘Rng.next(0,2) = 2’*** Is there to represent human error like we sometimes are so focused on one side of the board or concentrated on defensive moves that we miss the win.

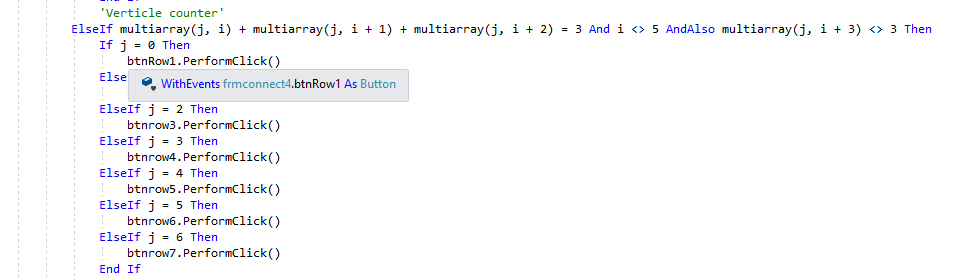
Left diagonal check





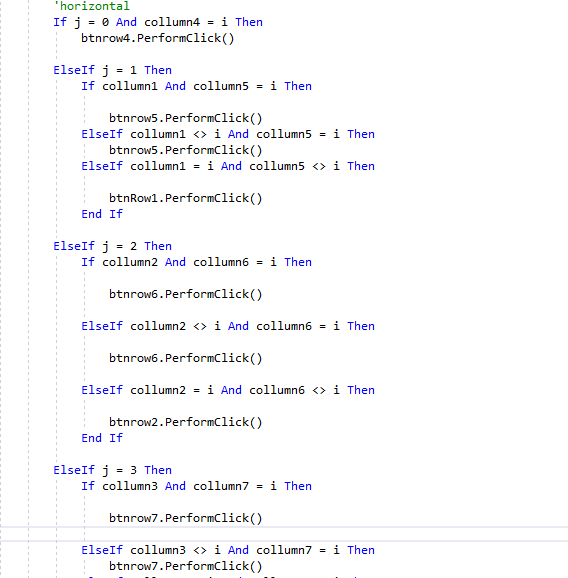
The algorithm ensures that the x and y values are less than 4 adding 3 to x and y values starting from 0 to get the first possible left diagonal win. As always it checks if it can place the counter in the right location in order to get a 4 in a row using the checkers in the game creation to see if there are 4 counters which contains the value e.g. 1 or 3, in this case 3. the ***‘Rng.next(0,2) = 2’*** Is there to simulate human error as I feel diagonal are much more difficult to spot and to constantly recognise than a vertical win.

Vertical check



This scans through the database column at a time upwards to check if 3 of the values is equal to that of 3 red counters while checking that the tile doesn’t already contain a yellow counter not needing to check if it contains a red counter because if it did then the game would’ve been over so it only checks for a yellow. Beneath the pop it says ***‘Elseif j = 1 then // btnrow2.performclick()’***. Which checks which column the top statement is true in checking if it’s in column two then at that point will press column two’s button to drop a counter. No random generation is required as anybody rarely misses a 4 in a row vertically.

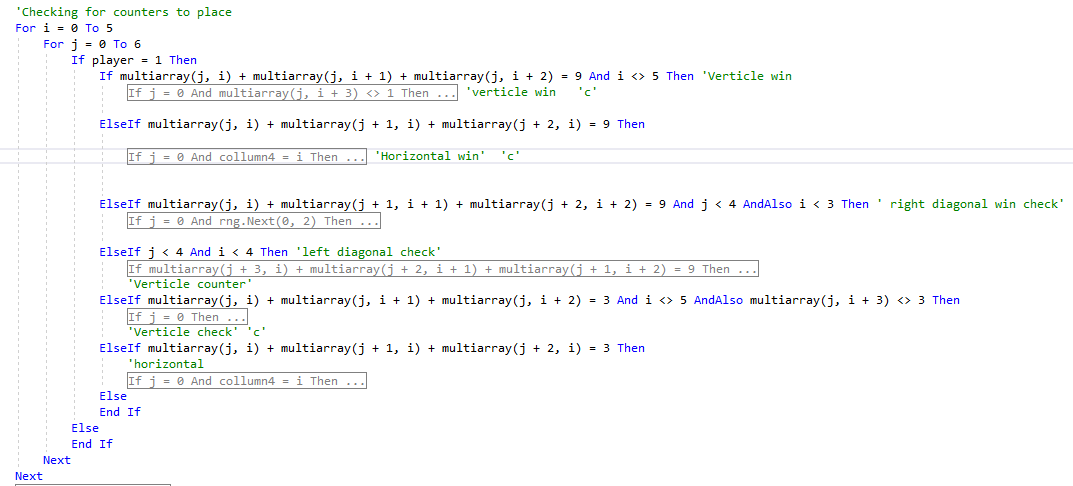
Horizontal



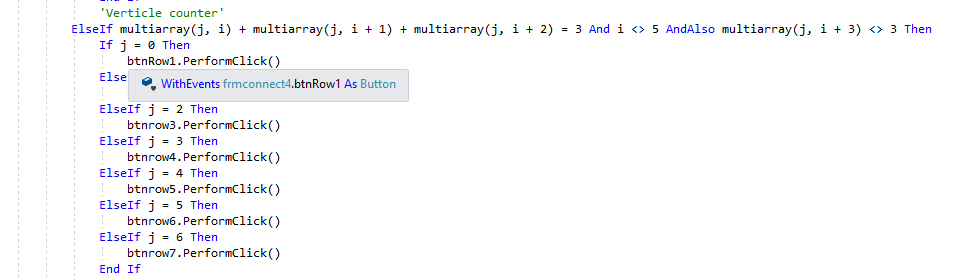


By scanning the rows of the board, it looks for 3 red counters horizontally and them being able to win with one turn. So, the artificial intelligence in this is designed to prevent this win by dropping a counter in that column if it’s the case. If the 3 counters horizontally are the same colour and isn’t in column one then there are two possible ways the player/artificial intelligence can win the one at the end and the one behind so once j > 0 then this will occur until j reaches the end. The line ***‘elseif collumn1 <> I and collumn5 = I then // (<--this meaning a space below) btnrow5.performclick’*** performs this action allowing the artificial intelligence to see ahead and behind to check if they create a 4 in a row.

### Errors faced



Initially for the first draft of the medium artificial intelligence It didn’t contain a ***‘if player = 1 then’*** statement which is vital in having the artificial intelligence only preforming actions when it was their turn. I found this problem by a combination of white box testing and black box testing, using these methods to find out that the artificial intelligence would move multiple times a button click (as the artificial intelligence only acts when button is pressed) meaning it would go through the algorithm and whenever it would hit a button it would then activate that artificial intelligence, seeing only the end product of half the connect four board filled up only stopping when there was a victor. I solved this problem by adding that statement in as you can see above beginning at the very start of the for loop so that during the loop if a button is pressed then it wouldn’t keep going through the algorithm in the for loop searching for buttons to press only stopping when the artificial intelligence won. So adding the statement stopped the artificial intelligence from constantly having its turn.

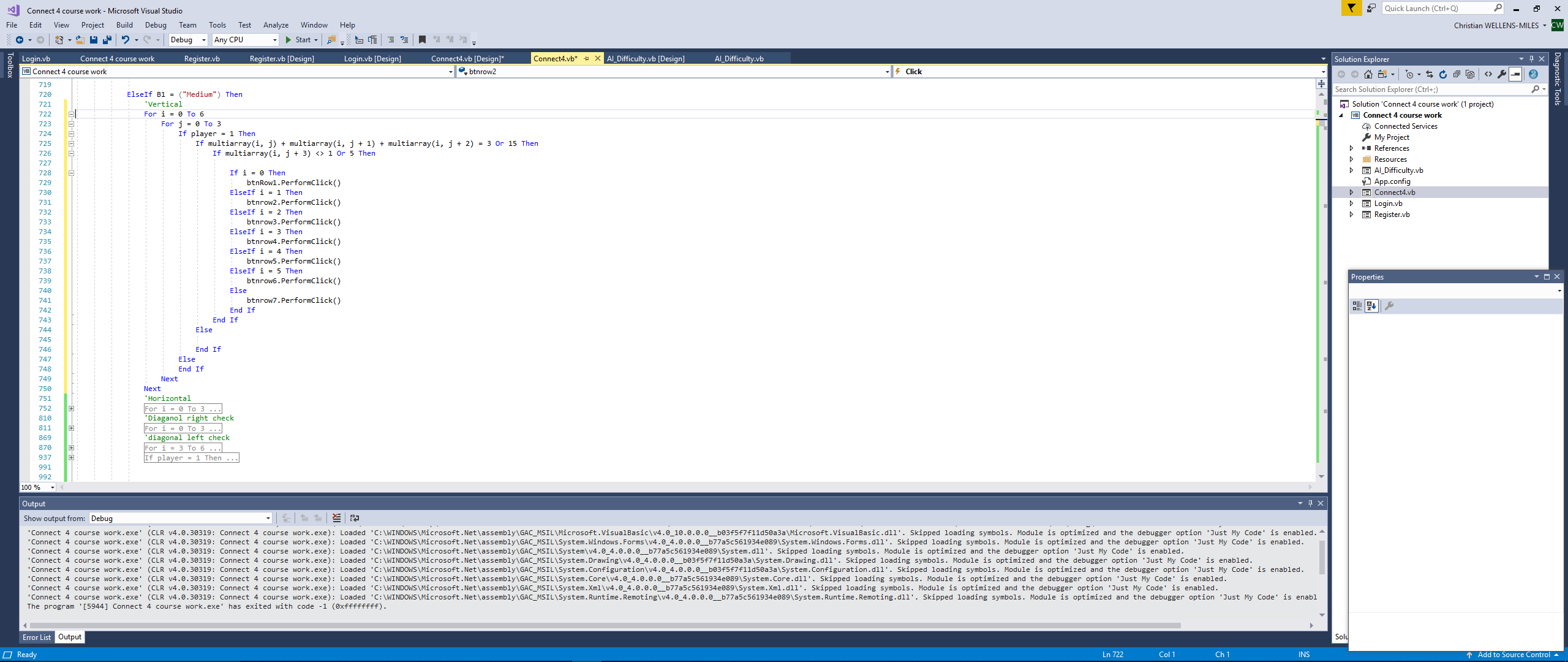


This statement couldn’t properly function the way it was intended as the two values set for the counters were overlapping with each other as 3 was equal to 3 red counters so when checking for 3 red counters if there was one yellow counter and nothing else in a column the code would execute anything below the statement leading to incorrect counter drops. Solution that I came up with was to merely change the value of the yellow counter making it to a value outside of the range of the red counters.

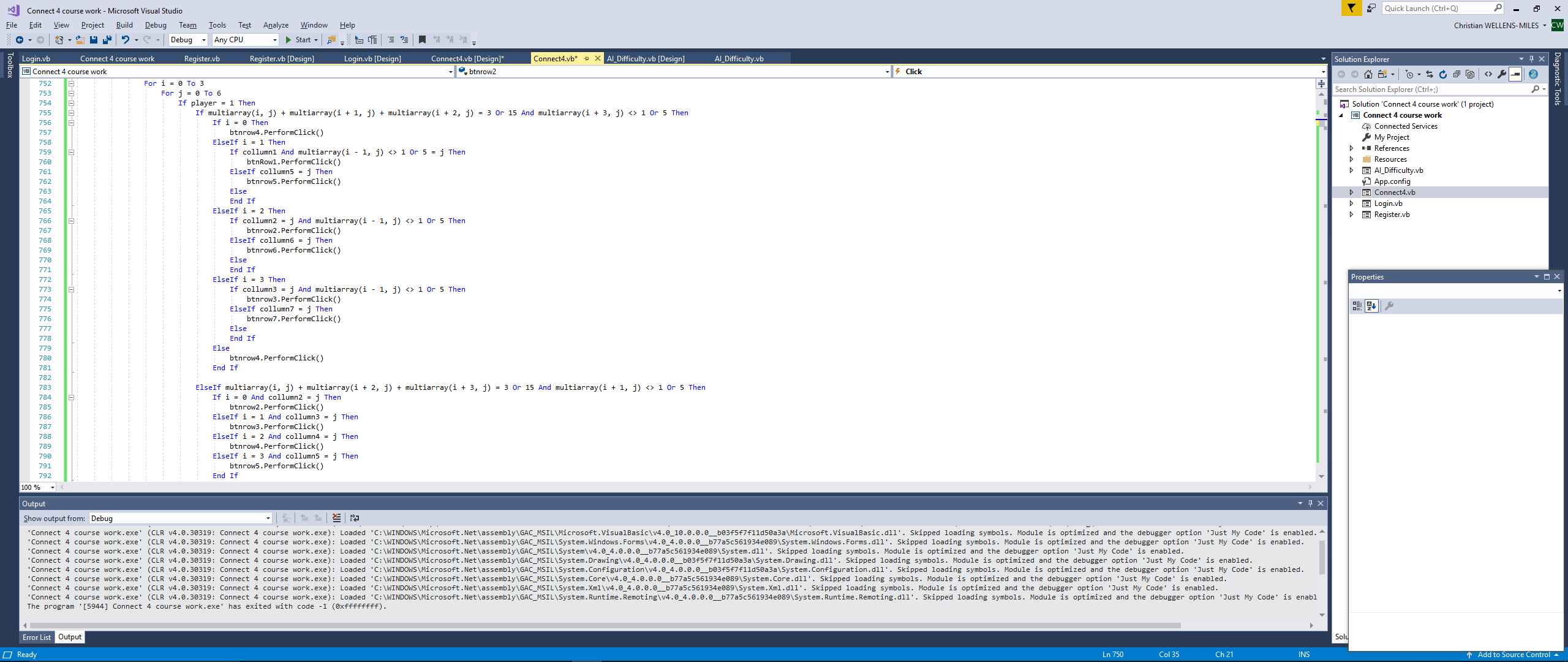


This Is placing the values of the counters into the virtual connect 4 board

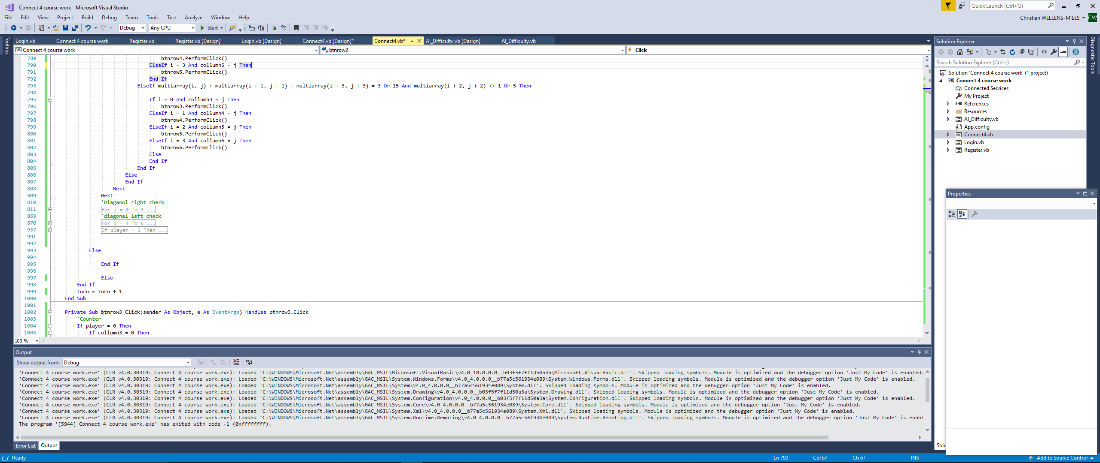
### Prototype 2

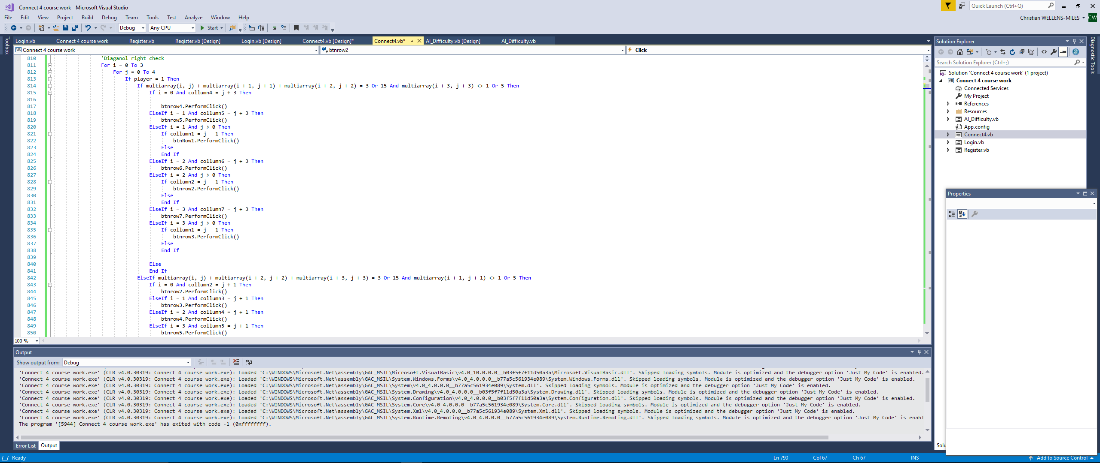


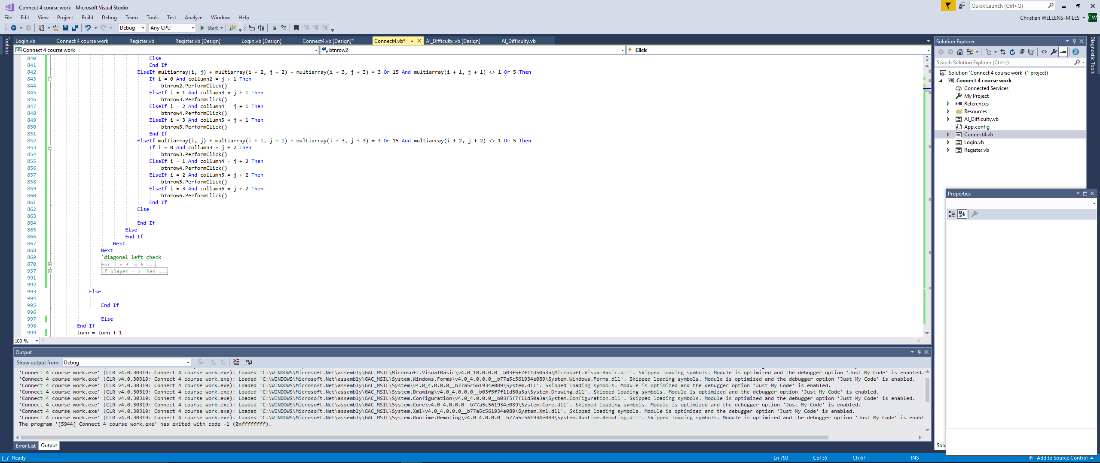
This line of code is supposed to be able to scan through the array where the values for each counter is held, creating a virtual connect 4 board instead with values and not counters. This 2d array table will be how the artificial intelligence will navigate the board and decide where a suitable move is.  
The ***‘If player 1 Then’*** line of code is to prevent the artificial intelligence constantly dropping counters. This line of code is supposed to scan firstly upwards to the top of the board using a for loop to go through the database for movements the line:  
 ***‘Multiarray(j,i) + Multiarray(j,i+1) + Multiarray(j,i+2) = 3 or 15 Then’*** with the multiarray being the array and the (j,I) acting as the x and y coordinates of the virtual connect 4 board. The line adds the values stored in the selected locations all being different location due to the multiarray y coordinate (or i) being incremented each time so it selects the base and the two other slots above it and if then it equals 3 or 15 (as each multiarray is set as 1 or 5 if a counter is in it) then depending on the x coordinate (i) it will drop the counter accordingly so if i = 1 then it drops the counter in column1.

Horizontal 

This algorithm does a similar function to what the vertical check for the artificial intelligence does. The for loop is to cycle through the virtual connect4 board of set values. Ensuring that the value ***‘Player’*** which is a value which alternates depending on whether the previous counter dropped was yellow or red. The ***‘multiarray’***  is used to select the counters on one row at any point in a column to check if the value of the 3 selected slots have either 3 red counters or 3 yellow counters so it can react by dropping a counter in the relevant location. To do this the line ***‘multiarray(i+3,j) <> 1 or 5’***   
this ensures the value of whatever the value is of the I the button to drop the counter depending if the value of that column which stores the how high the counter is on each column is viable so that the artificial intelligence can achieve a 4 in a row to win the game and does this with the middle counters missing too which is represented by the ***‘Elseif multiarray(I,j) + multiarray(I+2,j) + multiarray(I+3,j) = 3 or 15’*** this line enables the artificial intelligence to check other ways in which it can get a horizontal win always ensuring that the conditions for the win are all correct.

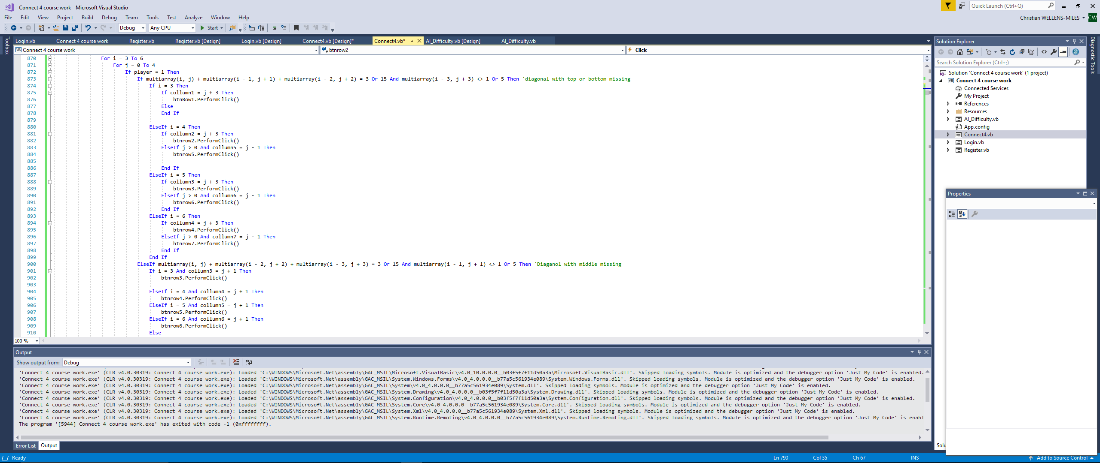


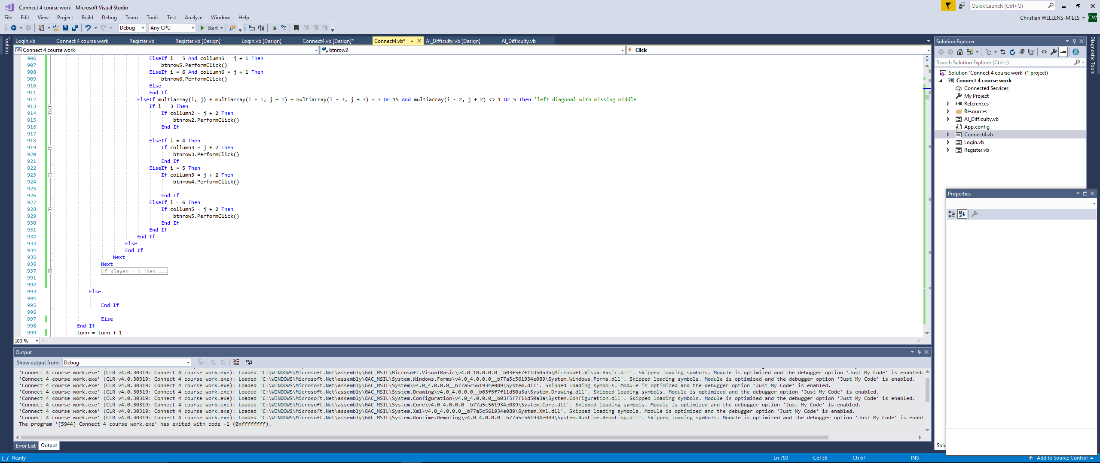




Did the exact same search method as the other artificial intelligence except this time it searched for both red and yellow counters going diagonal right by 3 so it can prevent it and win using the same for statement. The ***‘multiarray(I+1,j+1 <> 1 or 5’*** checked if the slot to either prevent the 4 in a row or win the game was clear then further down checking if ***‘column(x) (x being a value between 1-7) = I’*** which determines if the artificial intelligence presses the button if it would win or prevent the player winning the game if it didn’t then it would skip right to the end instantaneously.

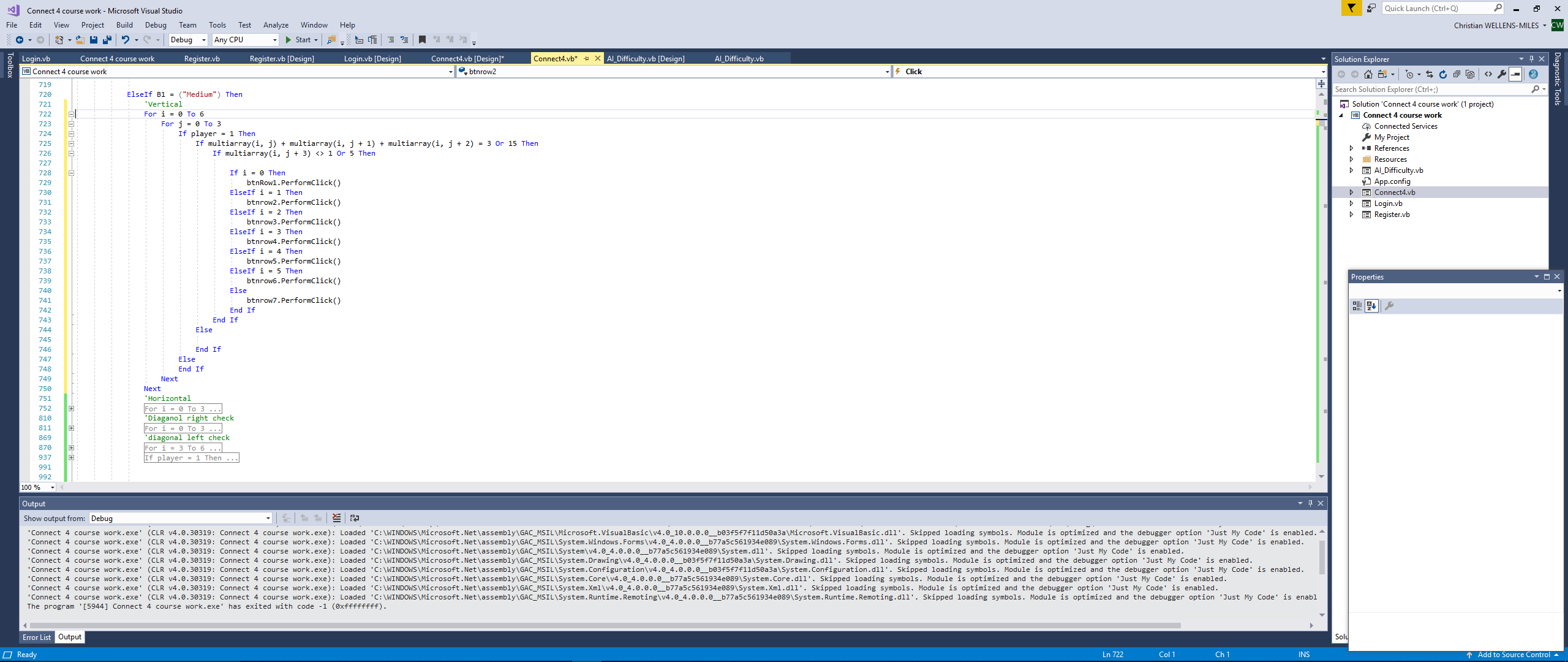
Diagonal left



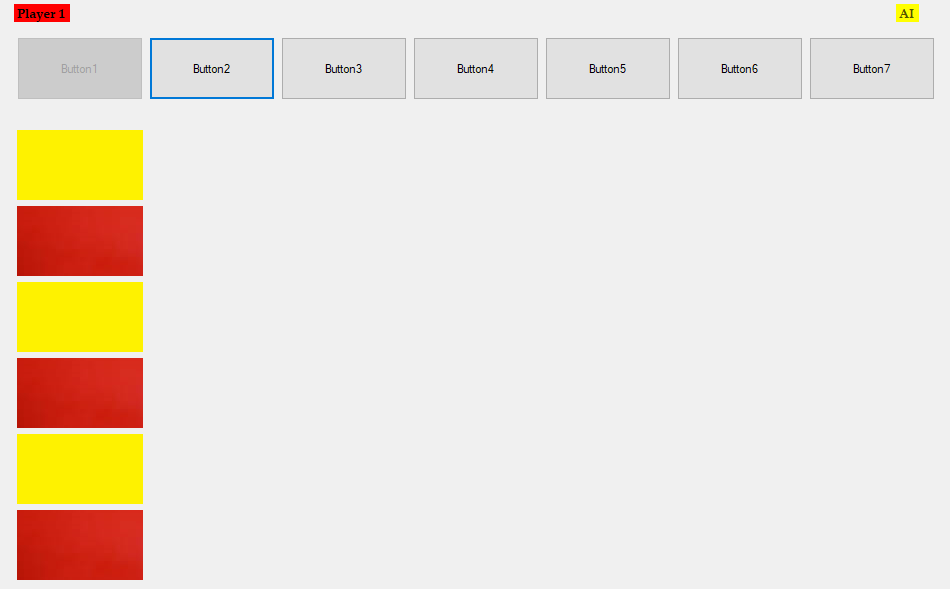


Same method different numbers entered within the multiarray to get the pattern (left diagonal) I needed. The for statement for i (x) started at 3 in the array or collumn4 as the conditions in getting a 4 in a row left diagonally is impossible before that point. Instead of adding one on the I and j values it would need to take 1 off the I value as then it will be checking one left and up one each time to check if the board has any counters of the same value again having it search for both red and yellow counters to make it more efficient.

### Errors with this code



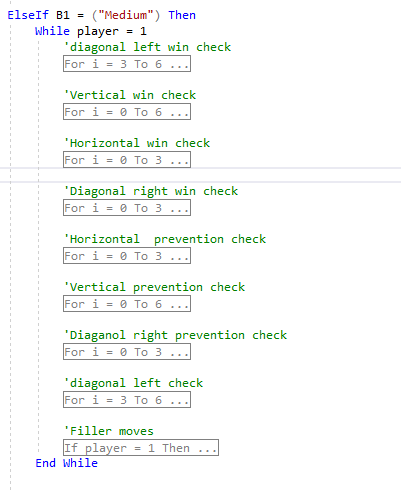
for some reason fails to do this function which has confused me as it always executes this line no matter what the values are in the array causing the artificial intelligence to constantly drop its counter in the first column until it can't anymore. I found it by firstly scanning through the code to check if any of it was incorrect or the wrong function, then added a message box as an indication of where the program was pressing the button adding it underneath the statement ‘elseif i = 0 then’ which each time showed me the message indicating to me when I executed the artificial intelligence that it was for some reason ignoring the ***‘if multiarray(I,j)+multiarray(I,j+1)+multiarray(I,j+2) = 3 or 15 then’*** so I found that this line was the issue.



As shown above the artificial intelligence was never endlessly going down the line of code not reaching the rest of the code, creating multiple issues. The solution for this problem was to separate the code from doing both the red counter and yellow counter as I had done before in the win checker where it scanned the board to see if there was a 4 in a row separating it into either winning moves or prevention moves each having its own for loop to keep it easier to manage.

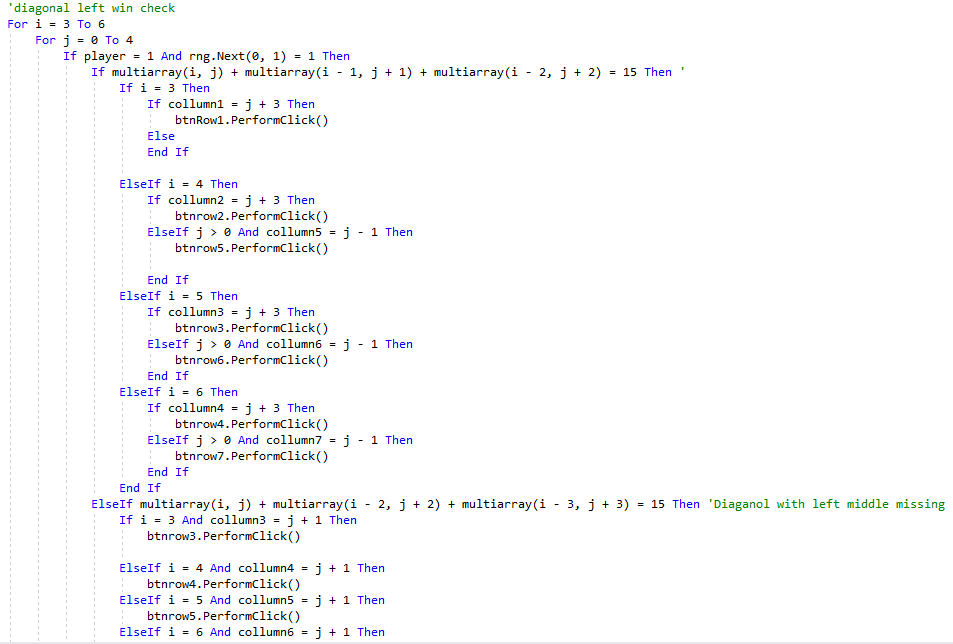
## Finished Medium artificial intelligence

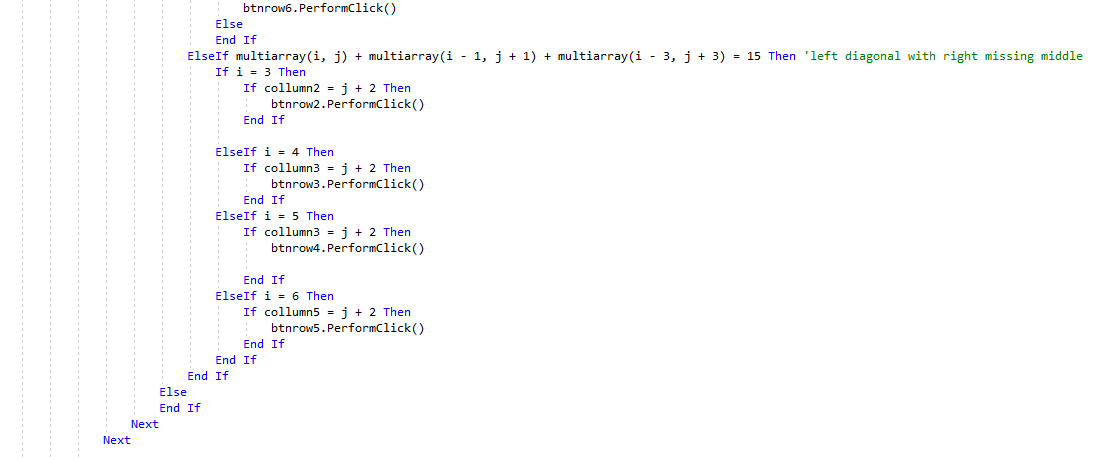
Medium difficulty AI build



The win checks in the code will be closest to the top as it’s the most logical thing to do, as you would rather place a counter to win the game rather than to stop the player from winning. I've separated the for loops for each check being winning moves and prevention moves. This allows me to easily alter or test any of the code as its clear to see where everything is.

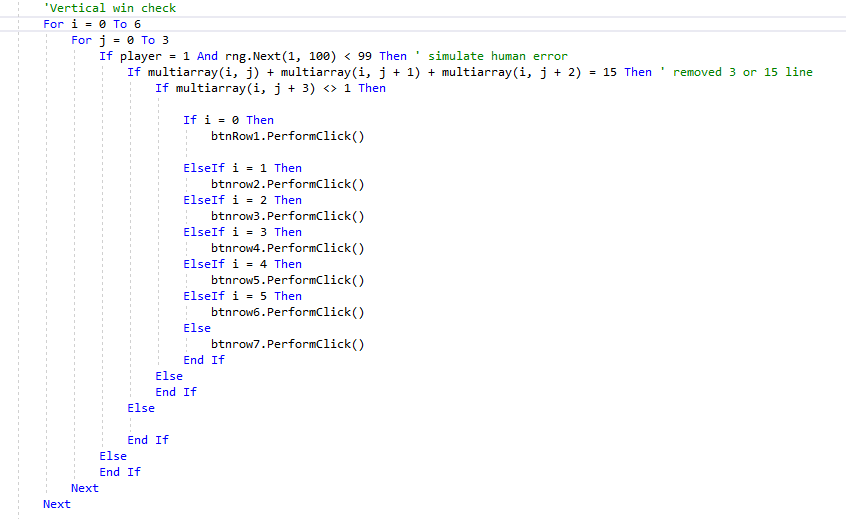
Left diagonal AI win





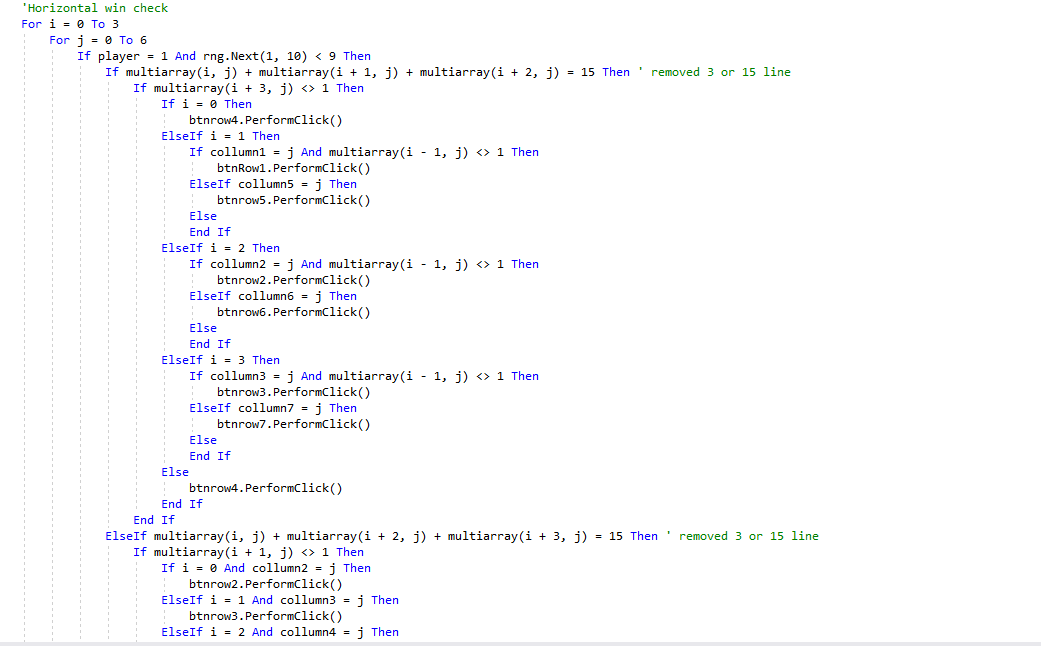
The left diagonal win initial for loop, i starts at 3 or column4 since it’s the first point on the board where it’s possible to get a 4 in a row by left diagonal. The ***‘rng.next’*** adds a bit of randomness, simulating human error as it generates a value between the numbers within the bracket and whether it meets the criteria it will either carry through with the code below or skip it. The first if statement for multiarray is the generic 3 in a line starting from the origin point taking one off the I value or the x value of the board while adding one to the j value or the y value increasing it by one to get a diagonal line going to the left. Once the I value is greater than 3 then the possibility of winning using that 3 once j is also greater than 1 double as it can also be won using the tile below and to the right of the original counter if there's nothing there. This isn’t the only way you can win with the left diagonal line you could have one of the middle 2 missing while the counter 3 above the origin and 3 to the left is the same, this line represents this ***‘elseif multiarray(I,j)+multiarray(I-2,j+2)+multiarray(I-3,j+3) = 15’*** checking to win with the left diagonal with the middle left missing and ***‘elseif multiarray(I,j)+multiarray(I-1,j+1)+multiarray(I-3,j+3) = 15’*** checking to win with the middle right missing for left diagonal. It does this for each column it can possibly win this way. While constantly checking that there is no counter in the winning position and that when the counter drops in the column where it’s possible to win it will always land so it does.

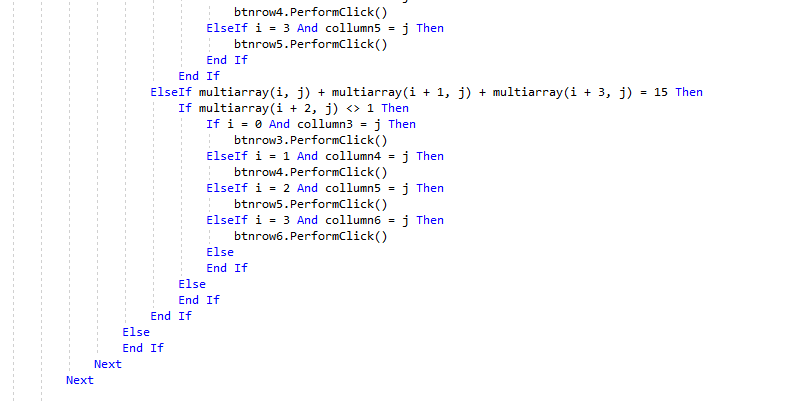
Vertical AI win



Cycles through the entire board firstly going upwards in one column to see if its possible for the artificial to win and if the value of player = 1 and also the rng.next(1,100) value is less then 99 then it will search to see if the 2 counters above it meet the value of the yellow counter multiplied by 3, if so then it will place a counter on top of it as long as the location isn’t occupied by a red counter

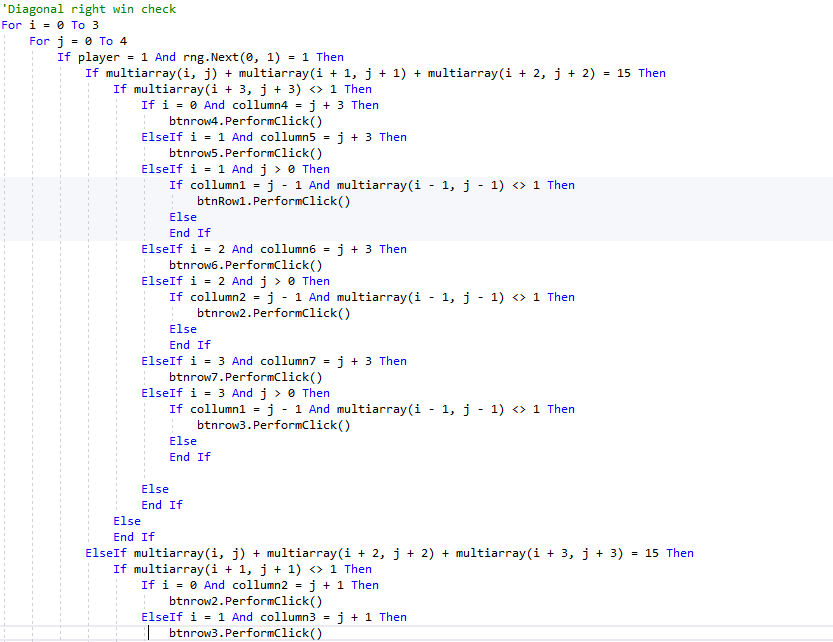
Horizontal AI win

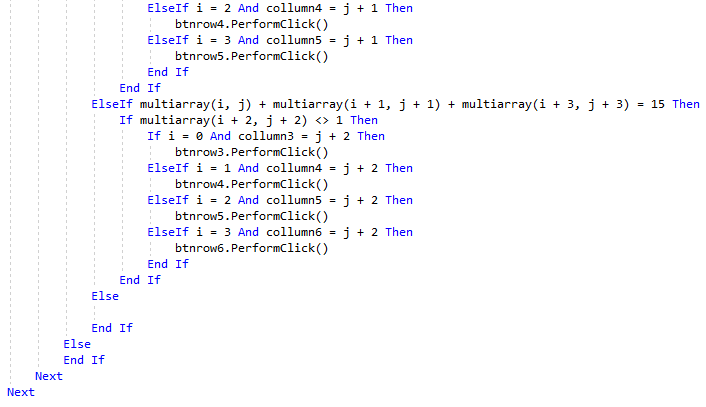




This scans the entire board firstly horizontally to check if there is a possible way for the artificial intelligence to win on that row then moves up a row and does the same by initially looking at 2 counter right of the original point on the board which is multiarray(I,j).Then it looks at the tile behind the original point when applicable to see if there is an open slot there where it can immediately place a counter to win. It then searches for any incomplete 3 in a row like when one yellow counter is on its own on a row then two are together two slots away checking if the gap between the pair and the singular counter is free and whether or not it is a win condition if they place a counter there doing this twice as you can have a pair of yellow counters on the left and one on the right and the reverse way round hence what the last two elseif multiarray(…...)….... statements do.

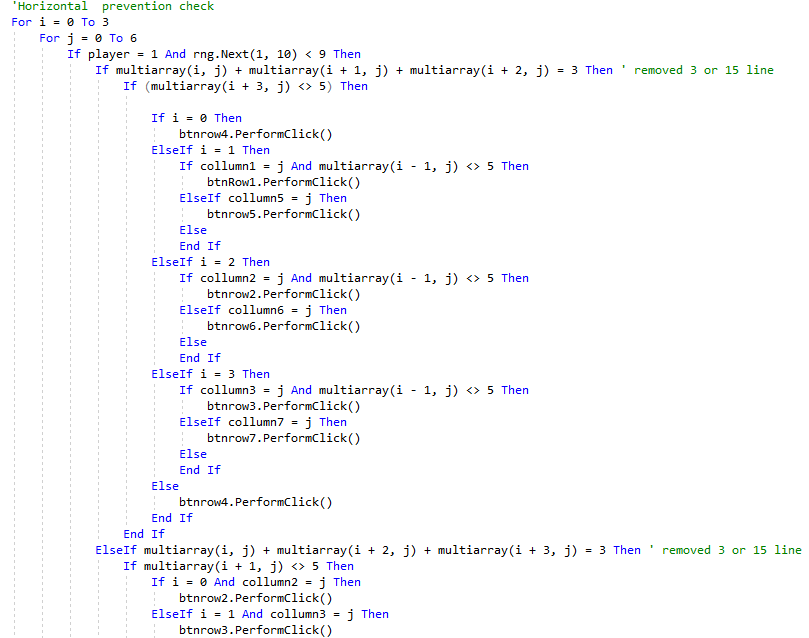
Right diagonal AI win

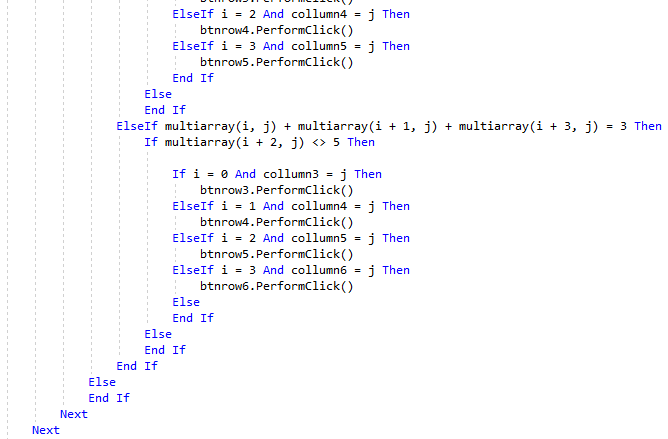




The diagonal right is similar to the diagonal left win check however flipped on the j axis having I be added by a positive one to represent one to the right on the connect4 board. Like in the left diagonal win check it will check firstly as if the counters was in a straight diagonal line having one counter after the other which is represented by this line ‘multiarray(I,j) + multiarray(I+1,j+1) + multiarray(I+2,j+2) = 15’. Then it goes on to finding the one behind it when the original point column is greater than 3 and is greater than row 1 represented by ***‘If collumn1 = j-1 and multiarray(I-1,j-1) then//btnrow1.performclick***’ doing this for each value of I which is greater than 0. After this the artificial intelligence then checks the right diagonal line while having the original point and the counter 3 rows above and 3 to the right on the connect 4 board alternating between each line which middle tile is equal to yellows counters value and if the other middle piece is free and there’s a counter below it then it will drop a counter there.

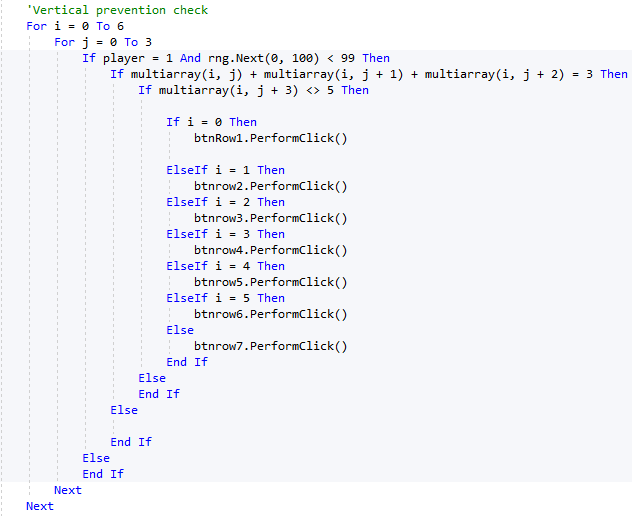
Horizontal prevention check



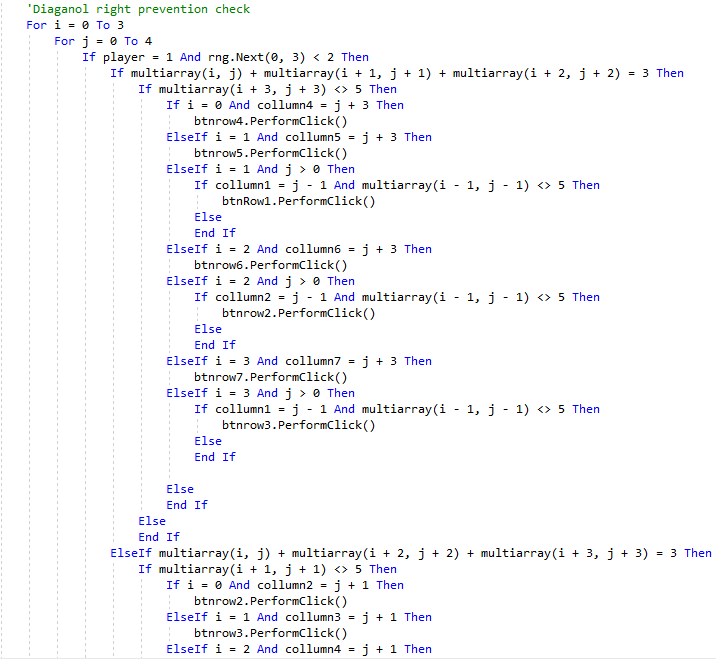


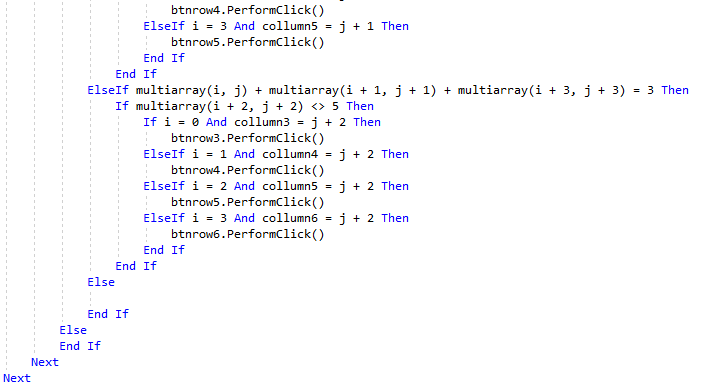
This does the exact same thing as the win check for this artificial intelligence instead the value for the three counters going horizontally in different ways is equal to the value of the red counter multiplied by 3. shown by ***‘multiarray(I,j)+multiarray(I+1,j)+multiarray(I+2,j) = 3 then’***

Vertical prevention check



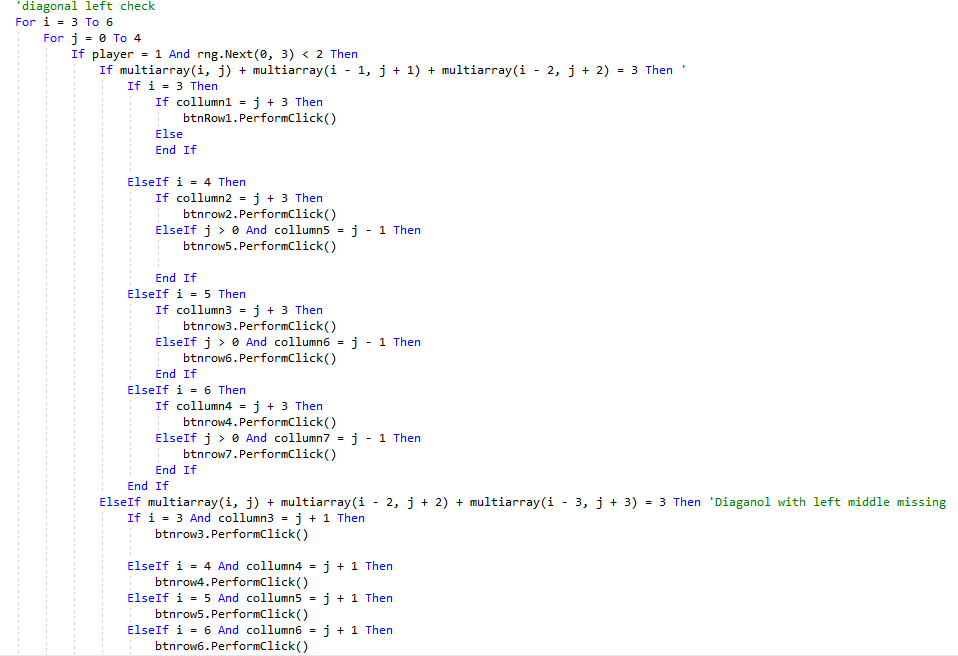
Does the same process as the vertical win check only that the value of counters its looking for. Instead of it being the 3 yellow counters in a column simultaneously its searching for 3 red counters having a ***rng.next*** to simulate the artificial intelligence slipping up.

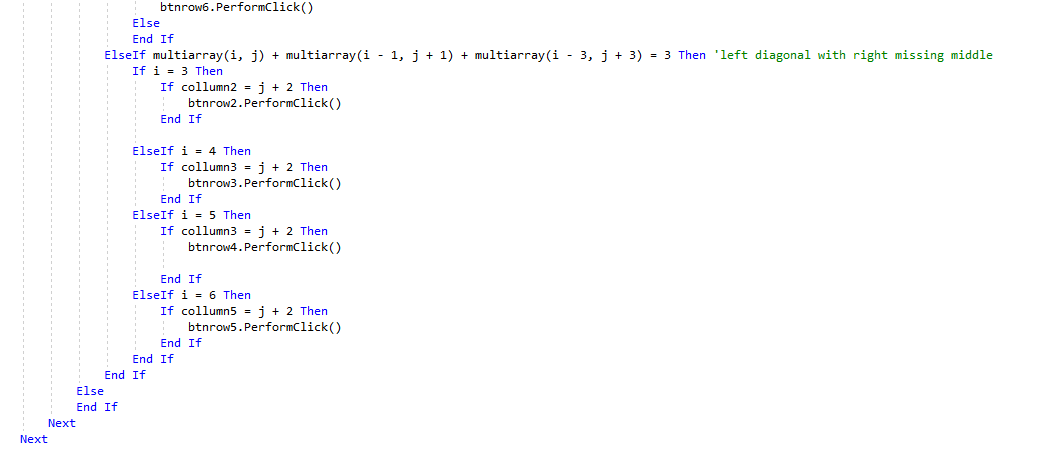




Does the same function but again the value all the multiarray adds up to is equal to 3 which is the value of the red counter multiplied by 3.

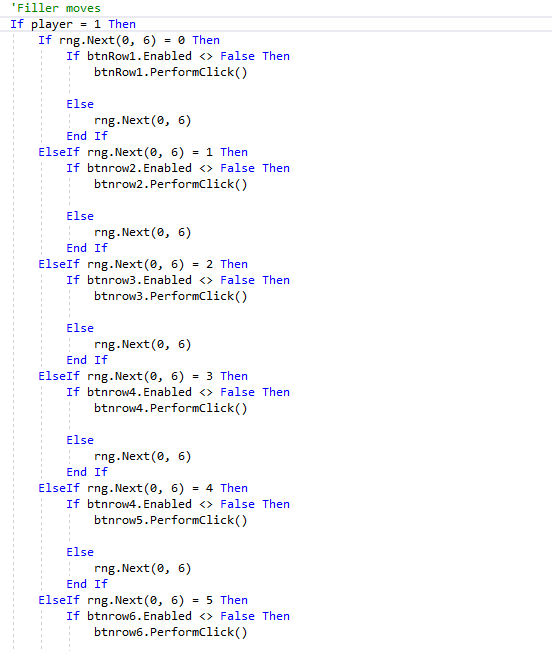
Diagonal left prevention check

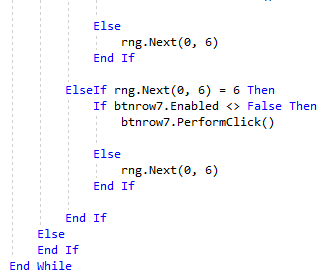




Does the same function as the left win check instead replaces the value of multiarray+..... = 15 with the value of 3 which represents 3 red counters on the connect 4 board.

Filler moves



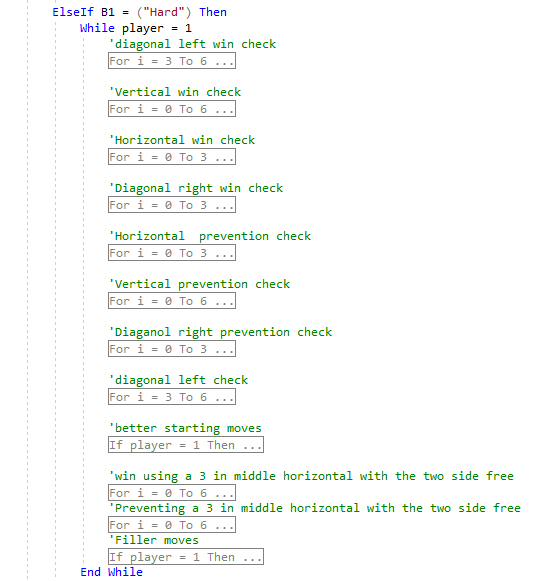


If the artificial intelligence has no moves then it will, like the simple artificial intelligence, will randomly select a row depending on what ***rng.nex(0,6)*** equals if its 0 the column1 will drop a counter and so on using a while loop in case of there being a button it cant press because of the column being entirely filled with counters then it will end the if statement generate a new random integer until it can place a counter and if it cant it will end as a draw.

### Errors

The errors I faced while creating this difficulty have mostly been listed above the only see able error from the final version is that the ***‘rng.next’*** generator is within the for loop which means for each loop it generates a new number which makes it a greater chance for the user to win and for the artificial intelligence not to. I found this error by doing the white box testing looking through the code seeing how it should logically function however for the other errors I had to use a mixture of the two testing methods as shown on the other variance of artificial intelligence it looked to be correctly coded however it didn’t function while playing the game the way it was intended.Solution for the random number generator was easy all I did was remove it from within the for loop and put it on the outside.

## Hard artificial intelligence



It builds on top of the artificial intelligence of the medium artificial intelligence having more function such as better start up moves and attempting a 5-piece win. So will only show what is new.

The Random number generator has a lower chance to fire which means that the player will have a near perfect player which sees all the counters and all the wins.

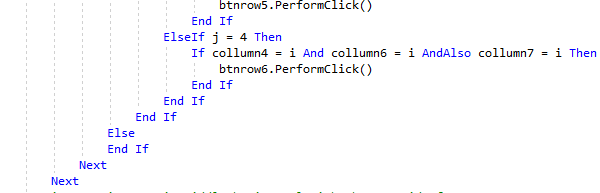
Better starting moves



The tiles in the centre of the board and around it are the most vital components in winning a game as it allows multiple various ways of winning so this code checks if the middle of the board has been interacted with from the player and capitalises on it.

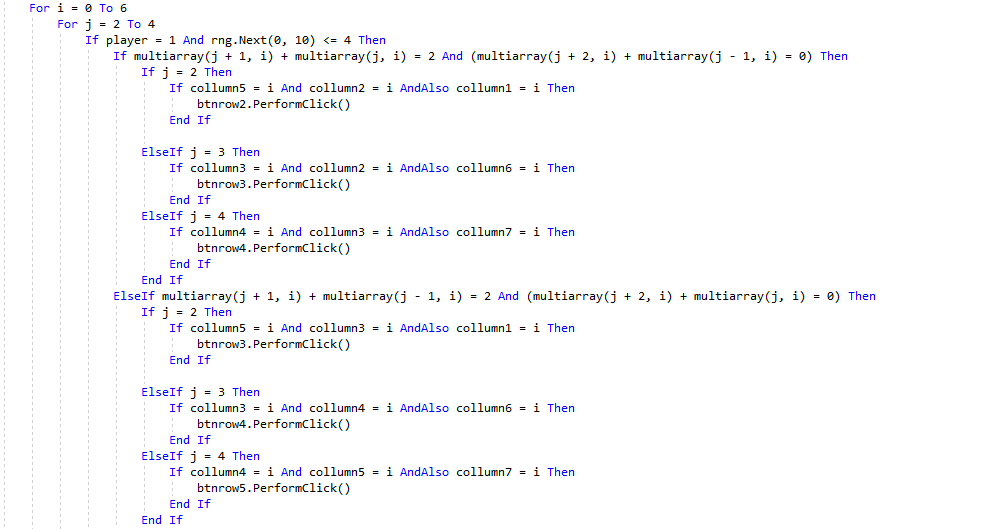
Winning with a 5 in a row

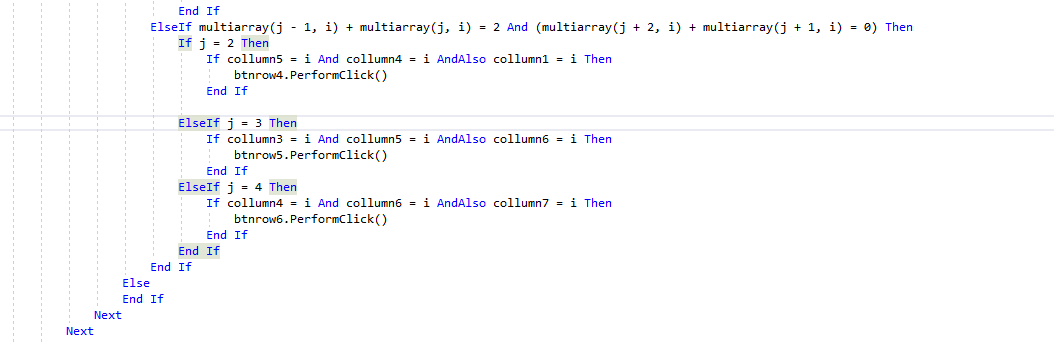




Its similar to the horizontal win algorithm only that it will look at two counters of the same colour, for this yellow or to see if the 4 counters around the two are free horizontally or whether there is a possibility to end the game right there and then by having two possible ways to win with the player only having one move securing the artificial intelligence the victory.

Countering 5 in a row

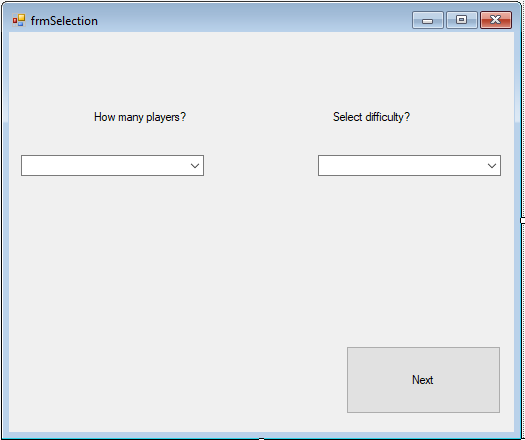




Checks if two red counters have 4 or 3 free spaces on that row to make it possible for the artificial intelligence to respond to the issue before it loses

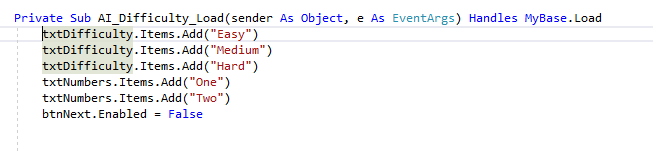
## Difficulty form

### Design



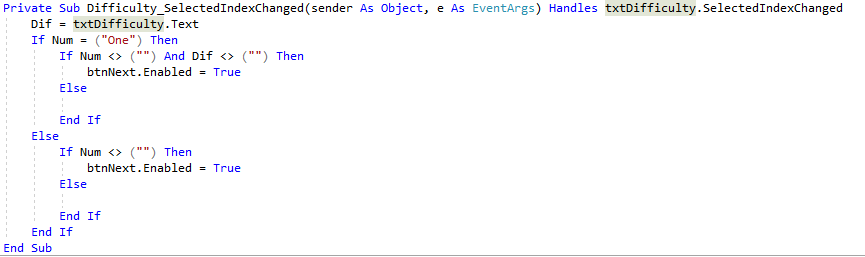
Simple design to input detail so that it can be used by the connect 4 board like difficulty of artificial intelligence and how many players will be active

### Creation



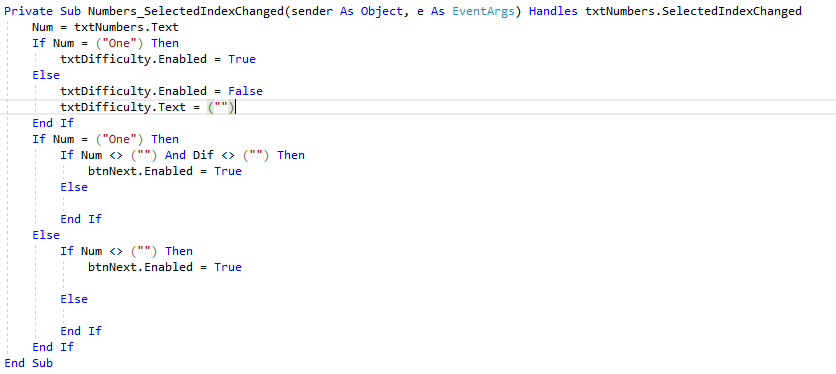
Adds text to the text boxes on the form. The next button starts disabled as to prevent the user from skipping entering the details necessary to run the connect 4 game.

Difficulty choice



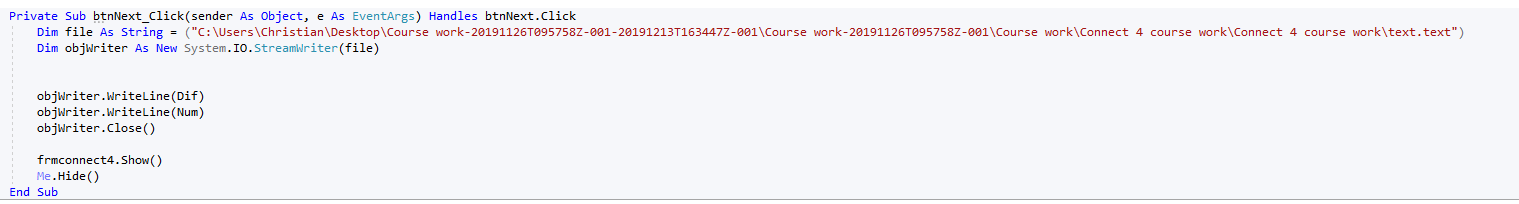
If statement to check if the information has been selected and if they match up, then they will enable the next button bringing them onto the game. This algorithm checks to see if there's text within the text box using ***‘if num <> (“”) and dif <> (“”)’*** to check this.

Number of players choice



This does the same thing except if there is more than 1 player then there can't have artificial intelligence implemented within the game so then the difficulty text box is disabled preventing the user from adding any text that will then write to the text file where the user would then have an issue on the game.

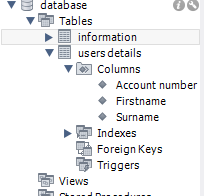
Next button



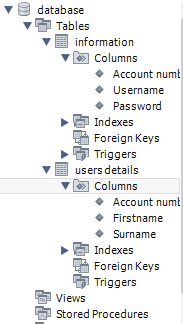
Writes to a file so data form select boxes can be used in the connect 4 form as I couldn’t add a select box where the game is, so I needed to make a new, separate form in order to accomplish this.

## SQL database

### My SQL database



This is an out view of the tables in my SQL database consisting of two tables linked by a foreign key. Information and users details. The information is the data required in the login stage while the users details is the data that the user entered during the register stage.

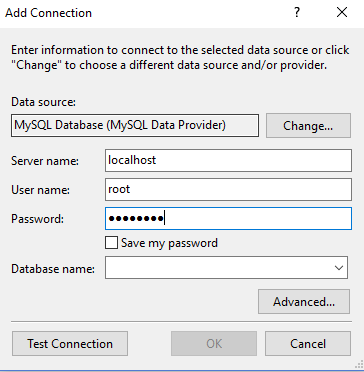


These are the columns the databases have being linked by the account number in each of them. The account numbe is unique and increments with each added column

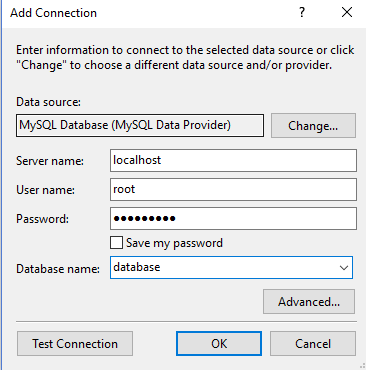


This is showing that the data base with the columns and primary key is functioning as supposed to only requiring one table for most of the information as it relies on it.

### Connecting it with VB



This is the process I took when linking the SQL database to the code.



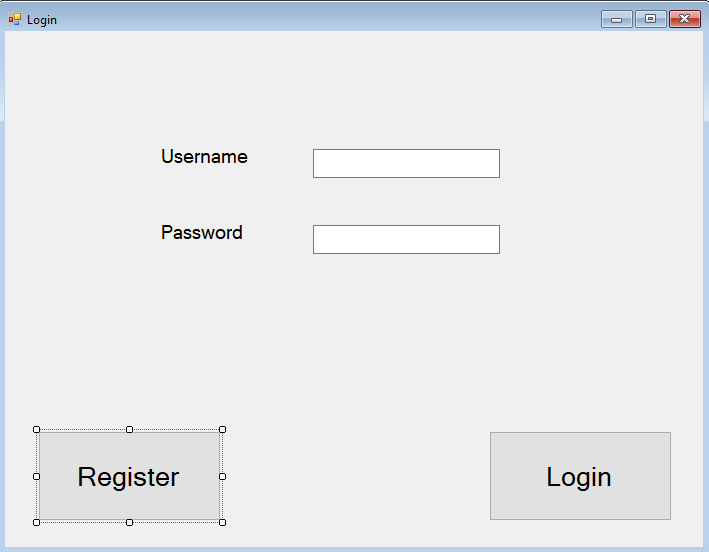
This is showing me select which database I want connecting to my program. So, I selected the database which was constructed on my SQL in order to link the two.

This is the expanded server tab on visual studio showing how the local database showing all the columns in the database

## Login system

### Login form

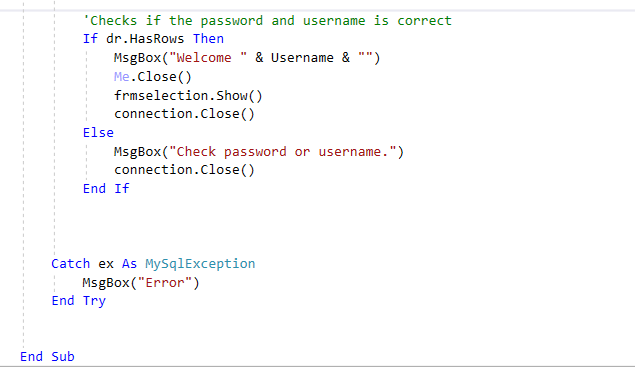
Design



Login code



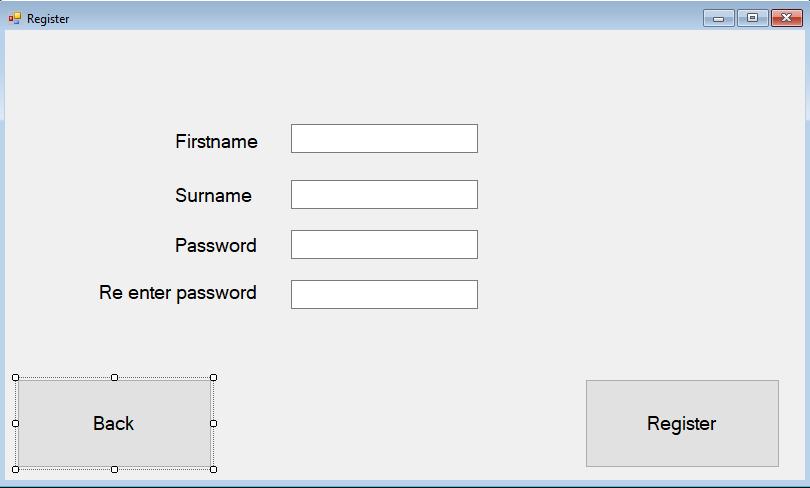
The connection string links the database to this algorithm. The SQL command selects the data from the information table collecting the Username and Password so it can be used in the line below



This line of code is checking whether the username and password is contained within the database if so then the ***Msgbox(“welcome “ & Username & “”)*** line will activate and will go the the selecion form.

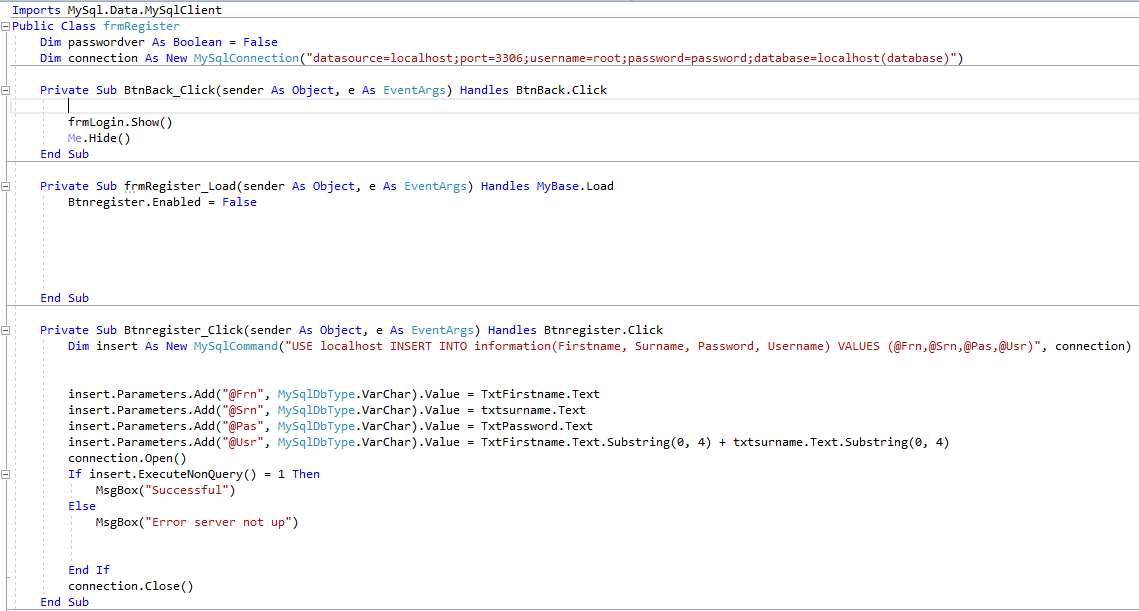
### Register form

Design



Made buttons and text big so its easier for people to read and understand what everything does.

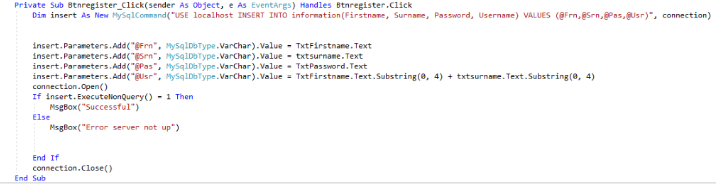
Register code



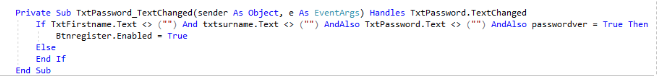
The back button is designed to simply move back to the login form.



Initially this code is used to input the entered details about the user into the SQL database which Is always online. Firstly, the code links the program with MySQL where the database is stored. Ig the button back is pressed then the register form will be hidden, and the login form will be shown again. The register button will be enabled if all the information is inputted correctly.



This line is the SQL query that will be inputted into the database using parameters to designate what each of the first name, surname, Password and username are in context to the form having them equal to the text that they have inputted. Then checking if the connection is linked.



This algorithm is to check if all the text boxes are filled out and making sure that the password, they've entered is the same for both text boxes

### Errors

The database wasn’t updating for some unknown reason from searching far and wide I failed to find a solution using my method so I opted out of this method and did another one

### New Register form

None of the buttons except the register button has changed so I'll only show the changes.



Firstly ,I’m setting variables required to set up a link with the SQL database. Then setting the connection string to enter the password of sorts. The try statement after that is used to probe the database to see if there's a connection, there for testing purposes only.



This if statement checks if the data is entered and if the passwords entered is correct.

The other if statement within it creates the username with fallbacks in case it doesn’t meet the requirements, so it doesn’t break the code.



In this I'm opening the connection with the database so that SQL commands can be implemented. The ***("SELECT IDENT\_CURRENT(`information`)+1 FROM `database`.`information`;")*** line is used to get the next incrementation in the database for the account number and use this and add this to the username so it’s unique.

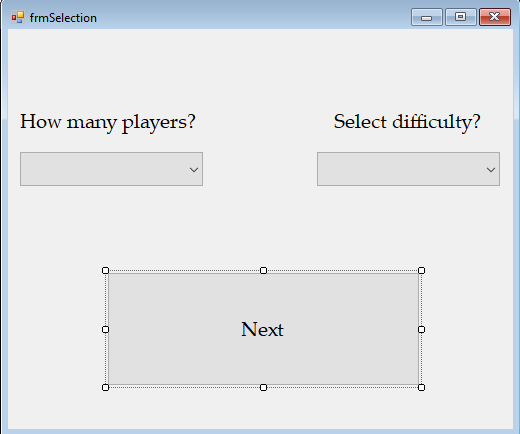
The SQL below this inserts data inputted by the user into the text boxes is used and entered in the database telling the user their username so they can use it in the login section.

## Problems with login system

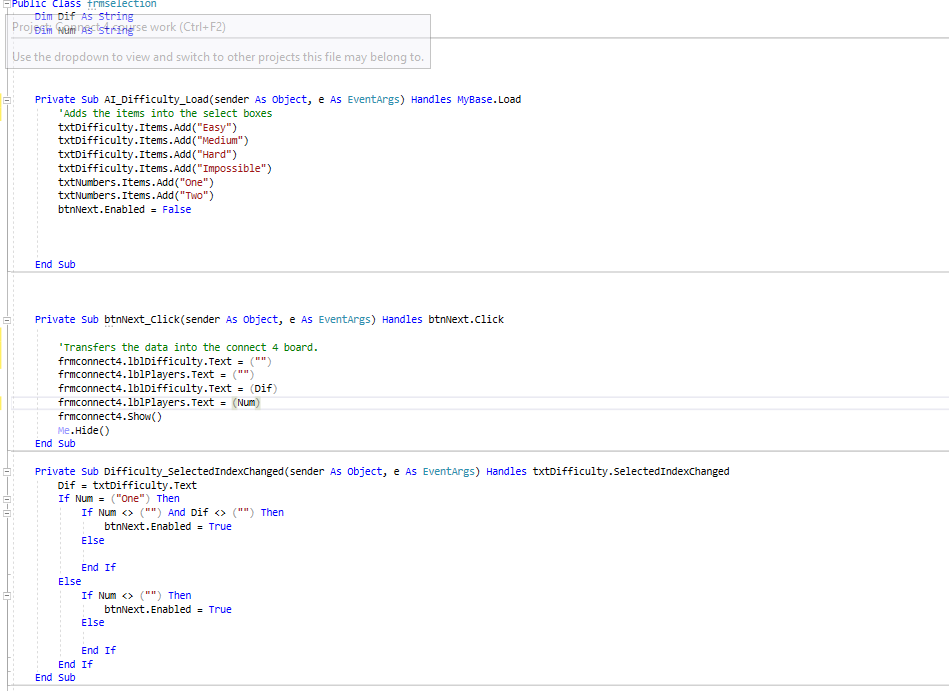
Initially I was planning on using Microsoft access in order to create the database as I'm used to coding in its format. I was coding the sql language in Visual basic and there was no link and no solutions that I found online to fix such programs.

## Final Product

### Difficulty



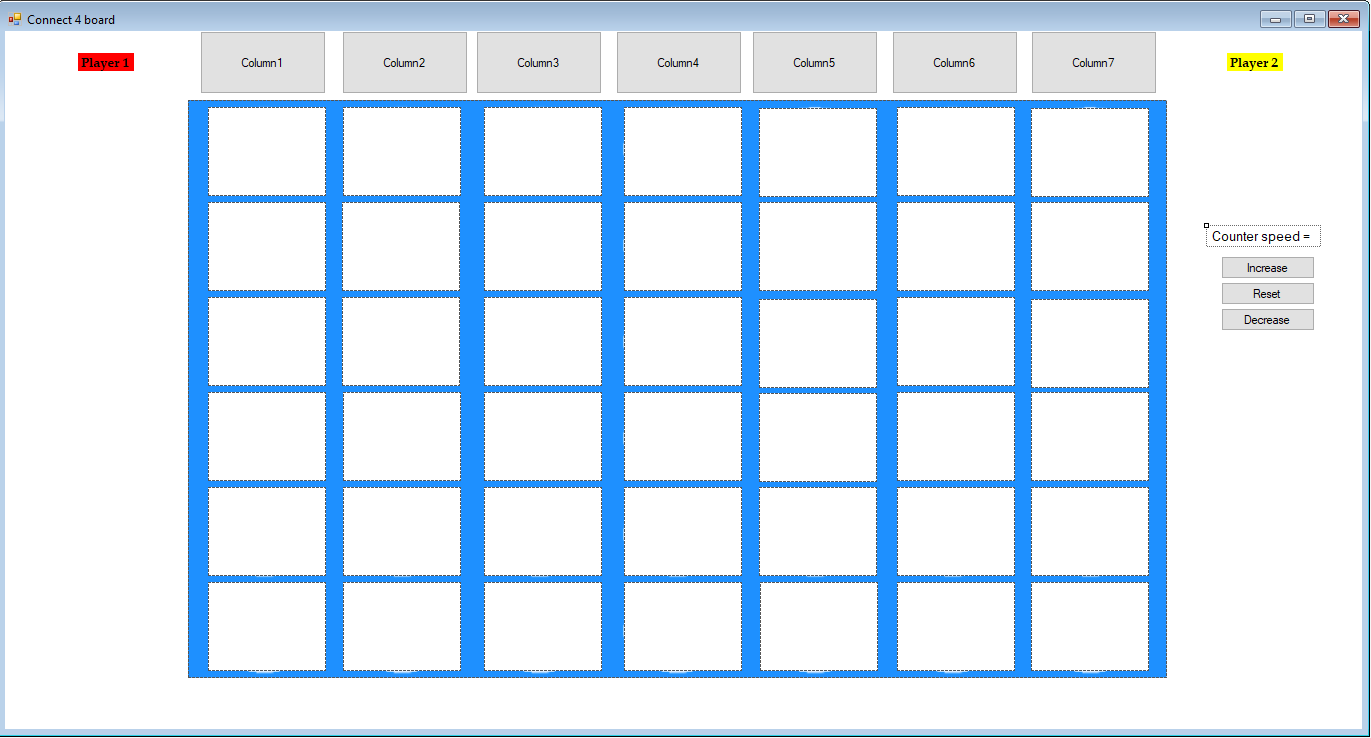
#### Code





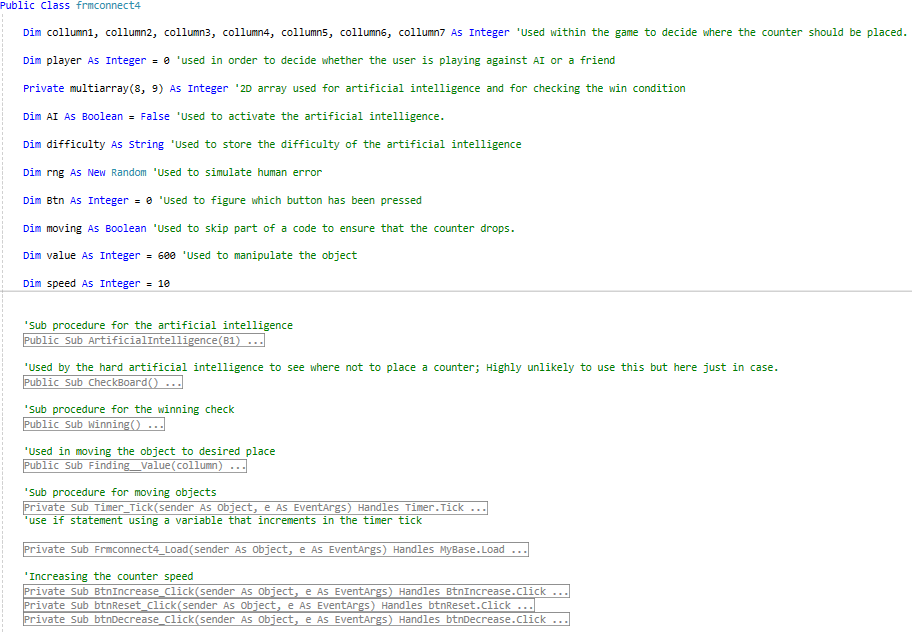
Firstly, the items within the select boxes are represented so that they can be used in the next form. The next button is used to transfer the data selected within the form to the connect 4 board using the labels to do this with. The if statement ensures that the data is selected within the two select boxes accounting for whether the user has selected one player or two players.

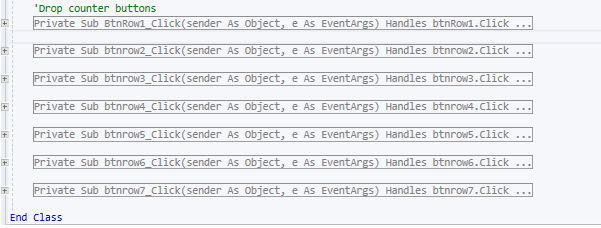
### Connect 4



Now adding some additional picture boxes to create the appearance of a traditional connect 4 board which I've found makes it easier to see what's happening for the user. Also adding some buttons to regulate the speed of the dropping counters.

#### Final code

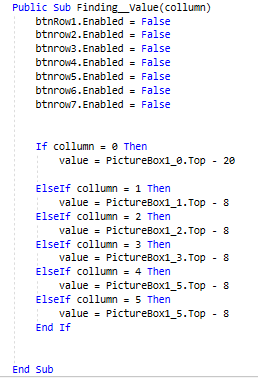




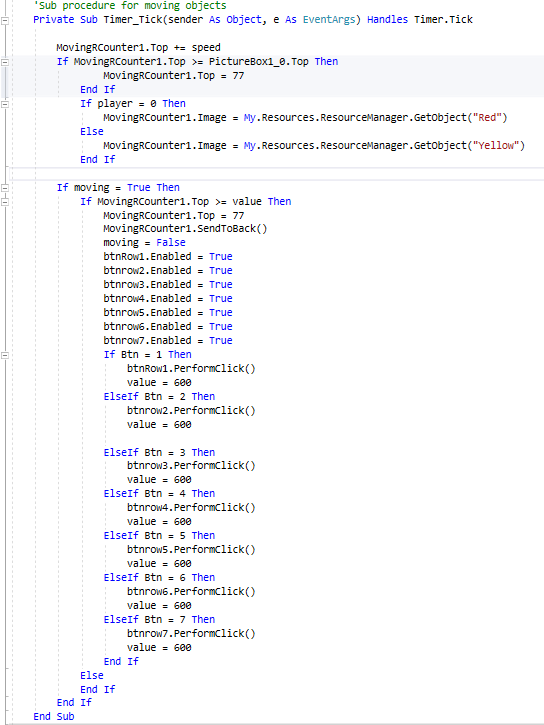
This is all the coding for the connect4 board containing the global variables at the top with the 2D array being defined there as well. All the variables have comments to distinguish the use of each one to make it easier for maintainability. After the variables the code is split into separate procedures rather than the code being copy and pasted into each button. The first algorithm is the code for the artificial intelligence containing 4 difficulties now: Easy – Artificial intelligence randomly dropping counters incoherently; Medium – The Artificial intelligence now has the ability to prevent and identify where they can win most of the time; Hard – This has more advanced coding so it can predict where the user needs to go in order to win so it will identify this and purposefully avoid that column ; Impossible – This difficulty is a copy of the Hard artificial intelligence with the only difference being human error, The impossible artificial intelligence will never make a mistake. The difference will be quite apparent. The next Procedure ***Timer\_Tick*** is used to set the location and move an object to a desired location to simulate a counter dropping onto the board. The procedure ***frmconnect4\_ Load*** is used to set the values of artificial intelligence and the number of players will be playing within the game. After this is the buttons which when pressed activate the code within it which identical to the rest using the procedures above to make it functional.

### Counter

#### Moving counter



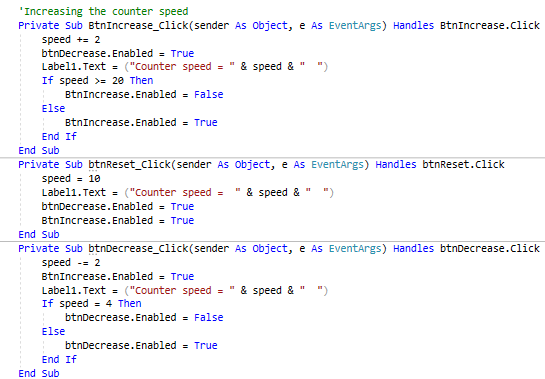
This procedure finds where the moving object should stop moving using the input given (Collumn) by the drop counter buttons. ***‘Value’*** is the variable used to store this value found being used in the next procedure to compare it to the object's height. The buttons at the start are disabled to prevent the user from pressing another button and dropping the counter elsewhere.



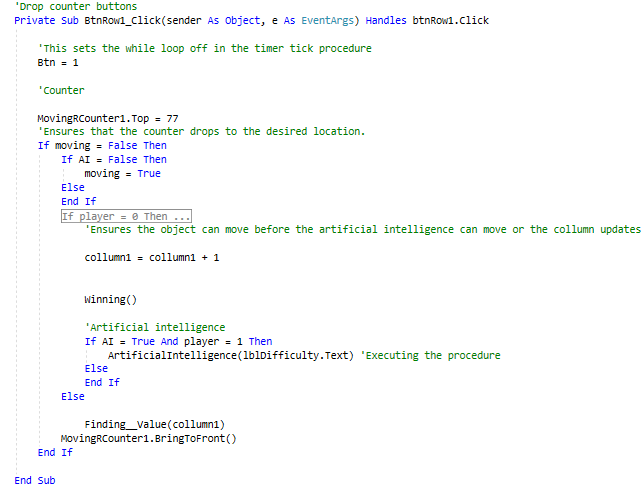
This timer is constantly going in the background, using it to move the object with time. At the top the speed at which the object moves are dependent on the ‘speed’ variable which can be altered to the users input. The first if statement is there to ensure that the ever moving object is remaining within the bounds of the connect 4 board. The second if statement is checking whether the Boolean variable ***‘Moving’*** is true executing the next if statement which is used to compare the objects height with the value declared in the procedure before this. Once true the object is sent to the back of the board so it can no longer be seen by the user and sent to the top of the board (77 value is a height on the board) The buttons that previously was disabled are reenabled so the program can do the input that the user has pressed using the ***‘Btn’*** variable which is declared within each of the drop counter buttons being declared a unique value for each one so it can be accurate. The value is then set to a large value that the object can't reach so the if statement isn’t a continuous loop.



When the connect 4 board loads this is instantly executed. The initial if statements set the labels for the artificial intelligence dependent on the difficulty set by the user on the difficulty section also it disables the timer if there is artificial intelligence due to an error I have face which I will discuss in more detail in error. If there's no artificial intelligence, then the labels will be designated to what the user inputs for the two names.

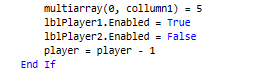


These buttons only function is to increase the speed variable which is used to increase the height of the object so this will increase the speed at which the object will move. It has caps to prevent any errors or going to slow.

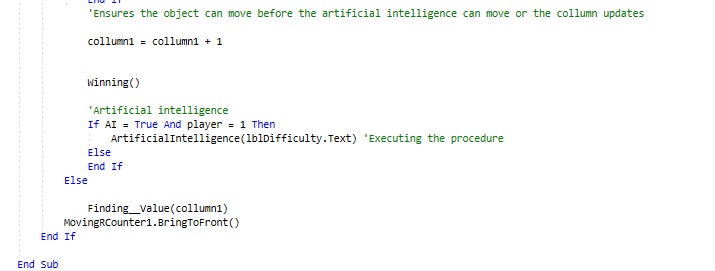


The row buttons are copies of each other only setting unique values so there is a game. Firstly the variable ***‘Btn’*** is set to a unique value to be used by the procedure above. The ‘***MovingRcounter1.top = 77’*** is used to set the object to the top of the board so it can then be dropped the value 77 being the height. The if statement after this is to counteract the errors that I faced with the artificial intelligence.





This algorithm is used to firstly get and set the image onto the picture boxes in a specific location. The player is incremented to indicate the next counter to be dropped is the opposite colour. A value is given to a location within the virtual board used entirely by the artificial intelligence so that it can manipulate the board.



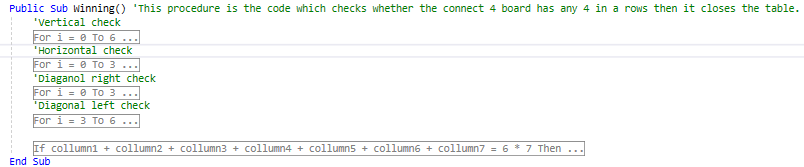
The variable collumn1 increments with each time the button is pressed so that the height can be stored, and necessary action can happen when it needs to like disable the button when the counters reach the top. ***‘The Winning()’*** is there to execute the winning procedure which goes through the entire virtual board to check every possible way you can get a 4 in a row. The next if statement is there to ensure that the artificial intelligence is activated and if it’s their turn to move. The ***‘Artificial intelligence()’*** is executing the artificial intelligence using the value set as the variable. If the ***‘moving’*** boolean value Is set to true then it will execute the ***‘Finding\_value(collumn1)’*** which will set the value to the value variable using the ***‘collumn1’*** so it knows how far to drop the counter before hiding it. The ***‘MovingRCounter1.bringToFront()’*** is used to display the counter to the user in order to show the counter dropping.

### Errors faced

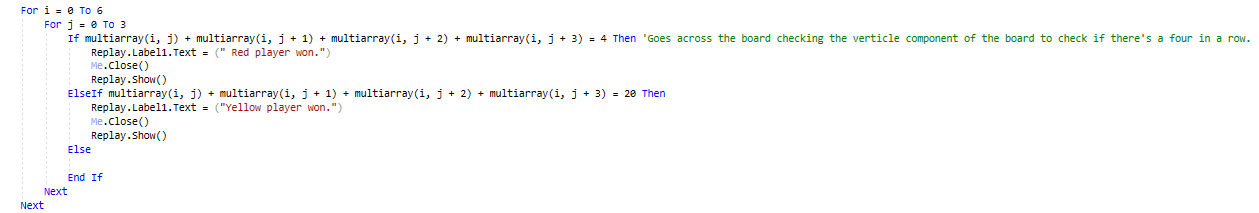
When designing the counters to be moving to give that graphical representation. I tested this function every line written using the two players mode rather than with artificial intelligence as I had an inclination that they would break down when counter moved. When in 2 player mode it worked perfectly no errors faced but once I tried to test it with some artificial intelligence the first counter would be placed then the program would drop the counter at whichever speed I chose then when it was the artificial intelligence’s turn to move it pressed a button accomplishing its function then the code would disable all the buttons as shown in the procedures above.

I found that the problem was the while loop which led t the timer not being abke to tick so to ensure that the artificial intelligence made a move if there was the rng didn’t function. Removing the while loop allowed the timer to start ticking, moving the object however there are issues which result in this like the AI sometimes just not moving and the moving object gets glitched due to the speed at which they press the button. In order to fix complete would require redoing the AI coding which I don’t have time to do so it will only be a feature in the local play mode.

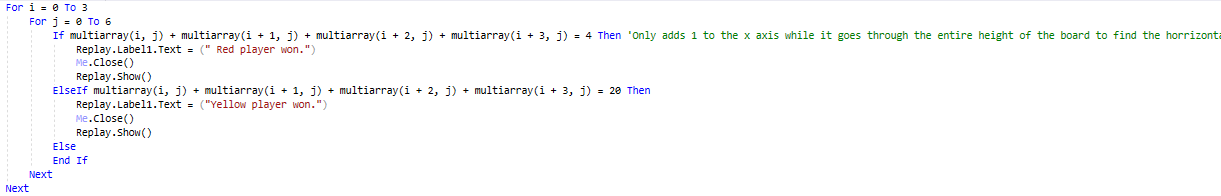
### Win check



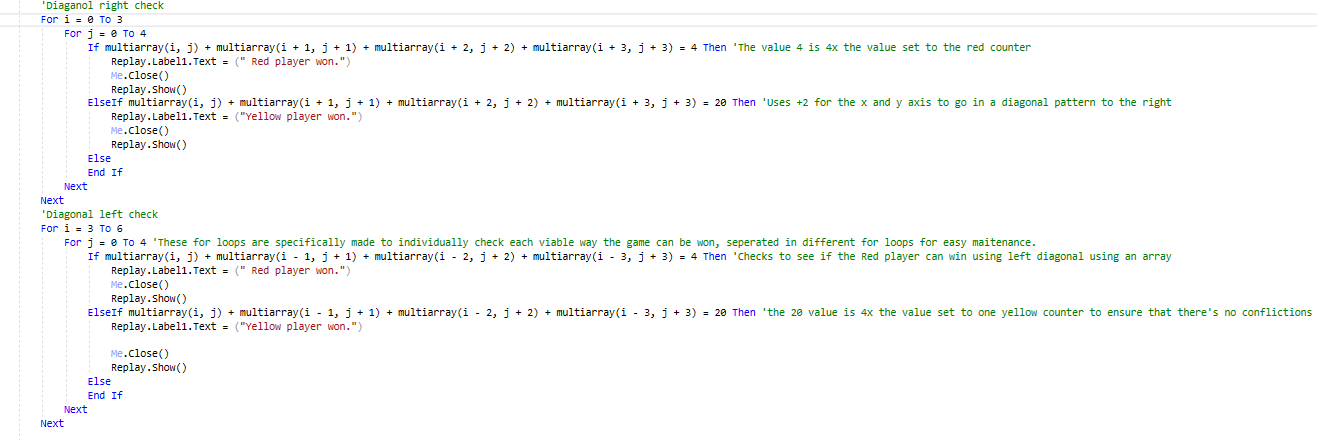
One the more important procedure as it finds whether the user has won, defeated or drawn. Each for loop is split into each function so vertical check, horizontal check and diagonal checks.



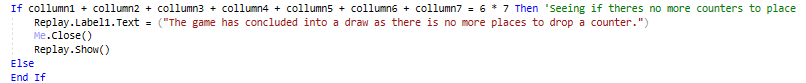
Within the vertical for loop the program goes across the connect 4 board looking at 4 cells (Slots which counters can go in) to see if the win clause is true. If so the label on the replay form will be set to whomever colour won.



The horizontal for loop is the same mathematical procedure only on the rows rather than the columns.



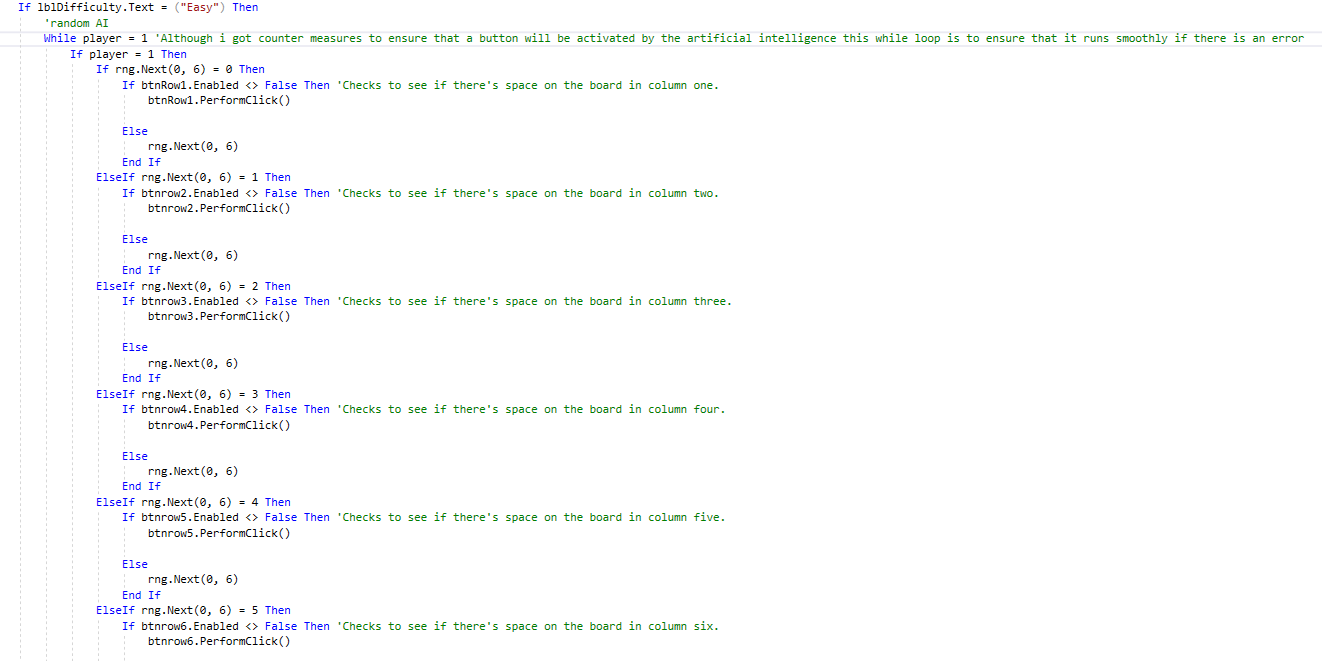
These two for loops are identical only one of them subtracting 1 from the rows to form a leftward diagonal so it can check the values set to the cells if applicable.

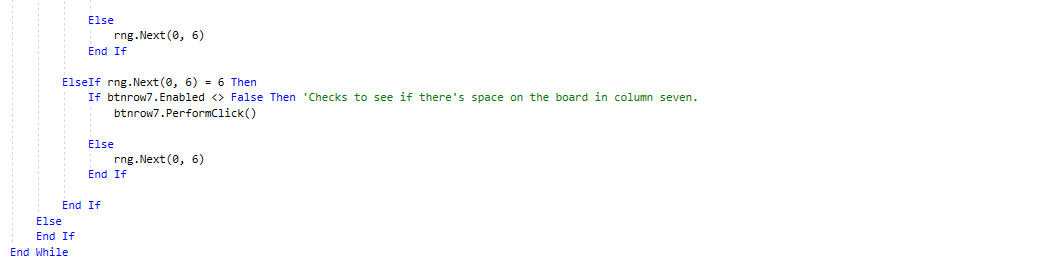


Finally the last condition which can happen in the ending of a connect 4 game is a draw which this if statement is designed to calculate this and end the game if the conditions are met.

### Artificial intelligence

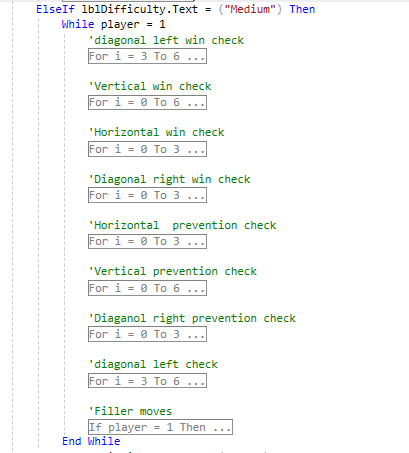
#### Easy Artificial intelligence





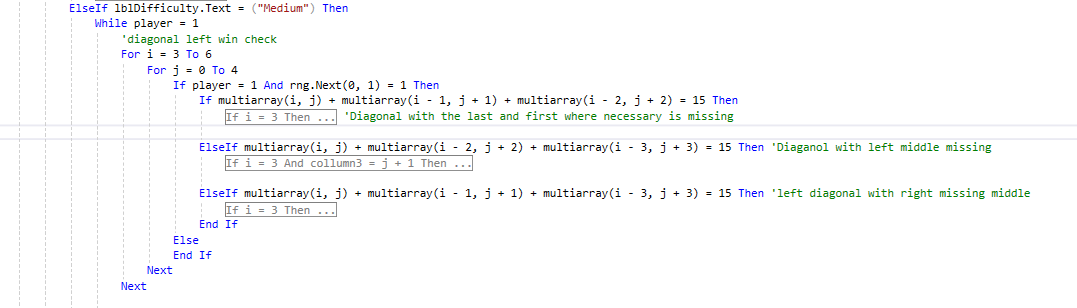
This coding for the easy artificial intelligence hasn’t been altered from the begging as I want a massive contrast from the most difficult artificial intelligence to show the user how artificial intelligence can change with more complex coding. What this code is doing: The line ***‘rng.Next(0,6)’*** is used to randomly generate an integer between 0 and 6 this has an impact in the game because the iterative statement is used to drop a counter in a column. If the number generated is 0 then it would drop the counter in the first column and so on. The comments explain the rest of the code.

#### Medium Artificial intelligence

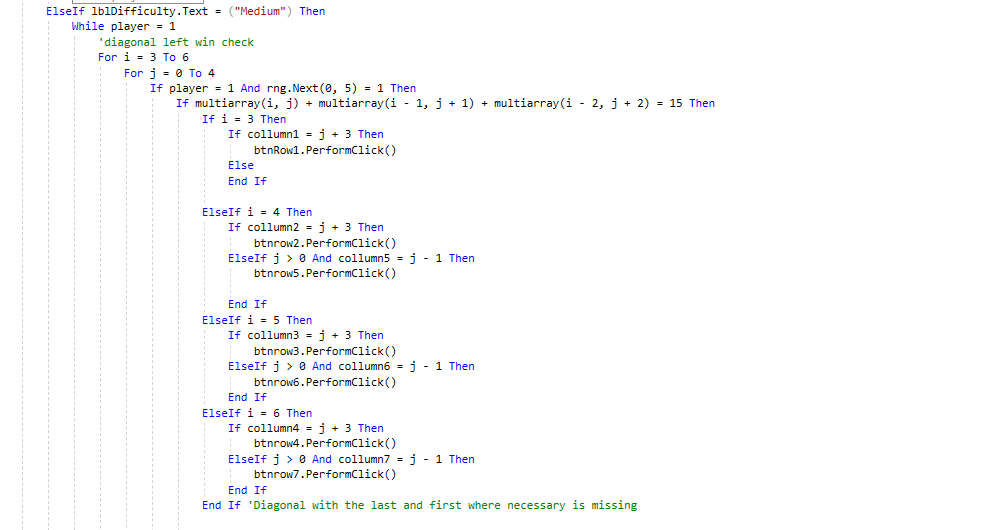


All the code for the medium difficulty are separated in individual cells labelled so it can be easily understood what each for loop will do for the artificial intelligence for someone who just looks at the code. Once more it checks for the difficulty of the artificial intelligence before executing any code to prevent unnecessary processing power being used.

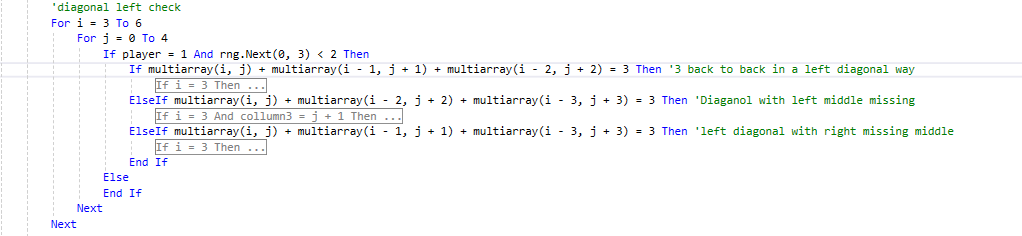
**Left diagonal**



Within the first left diagonal check where it checks whether they can get a four in a row is broken down into iterative statements labelled explaining what counter it’s looking for, making it easier for maintenance issues especially for people trying to improve the programming for the artificial intelligence. The ***‘If multiarray(i, j) + multiarray(i - 1, j + 1) + multiarray(i - 2, j + 2) = 15 Then’*** is used to cycle throughout the virtual connect 4 board which the artificial intelligence uses setting values to each of the sectors within it then using mathematical solutions to find whether it can get a 4 in a row by adding up a pattern within the array. This pattern is four in a row in a leftward diagonal with the last counter and potentially the first one missing. The ‘***rng.next’*** is used to simulate some sort of difference between hard and medium but to also represent human error as people can a do make mistakes so the artificial intelligence will also have the ability to make the same mistakes.

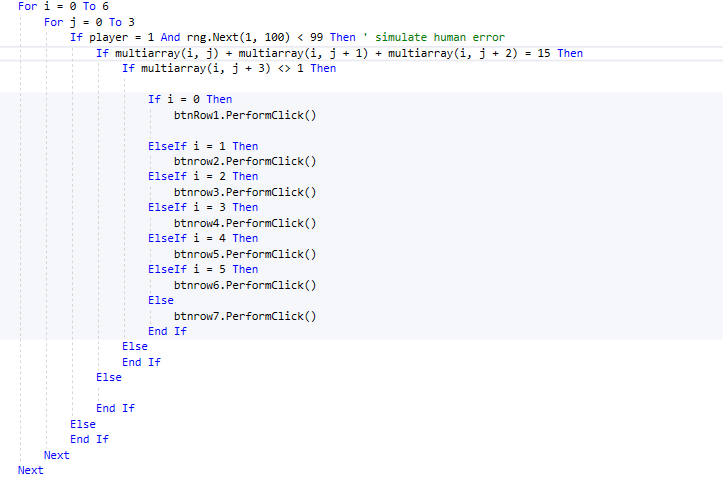


This uses an if statement to identify where the original if statement is true and acts accordingly by placing a counter down where the counters add up to 20 (Value which indicates a four in a row for the yellow counter). The if statements are split with an ‘***elseif***’ instead of having its own if statement because if one of the statements is true then it doesn’t matter if multiple is true as it only requires to win once using a four in a row so it's all together. The algorithm also checks to see if the ***‘collumn(x)***’ (a variable which keeps track on the height of the column with respect to the counters) for the desired column is available to get a 4 in a row ,using mathematics in order to find this. The rest of the code is similar maths but with different identifyers e.g(***‘If multiarray(i, j) + multiarray(i - 1, j + 1) + multiarray(i - 2, j + 2) = 15 Then’***) to manipulate the table to find the desired pattern.



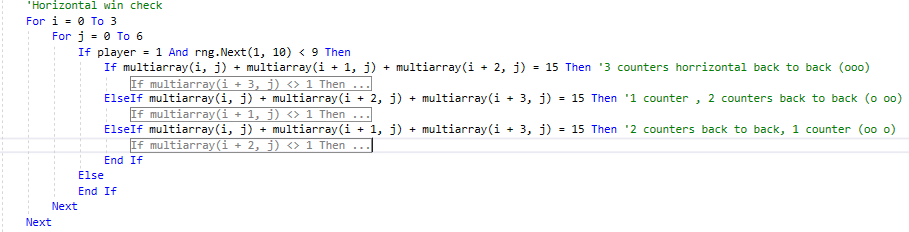
The coding is very similar to the left diagonal win with the only difference is substituting a 3 in where 15 should be in (***‘If multiarray(i, j) + multiarray(i - 1, j + 1) + multiarray(i - 2, j + 2) = 15 Then’***) As red counters are set as 1 in the virtual board.

**Vertical win**

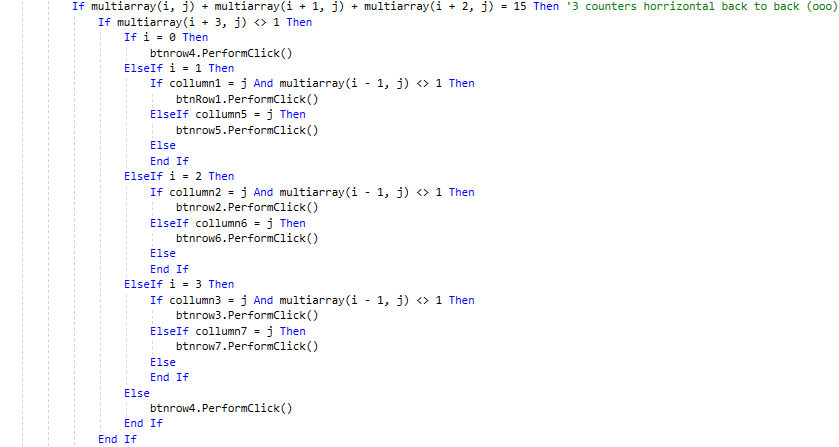


Since this is the easiest way to win in connect 4 it’s no surprise that the maths behind this one is rather simple. The for loop goes across the board 4 times in order to find every vertical option. The first if statement just manipulates the Y axis adding 1 twice to locate where there’s 3 counters vertically so it can place a counter to get a 4 in a row using ***‘If multiarray(i, j) + multiarray(i , j + 1) + multiarray(i , j + 2) = 15 Then’*** to do this with. Firstly, it checks to see if there's a red counter in the location where it can get a 4 in a row (Not needing to check for a yellow counter as the game would’ve ended if it's there). The if statement within it is used to find which row where there's 3 counters upon each other, then drops a counter on top to win.

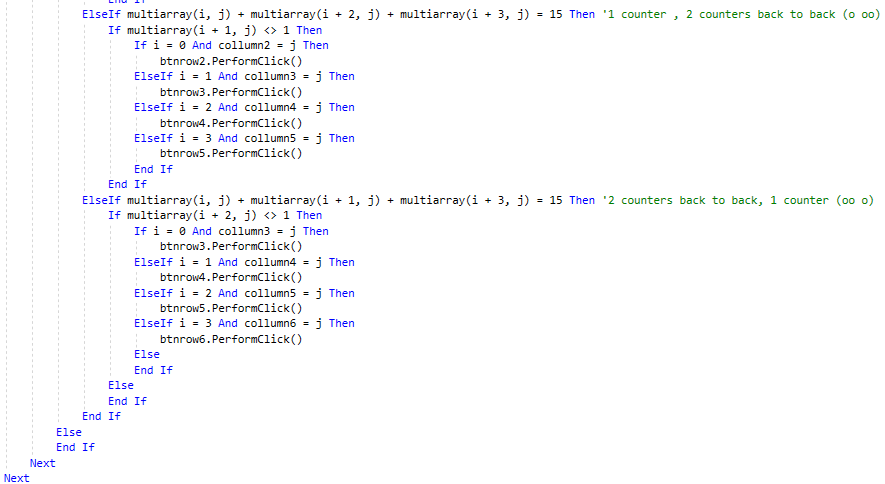
**Horizontal**



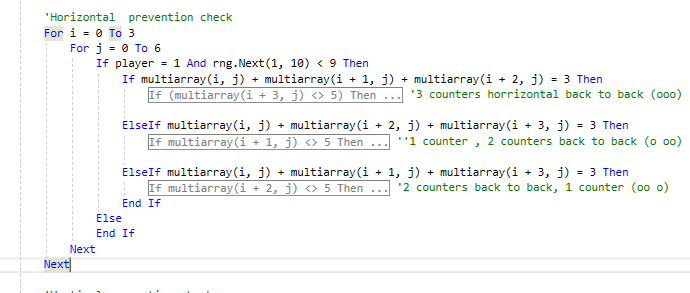
Once more the if statements are labelled with what they each identify and what they look for in order to win firstly making sure that the AI is supposed to place a counter by checking the (player) integer then seeing if the randomly generated value is suitable to look at the code at the current point since it’s within the for loop it will keep generating numbers so it could miss a line for one cycle. The maths behind this is also relatively easy its only adding values onto the x axis to find if there's any 3 counters back to back etc.



Within it, it’s the same process of checking whether the location viable to get a horizontal 4 in a row is available for the artificial intelligence to drop the counter there to win. The code ensures once  
(***‘i > 0’***) the code then has to check the two possible solutions when it’s 3 back to back. (xooox) x indicating where the counters can be placed and o being the yellow counters.

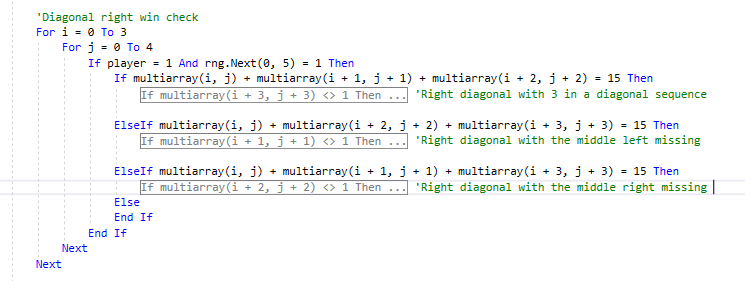


These use the same method to find if a counter is free so the artificial intelligence can win after identifying that there are a combination of these counters within the board.

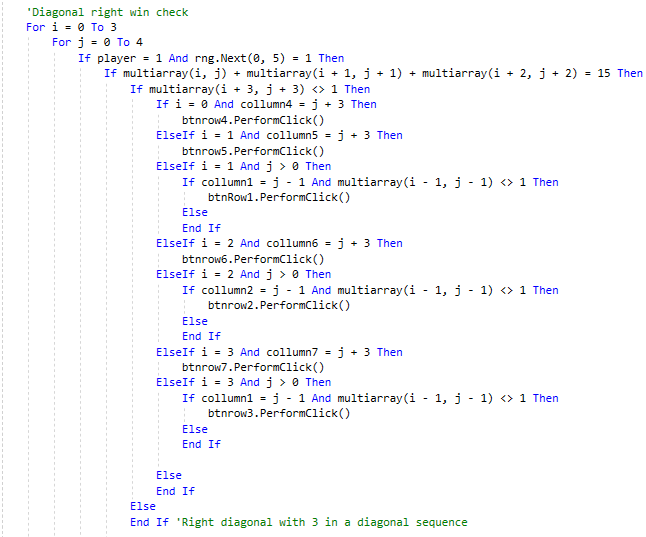


This algorithm is for the use of the artificial intelligence to find if the player is about to win horizontally. The only difference would be what the values of the cells add up to, being 3 instead of 15 as red counters are set to 1 in the array and yellow set to 5 to avoid any conflictions. The other difference would be instead of checking for a red counter in the slots which could result in a 4 in a row, it checks for a yellow counter instead as there wouldn’t be any point checking for a red because the game would’ve concluded.

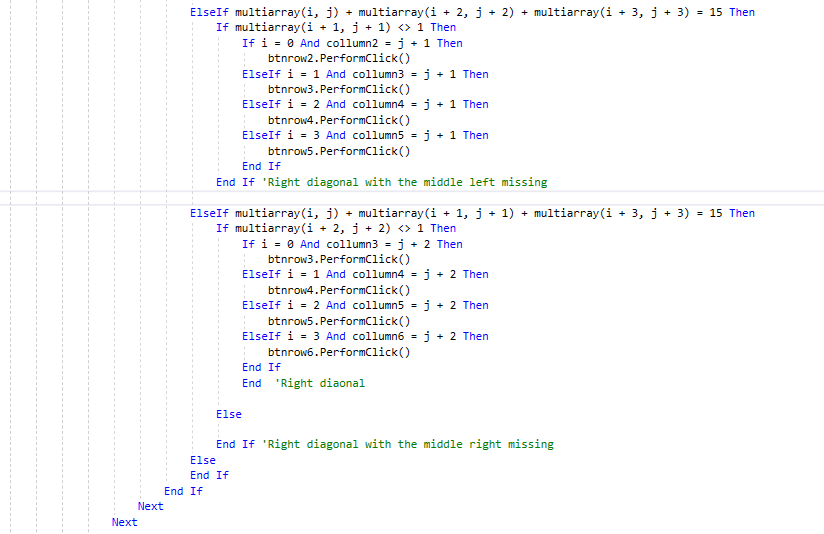
**Diagonal right**



Within the labelled win check is further labels indicating which iteration does what. With similar maths used in the diagonal left but instead of subtracting the x value to get a left diagonal we just use addition instead.

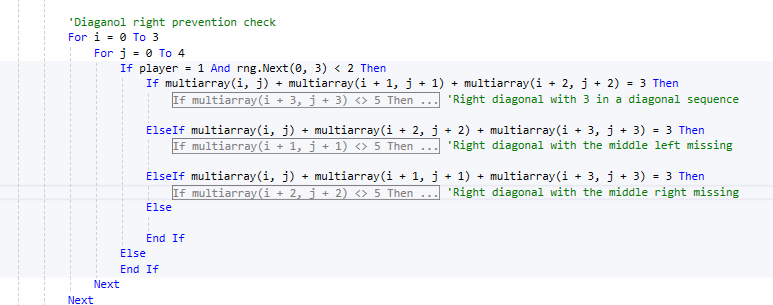


Simulating human error once more in this algorithm doing similar code, I've discussed above using other mathematical solutions such as ***‘Multiarray(i - 1, j - 1) <> 1’*** This code is looking at the location to the bottom of the diagonal when it’s above the first row reasuring that there will be a level of accuracy.



The rest of it is using different mathematical solution to get the desired pattern in the right diagonal which is explained in the code with what each if statement does.

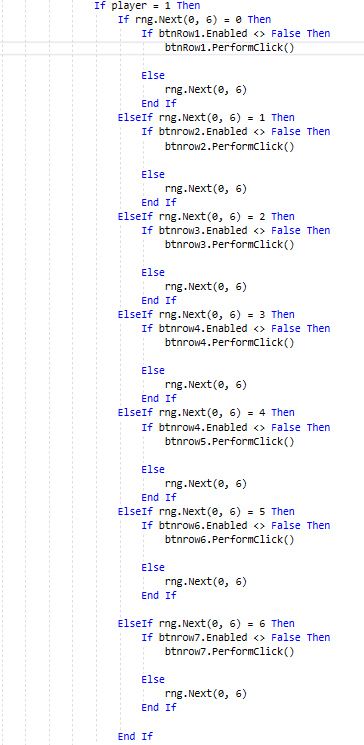
prevention check



This code is identical to the win check with certain values altered to replace the yellow counters in the pattern with red counters. Using the same method to find this and ensure that instead of getting a four in a row this attempts to block one.

**Filler moves**

This Algorithm is used when none of the above can be used so in order to play the game the program randomly drops counters until the algorithms above can be executed.



This algorithm is pinched from the easy artificial intelligence as this is to do the basic function.

#### Hard Artificial intelligence

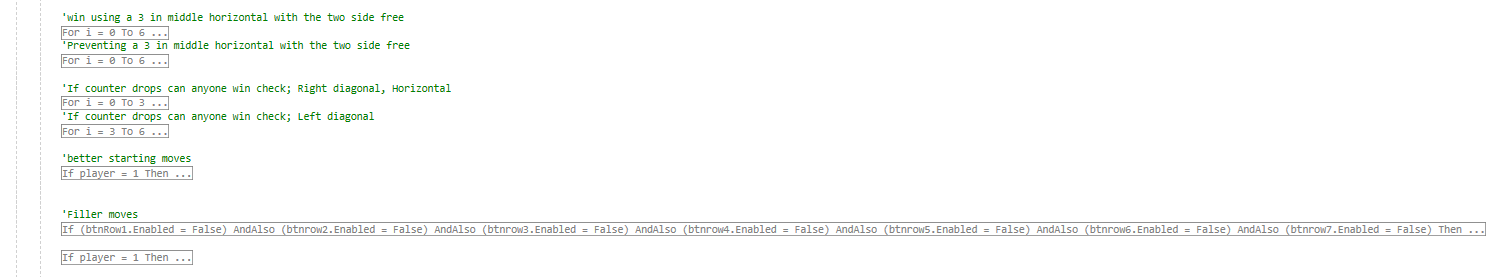
The hard-artificial intelligence has the same functions of the medium artificial intelligence with some alterations for the code so the artificial intelligence is noticeably more difficult than its predecessor. I will only point out the alterations made between the two.



Similar layout to the previous code being labelled for what each for loop does so it can be easily altered and easy to isolate the issues rather than searching through one massive for loop.



Firstly, the difference I have made is the RNG range so that there’s a slimmer chance that the artificial intelligence will miss a sequence of counters that can achieve a 4 in a row.

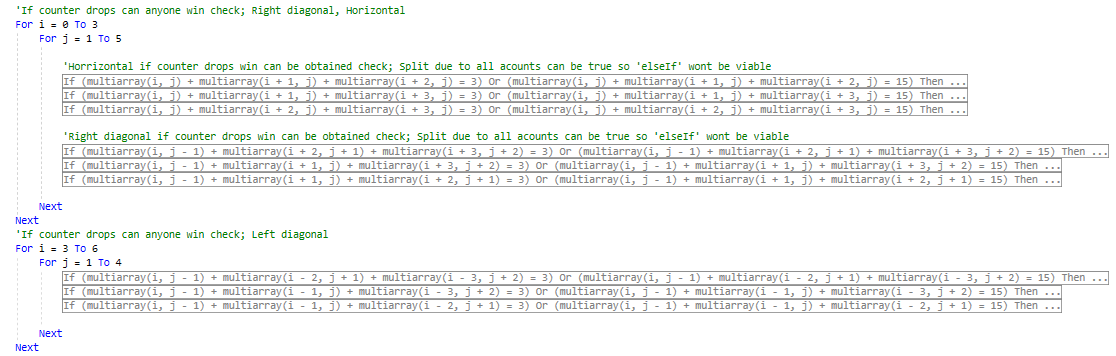


These are the new algorithms for the hard-artificial intelligence. Within the game of connect 4 there is a way where the user can have a certain win if the opposition doesn’t predict it, Im talking about having a 3 back to back horizontally where the two sides on either side are free for the counters to be dropped, meaning only one of the two spaces where you can get a 4 in a row can be blocked resulting in a certain win. That’s what the first two for statements aim to accomplish to prevent it and attempt to achieve one. Positioning is key in these for statements as you can see it’s near the bottom of the hard-artificial intelligence as you don’t want this to happen rather than the AI stopping a four in a row, so positioning is used to prioritise what the artificial intelligence should look at before they randomly place a counter.

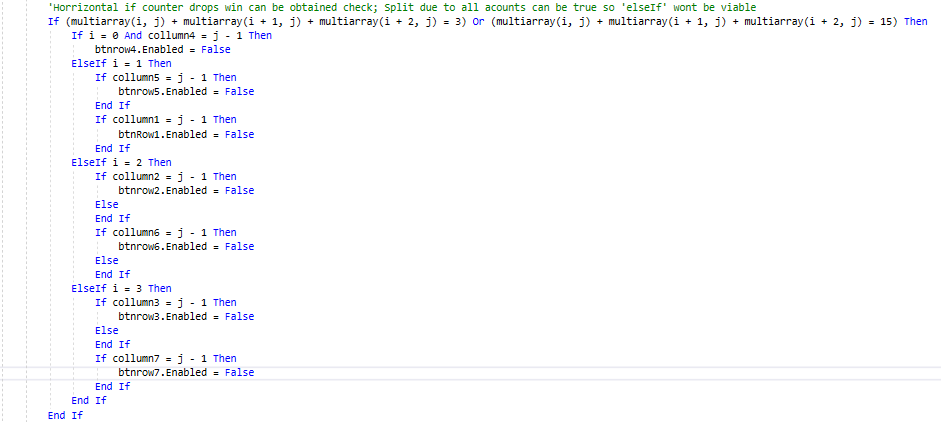


The RNG still applies for this as it’s a guarantee win if the AI can place a counter. This code checks to see if there is 2 counters back to back or 2 counters within 3 slots on the same row and check the necessary slots to check if there's a counter there to block the attempt of a 3 in a row.

The prevention just switches the 10’s and 15’s to 2’s and 3’s.

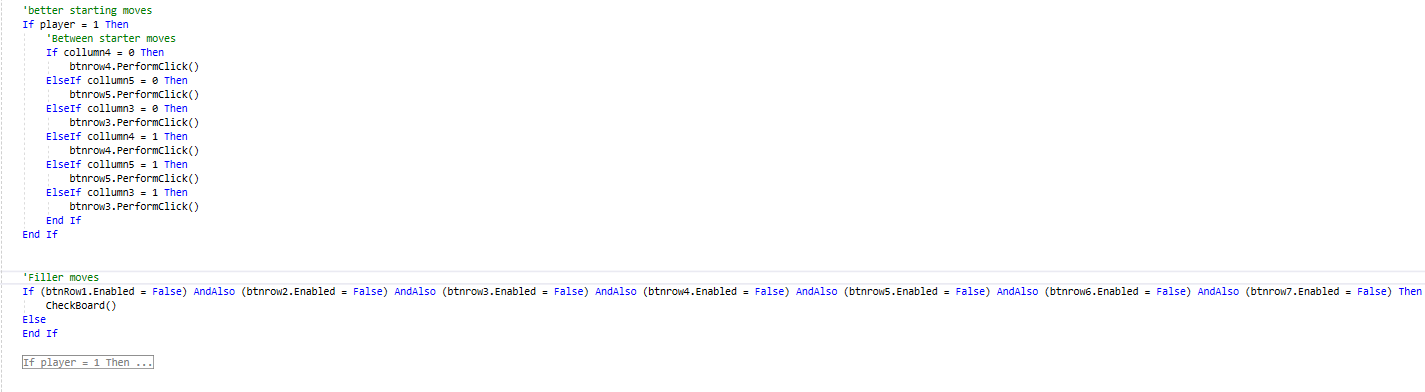


This algorithm is the main body of the hard-artificial intelligence which differentiates itself from the medium difficulty. The purpose of this code is to predict where the user might place their next counter. The Y axis is focused on in this algorithm seeing when a counter is placed can the opponent get a 4 in a row if so then the artificial intelligence makes note of it and blocks that column from being used on it’s turn. The program does horizontal and diagonal leaving out vertical due to it not being relevant. Each of them have a separate IF statement due to the AI just identifying which column would result in a loss and multiple of these statements could be true so in order for the Artificial intelligence to identify all the columns the if statements have to be separate.

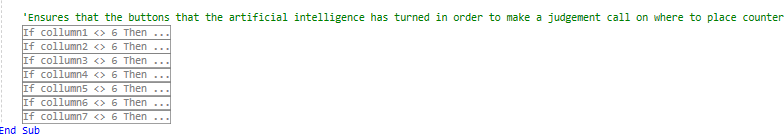


Instead of pressing a button the artificial intelligence sets the button as disabled so it can't be pressed to avoid from making that mistake as we humans self-consciously identify this and block that column. Other instances of this to find the different ways of getting a horizontal 4 in a row are in the other if statements.

The right and left diagonal checkers have the same function of identifying when the player is viable to get a four in a row if the AI places the counter in a place but this time in diagonal trajectory.



The first bit is hardcoded moves the artificial intelligence should take if they want to stand a decent chance at winning against a player. The procedure check board is described in the intro to the connect 4 program. The filler moves if statement is there as sort of a checker just in case somehow every single button is disabled for the artificial intelligence so it will then check the board to see where it can place a counter in order to not lose. If that cannot be accomplished and the AI has been tricked, then the AI will enable all buttons it’s disabled due to the program above and drop a counter.



At the end it reenables the buttons that the artificial intelligence has disabled.

#### Impossible Difficulty

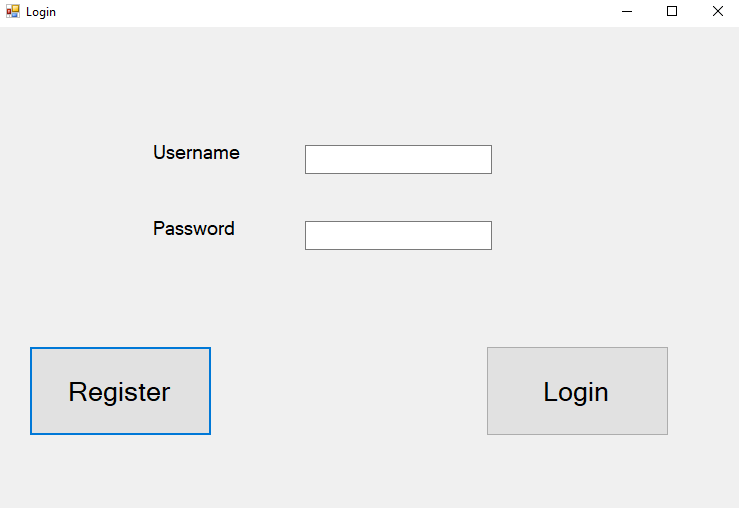
The impossible difficulty removes any RNG aspects added into the hard artificial intelligence so it will also go through all of the code no matter what. This was created firstly to make a real challenge for people, secondly to have a major contast between easy artificial intelligence and Impossible difficulty and finally for testing purposes to ensure that the functions of the artificial intelligence would actually function which was difficult to determine as the RNG sometimes played a factor in testing.

### Replay

# Functional program

The following forms will be checked against the success criteria to ensure that everything necessary is completed and is fully functioning

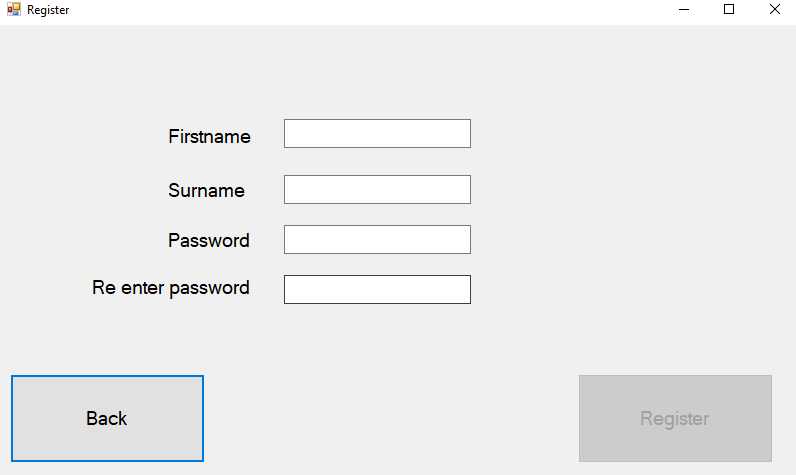
## Login form



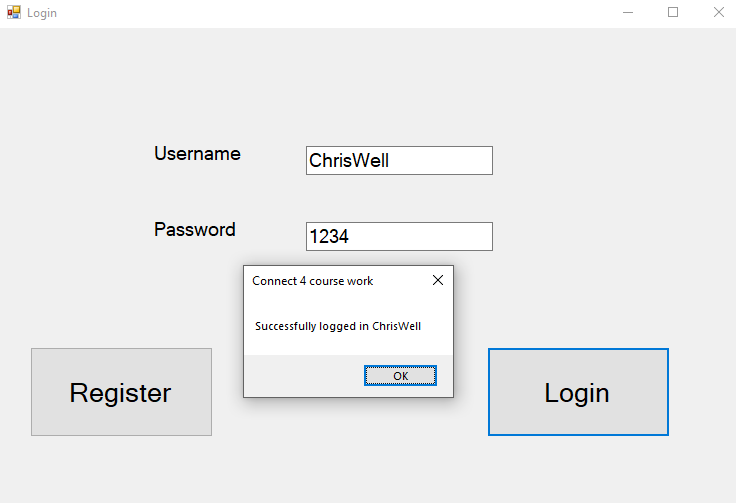
|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game |  |
| Inputted data is in correct data type e.g only characters for names |  |
| Allows user to input into database |  |
| Database can retrieve relevant information and pass it to program |  |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form |  |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) |  |
| ‘Register’ button inputs data within the database and displays it |  |
| User can select Artificial intelligences difficulty |  |
| ‘Drop counter’ button allows the user to drop the counter |  |
| Winning conditions can be met and is viable |  |
| Artificial intelligence is present within the game |  |
| Artificial intelligence with simple coding works |  |
| Medium difficult artificial intelligence is functioning |  |
| Hard difficulty artificial intelligence is functioning |  |
| The game terminates after the user is finished with program |  |

Testing the Register Button.

Once pressed this form pops up you can now safely say that the register button functions



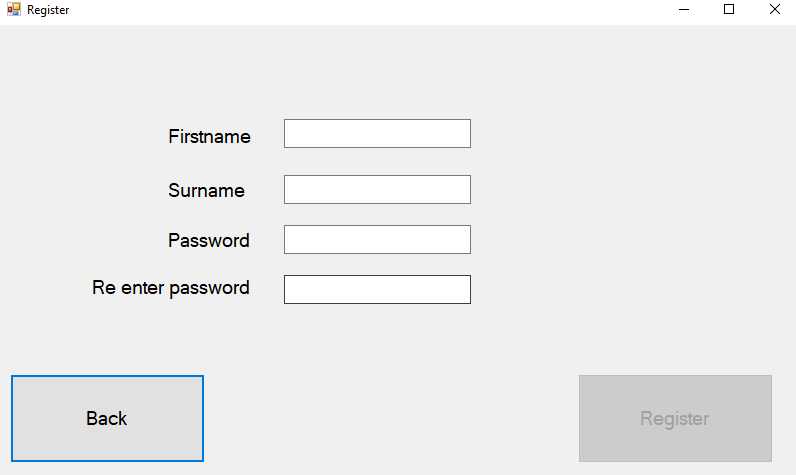
Once login is pressed however with the valid information



Once login is pressed you are welcomed by the program identifying that you entered the correct details.It then opens the difficulty form.Since it retrieves data from the database it safely functions.

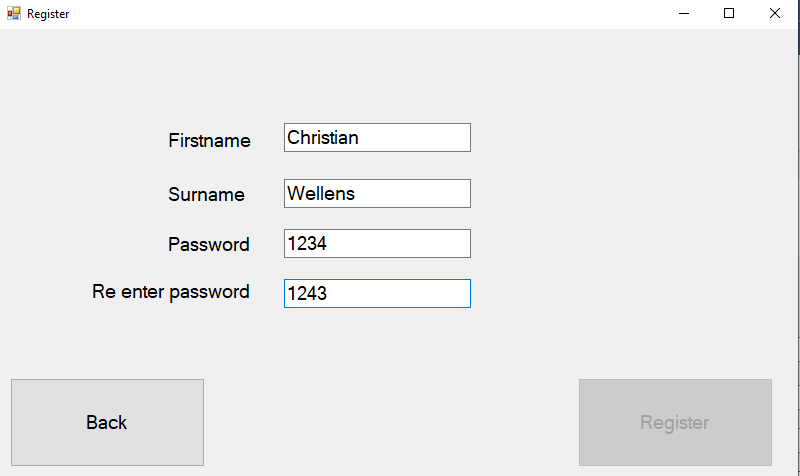
|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game | **YES** |
| Inputted data is in correct data type e.g only characters for names | **YES** |
| Allows user to input into database | **YES** |
| Database can retrieve relevant information and pass it to program | **YES** |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form | **YES** |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) |  |
| ‘Register’ button inputs data within the database and displays it |  |
| User can select Artificial intelligences difficulty |  |
| ‘Drop counter’ button allows the user to drop the counter |  |
| Winning conditions can be met and is viable |  |
| Artificial intelligence is present within the game |  |
| Artificial intelligence with simple coding works |  |
| Medium difficult artificial intelligence is functioning |  |
| Hard difficulty artificial intelligence is functioning |  |
| The game terminates after the user is finished with program |  |

## Register form



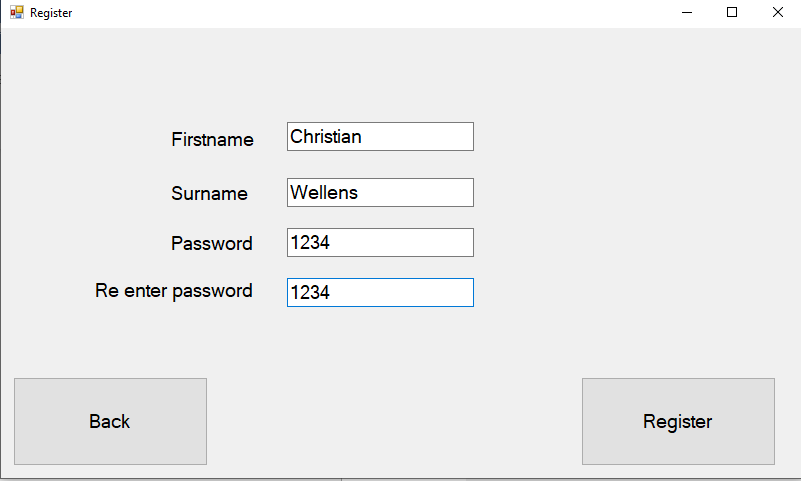
Testing Back button, Once pressed it goes back to the login form. Making the back button functional.

|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game | **YES** |
| Inputted data is in correct data type e.g only characters for names | **YES** |
| Allows user to input into database | **YES** |
| Database can retrieve relevant information and pass it to program | **YES** |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form | **YES** |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) |  |
| ‘Register’ button inputs data within the database and displays it |  |
| User can select Artificial intelligences difficulty |  |
| ‘Drop counter’ button allows the user to drop the counter |  |
| Winning conditions can be met and is viable |  |
| Artificial intelligence is present within the game |  |
| Artificial intelligence with simple coding works |  |
| Medium difficult artificial intelligence is functioning |  |
| Hard difficulty artificial intelligence is functioning |  |
| The game terminates after the user is finished with program |  |

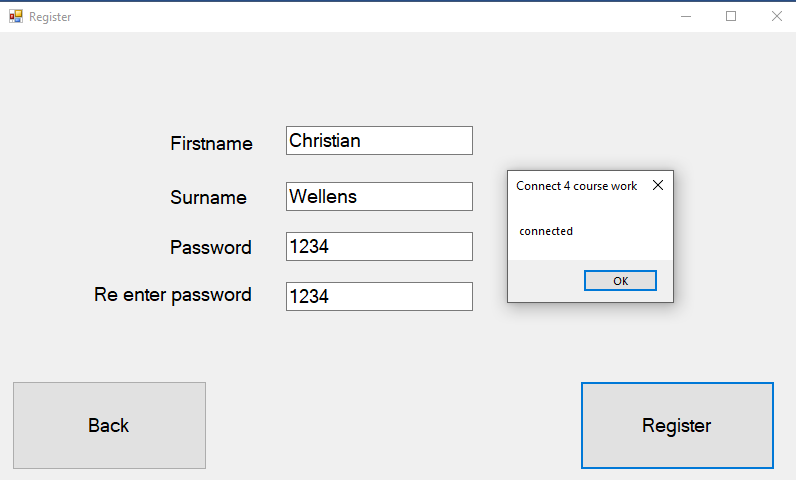


Once the details are entered the register button will be enabled, however in this case it wont as the password that they re-entered is different to the password.

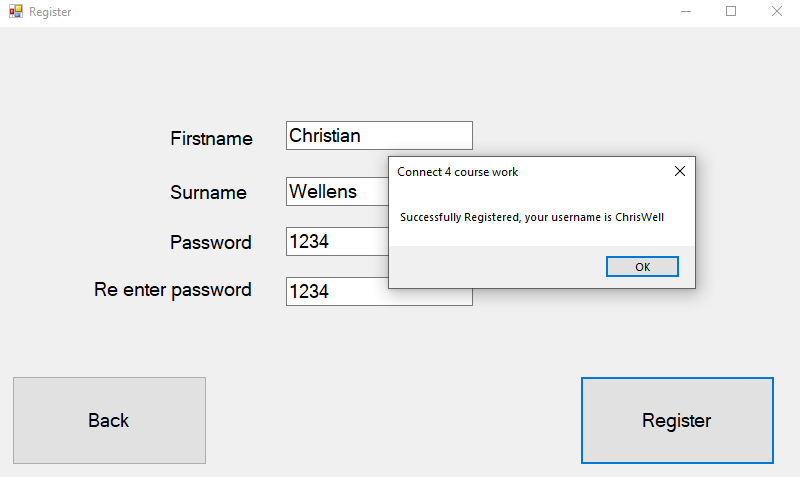
Once amended the register button is enabled as seen below



Once register is clicked the database should be amended and values should be added



Firstly, the test is commenced to check if the database is connected to the register



Secondly the program shows that you have registered and displays your generated username then the login form is shown for you to log in.

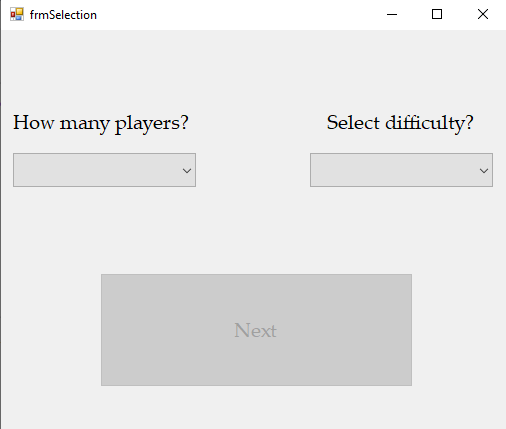
|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game | **YES** |
| Inputted data is in correct data type e.g only characters for names | **YES** |
| Allows user to input into database | **YES** |
| Database can retrieve relevant information and pass it to program | **YES** |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form | **YES** |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) | **YES** |
| ‘Register’ button inputs data within the database and displays it | **YES** |
| User can select Artificial intelligences difficulty |  |
| ‘Drop counter’ button allows the user to drop the counter |  |
| Winning conditions can be met and is viable |  |
| Artificial intelligence is present within the game |  |
| Artificial intelligence with simple coding works |  |
| Medium difficult artificial intelligence is functioning |  |
| Hard difficulty artificial intelligence is functioning |  |
| The game terminates after the user is finished with program |  |



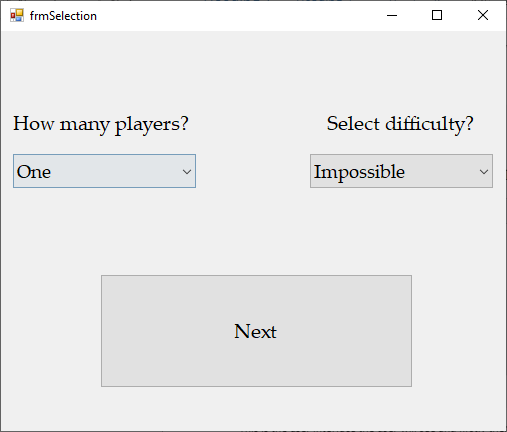
This is within the database as you can see the data has been submitted

## Selection form

Within this form you should be able to choose how many player are in the game and what difficulty the artificial intelligence should be.



This Is the user interface the user will see and firstly there are selection boxes that cant be typed into in order to enter details. In order to be able to press next the user must have the boxes filled before they can press the next button.



The boxes are now filled out and the next button is now available for the user to press to go into the connect four game. Selecting one player and impossible difficulty.



If there are 2 players within the connect 4 game, then there will be no Artificial intelligence present.

The user from this form can select how many players are playing and what difficulty if applicable the artificial intelligence should be set as. Although the difficulty select box remains with the last chosen item it ensures the variable set to it is null when there are multiple player.

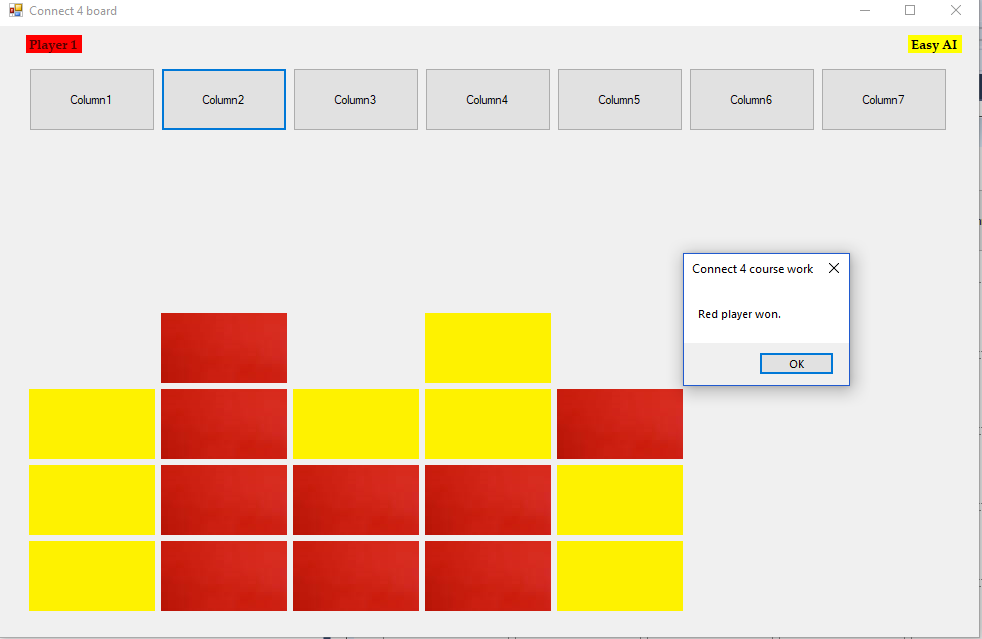
|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game | **YES** |
| Inputted data is in correct data type e.g only characters for names | **YES** |
| Allows user to input into database | **YES** |
| Database can retrieve relevant information and pass it to program | **YES** |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form | **YES** |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) | **YES** |
| ‘Register’ button inputs data within the database and displays it | **YES** |
| User can select Artificial intelligences difficulty | **YES** |
| ‘Drop counter’ button allows the user to drop the counter |  |
| Winning conditions can be met and is viable |  |
| Artificial intelligence is present within the game |  |
| Artificial intelligence with simple coding works |  |
| Medium difficult artificial intelligence is functioning |  |
| Hard difficulty artificial intelligence is functioning |  |
| The game terminates after the user is finished with program |  |

## Connect 4 game (Testing done before graphics added)

Within the connect 4 game most of the success criteria are based in this part as its one of the main and crucial parts of the code

Firstly, we must select how many players and the difficulty of the artificial intelligence as the connect 4 game would just function as if there were only 2 players. I will represent how the game functions with the winning condition representing proof that the winning condition works.

### Vertical win

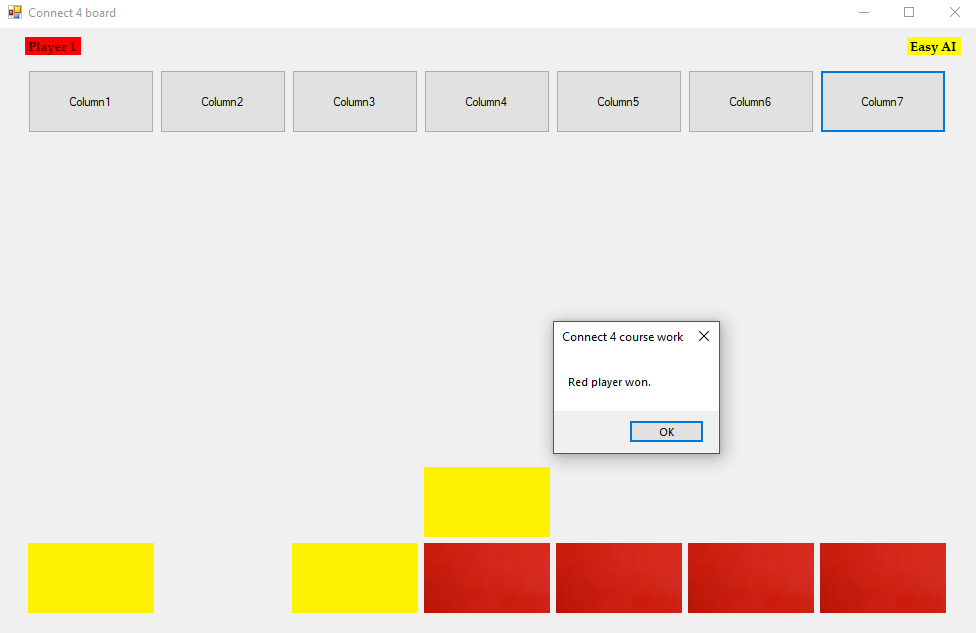


Once you have gotten 4 in a row it displays which player has won using the win checker. Currently this shows it working with vertical 4 in a row. Being displayed against the easy artificial intelligence as it won't counter any of it except maybe by chance.

With accordance to the screenshots above, this checks if the counters on the board whichever colour it may be if there’s 4 of them in a row, checking for a winning condition. So when comparing it to the success criteria you can see that it meets ‘winning conditions can be met’ and we can also cross off ‘Game terminates’ as you can see that the game closes when someone wins.

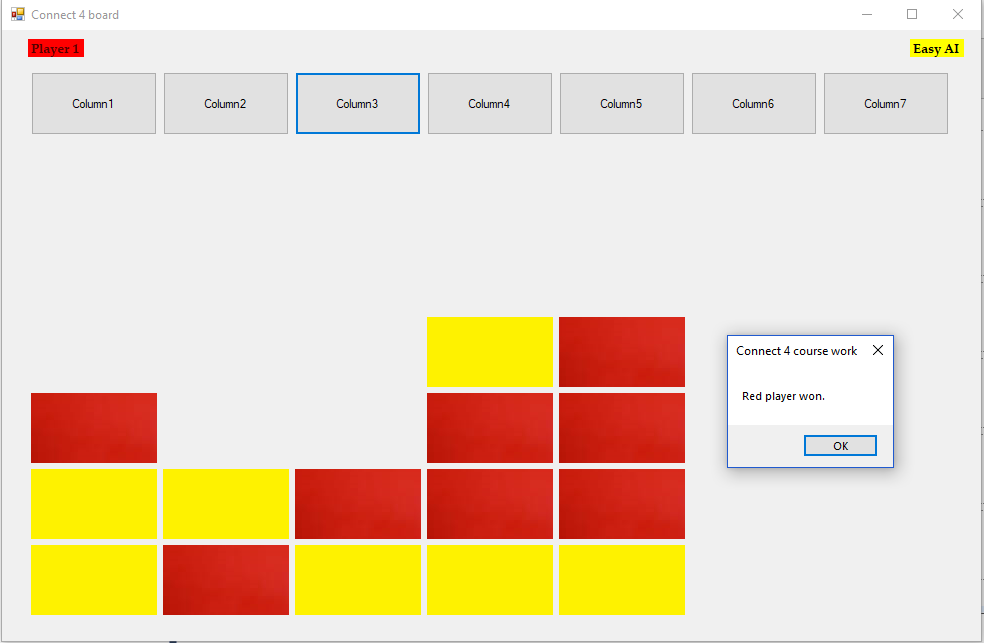
|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game | **YES** |
| Inputted data is in correct data type e.g only characters for names | **YES** |
| Allows user to input into database | **YES** |
| Database can retrieve relevant information and pass it to program | **YES** |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form | **YES** |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) | **YES** |
| ‘Register’ button inputs data within the database and displays it | **YES** |
| User can select Artificial intelligences difficulty | **YES** |
| ‘Drop counter’ button allows the user to drop the counter | **YES** |
| Winning conditions can be met and is viable | **YES** |
| Artificial intelligence is present within the game |  |
| Artificial intelligence with simple coding works |  |
| Medium difficult artificial intelligence is functioning |  |
| Hard difficulty artificial intelligence is functioning |  |
| The game terminates after the user is finished with program | **YES** |

### Horizontal win



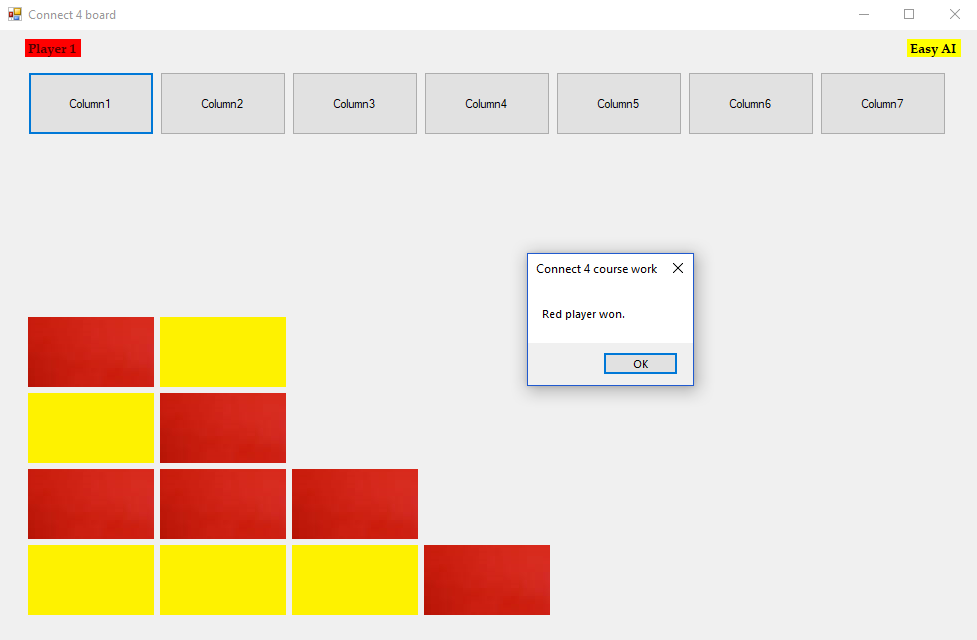
This shows that the horizontal win checker is functioning as planned while displaying how the easy artificial intelligence is randomly placing counters making it the easiest artificial intelligence to display these wins.

### Diagonal wins



As shown above playing against the easy artificial intelligence the red player has placed a 4 in a row against the yellow player which the program has identified and has labelled red player as the victor

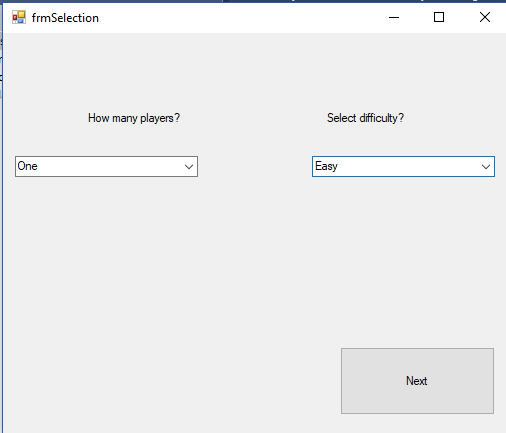
Left diagonal win



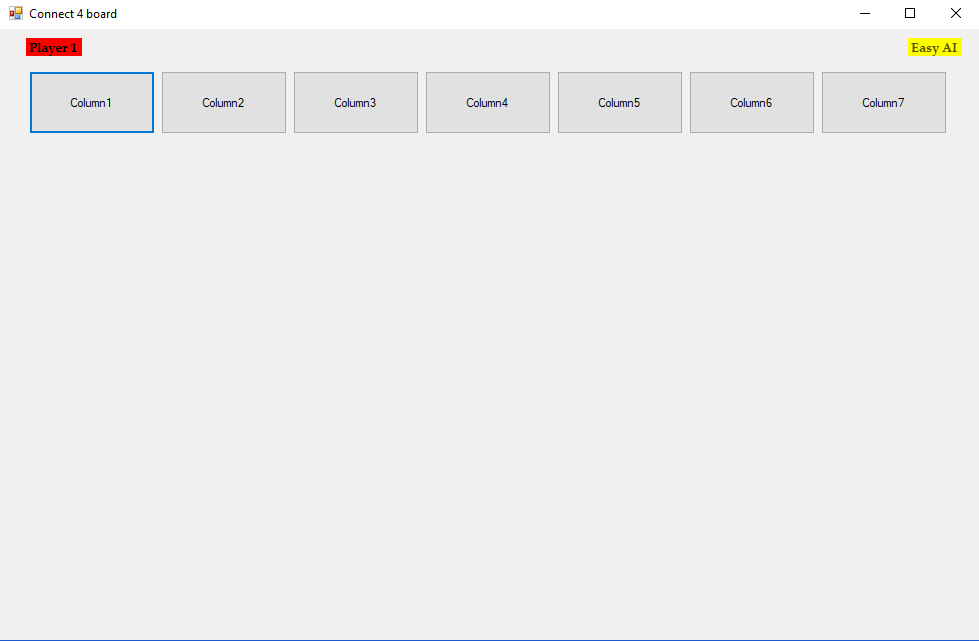
This shows that the program identifies when there is 4 in a row going the diagonal left way claiming the red player as the victor.

### Easy artificial intelligence

This artificial intelligence relies on a random integer being selected between 1 and 7 representing the columns in which can drop a counter to have a very simplistic challenge to the player to be easily distinguished between it and its other difficulties



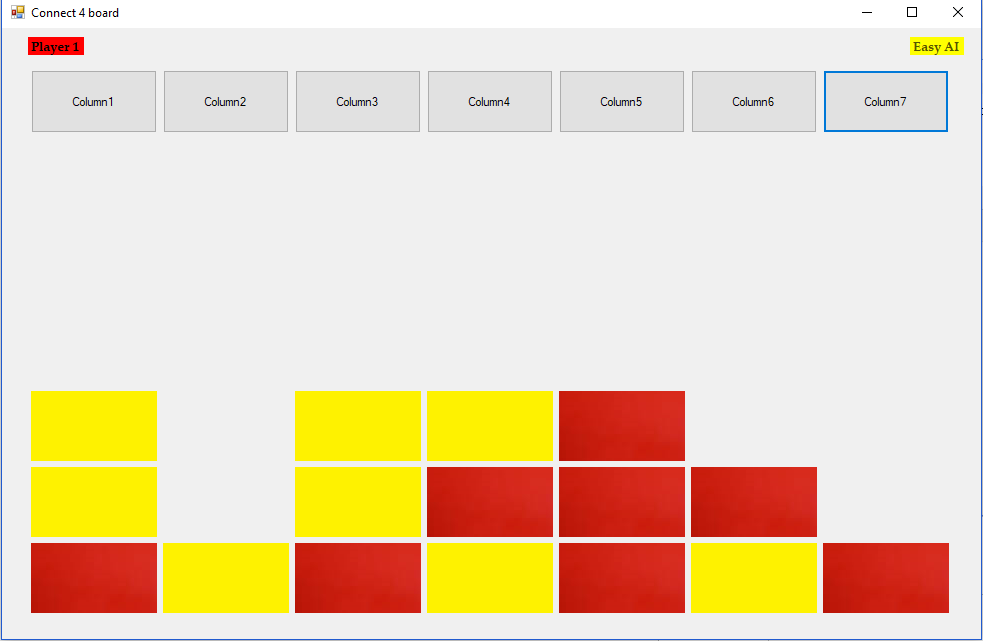
Once next is pressed it will continue the program to the actual game where it will present the difficulty of the artificial intelligence enabling that difficulty to a set of programming.



It will represent the artificial intelligences difficulty on the top right, waiting for player1 to make the initial move to randomly place a counter in one of the columns



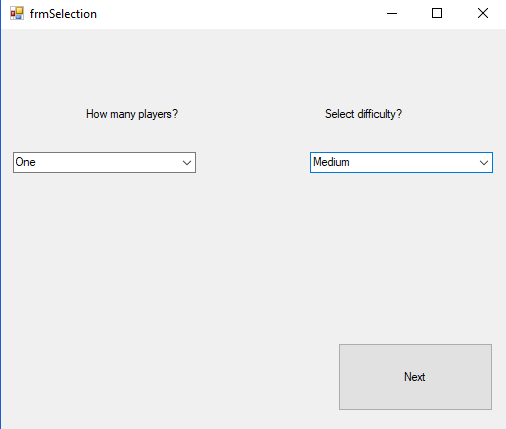
The button column1 was pressed and the artificial intelligence randomly selected column4 to drop their counter



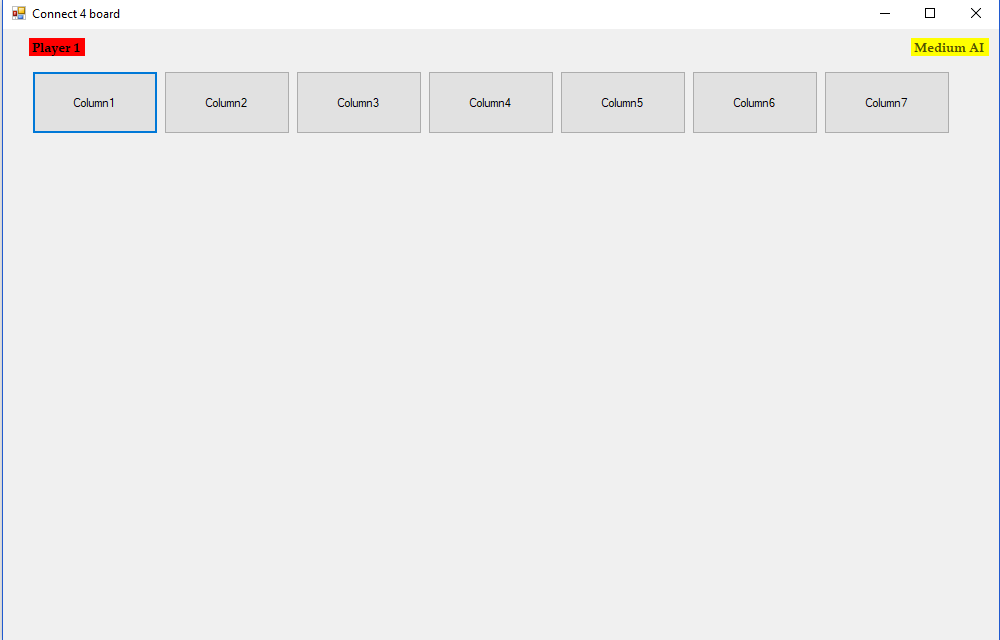
After a few counters drops it shows that the artificial intelligence doesn’t react when the user dropping red counters is going to win and similarly when the artificial intelligence is going to win as they haven't dropped one in column5 or in column 7 to prevent red from winning.

### Medium artificial intelligence

This artificial intelligence will be more of a challenge to the player as it will prevent wins and search for a 4 in a row for itself dropping random counters until this is true



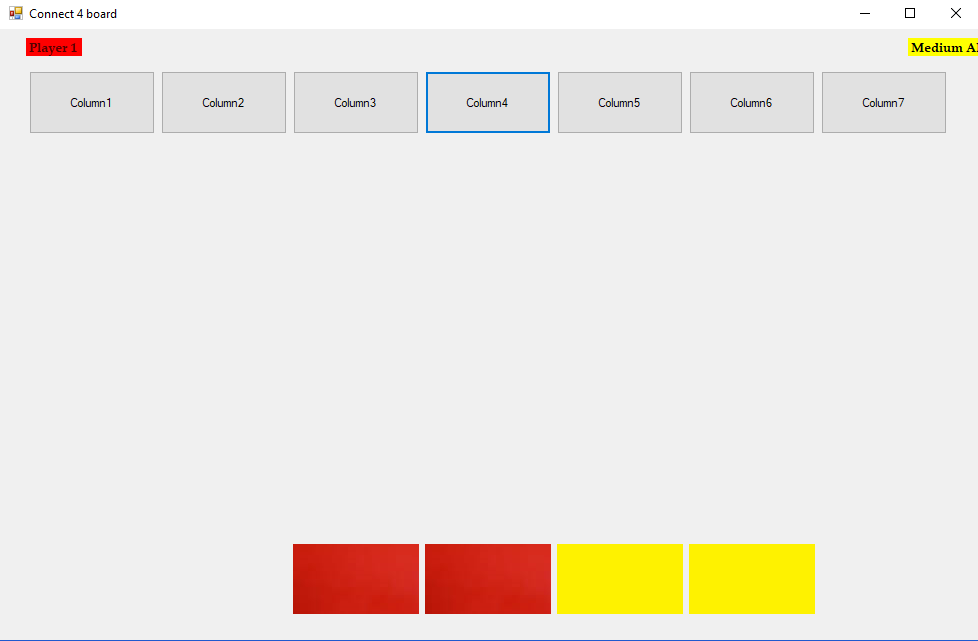
This just sets the difficulty to be used in the connect 4 game without this form it will act just as a game between 2 people.



Once next has been pressed the board game will be presented with the medium artificial intelligence shown in the top right highlighted in yellow to represent to the user that they’re dropping the yellow counters.



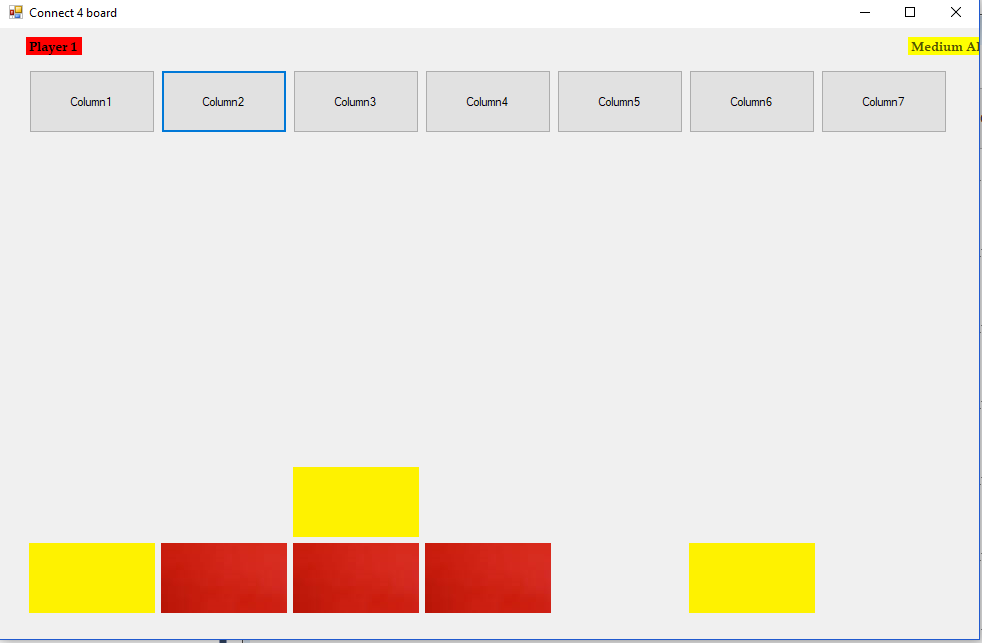
Column3 was pressed dropping a counter in that section now incrementing that column to present that the counter will now land on top of it in the next slot 1 above.



Column4 is pressed to drop counter in 4th column in which the artificial intelligence has responded in putting a counter in column 5.

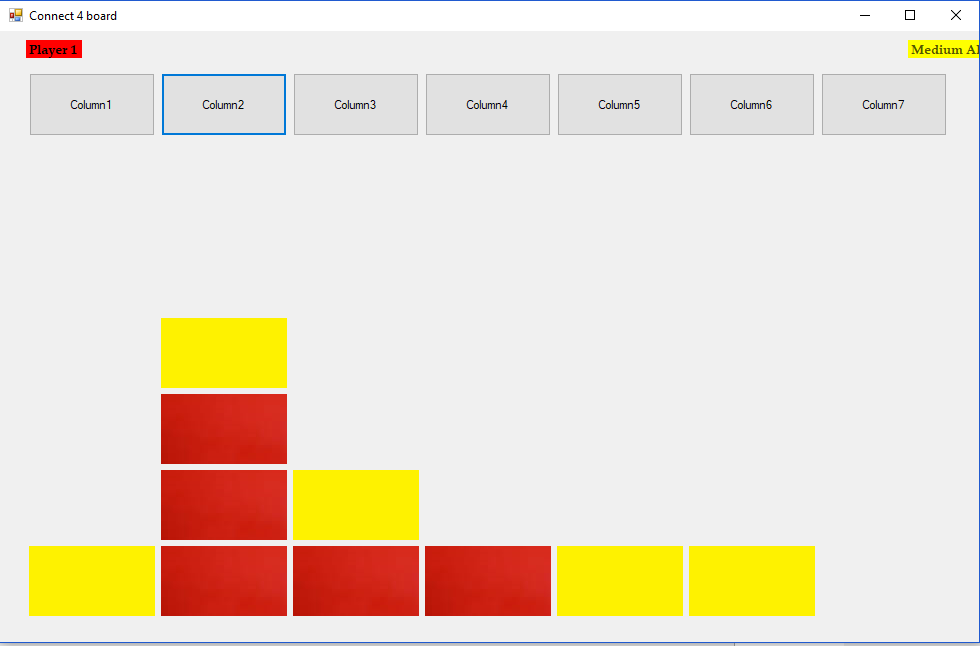
Preventing the player from winning

Preventing horizontal win



Once column 2 was pressed the artificial intelligence has prevented the swift 4 in a row by dropping their counter in column1 as I had column 5 and 1 open with 3 red counters between them the artificial intelligence has lost regardless in what they do but for experiment wise I will allow the artificial intelligence to prevent the win in column5. This might not happen though as the artificial intelligence also need to role a certain number in order to execute the code which searches for a 4 in a row which is there to simulate human error.

Vertical and horizontal prevention



As shown before it placed a yellow counter in column5 to prevent the horizontal win. After trying to get a vertical 4 in a row the artificial intelligence once again has prevented this from happening by dropping a counter in column2 as well, challenging the player significantly more than the easy artificial intelligence.

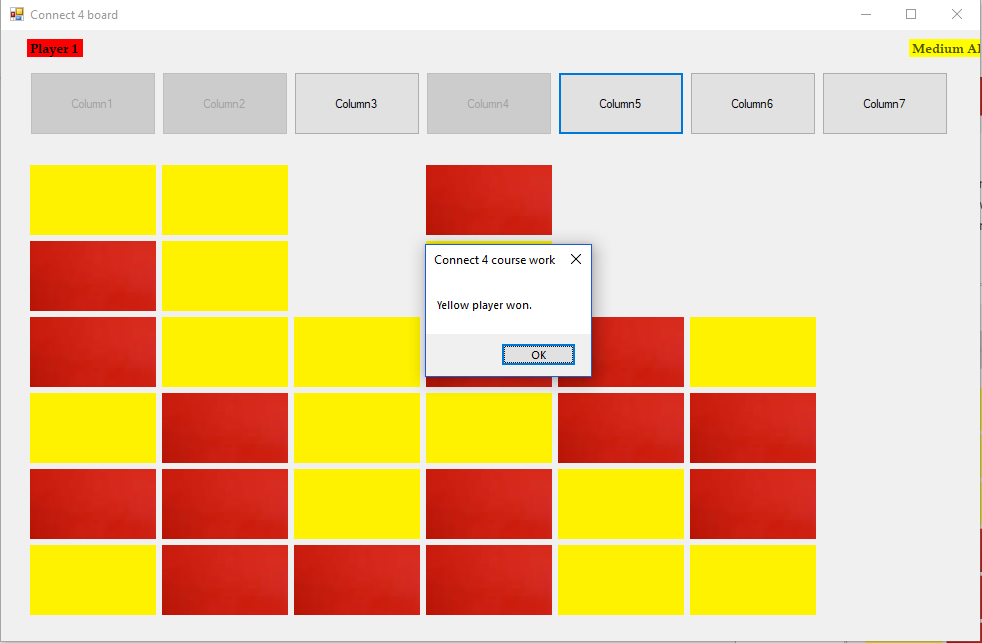


After skipping a few turns, I have reached a point where the artificial intelligence has slipped up and haven’t rolled the correct value to counter the 4 in a row diagonal right starting a column3.However for testing purposes I won't end this and else see if the artificial intelligence either sees that they can win in column3 or if they counter me as the turn before they dropped a counter in column3.

Right diagonal prevention

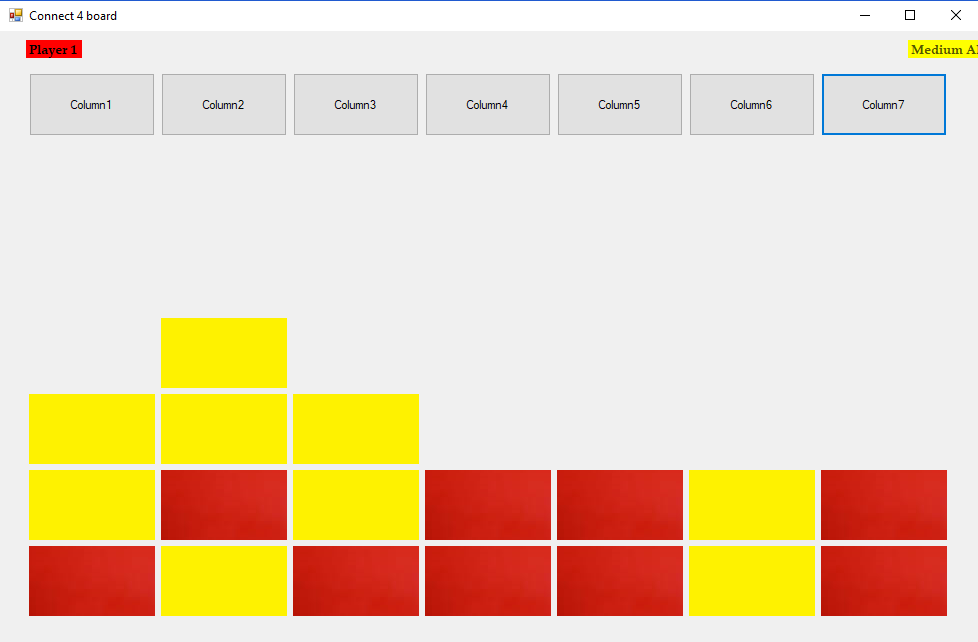


The artificial intelligence has recognised that I could win by dropping a counter in column 6 so they have dropped a yellow one there instead, ignoring that they could win in column 3 which means they have rolled incorrectly again preventing them from placing a counter.



Eventually the artificial intelligence has recognised it can win placing a counter in column 3 to get a 5 in a row yellow counter win. So in this experiment we have successfully shown that the medium artificial intelligence prevents the red player from winning vertical, horizontal and diagonally. The next test will be to show that it is competent in winning horizontally and vertically as its been shown to identify diagonal wins.

Horizontal win

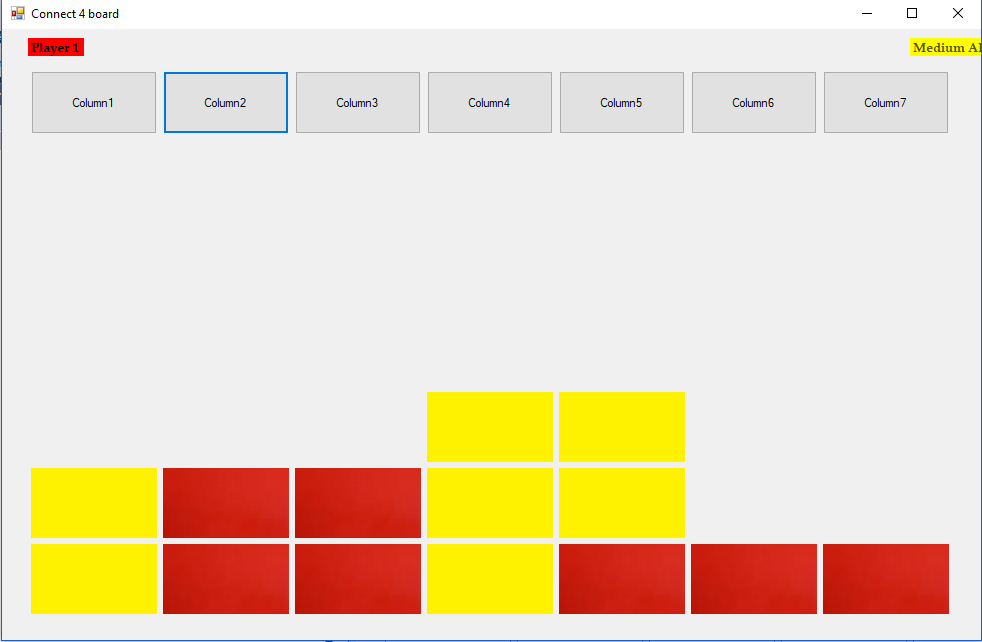


Currently the artificial intelligence has got a 3 in a row along the 3rd row so if I drop a counter on column 6 the medium artificial intelligence assuming it gets the correct value from the rng to enable dropping a counter.

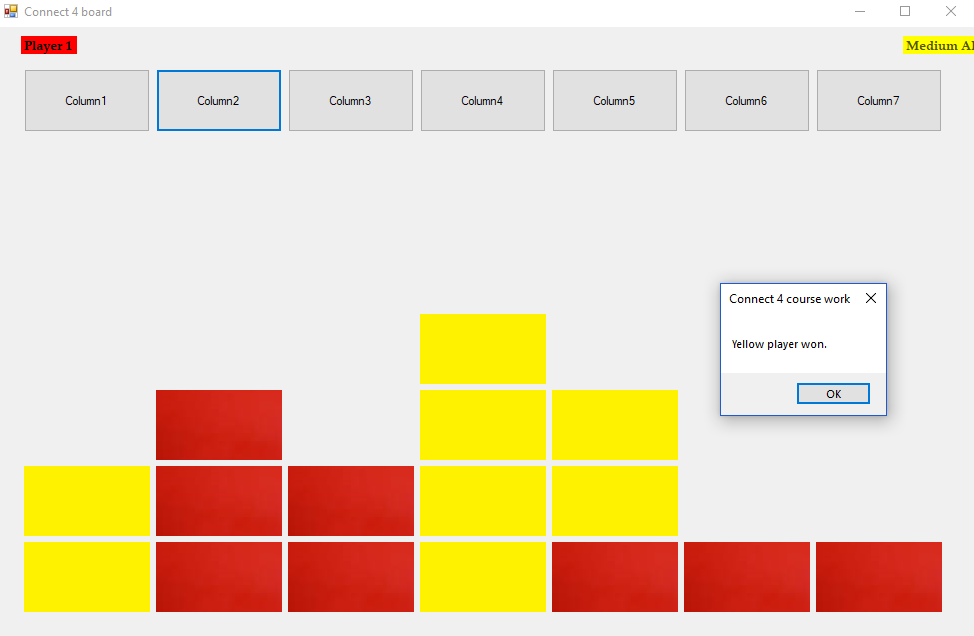


The artificial intelligence has indeed once I dropped a counter in column6 has dropped theirs in column4, showing that they have indeed acknowledged the existence of the 4 in a row and have won the game. If I had placed my counter in column 5 or 7 to get a vertical 3 in a row, then assuming they roll the correct value will still drop a counter in column4 as it will prioritise winning the game over preventing the other player from winning.

Vertical win



Now that the artificial intelligence has gotten a 3 yellow counters vertically in column4 I can now present if I drop a counter in column 2 that if the artificial intelligence gets the right integer will drop a counter in column4 to secure the win rather than stopping a 4 in a row for red counters.



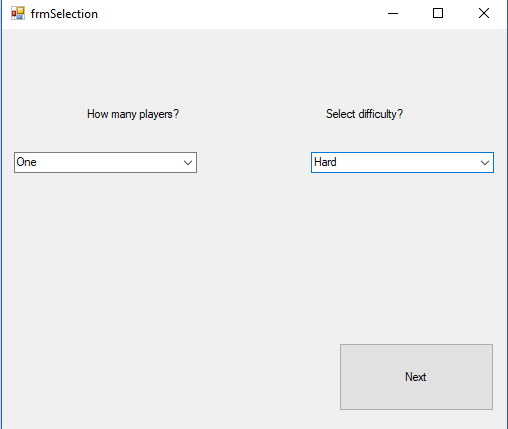
So, the artificial intelligence has rolled correctly in getting the vertical win in column 4. if they didn’t then the artificial intelligence would’ve skipped dropping the counter in column4 and instead looked at column2 as there’s 3 red counters vertically and if it rolled the correct integer then it would’ve stopped the player.

Comparing these screen shots to the success criteria you can see that the program now has a functioning medium artificial intelligence. So knowing this we can check off ‘medium artificial intelligence is functioning’.

|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game | **YES** |
| Inputted data is in correct data type e.g only characters for names | **YES** |
| Allows user to input into database | **YES** |
| Database can retrieve relevant information and pass it to program | **YES** |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form | **YES** |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) | **YES** |
| User can select Artificial intelligences difficulty | **YES** |
| ‘Register’ button inputs data within the database and displays it | **YES** |
| ‘Drop counter’ button allows the user to drop the counter | **YES** |
| Winning conditions can be met and is viable | **YES** |
| Artificial intelligence is present within the game | **YES** |
| Artificial intelligence with simple coding works | **YES** |
| Medium difficult artificial intelligence is functioning | **YES** |
| Hard difficulty artificial intelligence is functioning |  |
| The game terminates after the user is finished with program | **YES** |

### Hard artificial intelligence

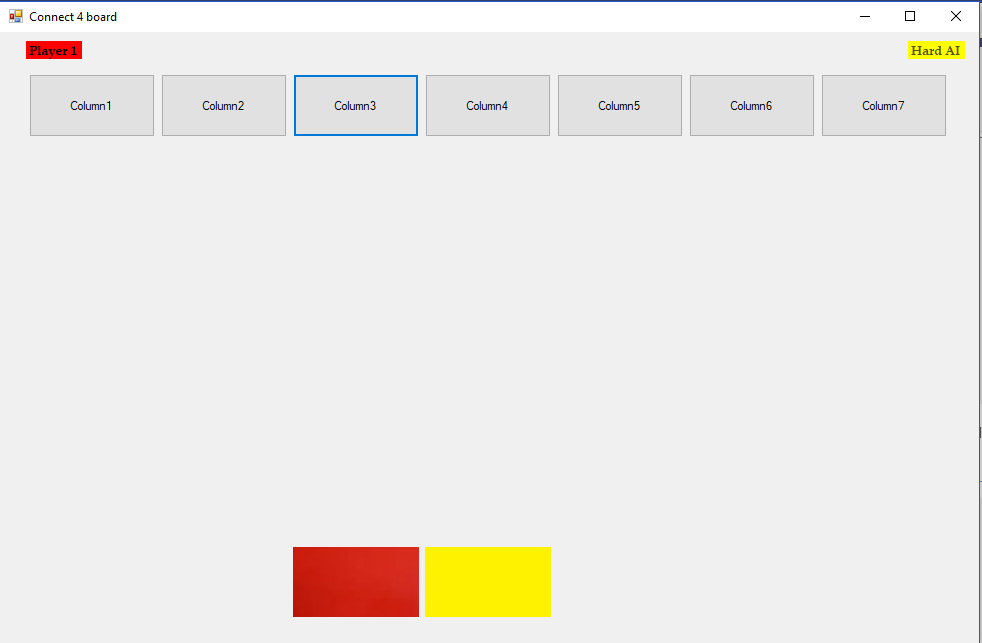
This difficulty will have similar coding to the medium difficulty except the number they have to roll to execute some of the code is significantly lower which will represent a more difficult opponent that will hardly miss a 4 in a row. The hard-artificial intelligence has also been coded to prevent getting in a situation of getting 3 counters in the middle with two free slots on either side allowing the player to get a 4 in a row on either side.



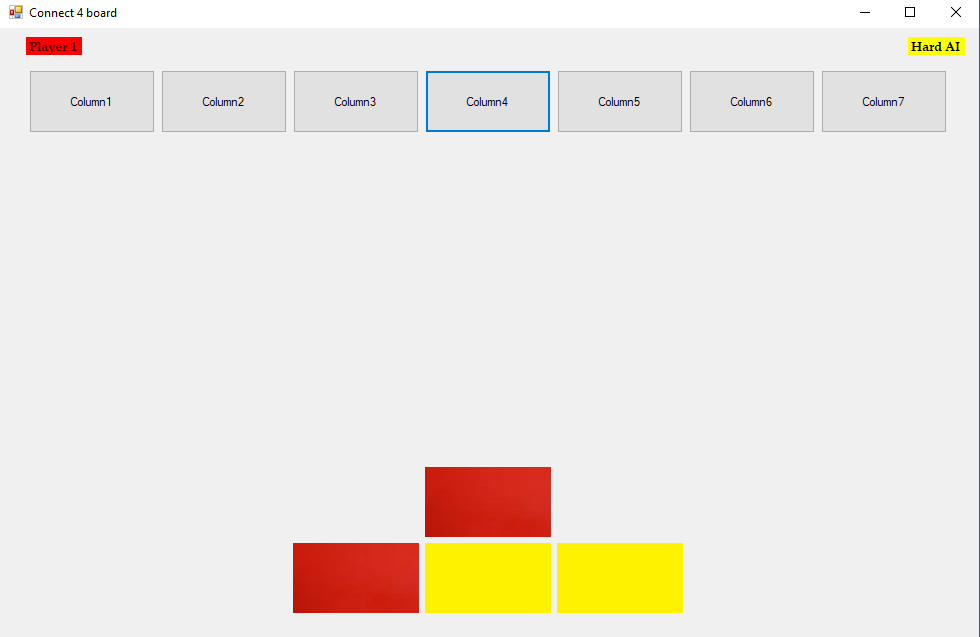
Firstly this will have to be selected in order for the connect 4 game to even run the artificial intelligence.



Like before the artificial intelligences difficulty will be clearly represented in the top right highlighted by the colour of its counter.



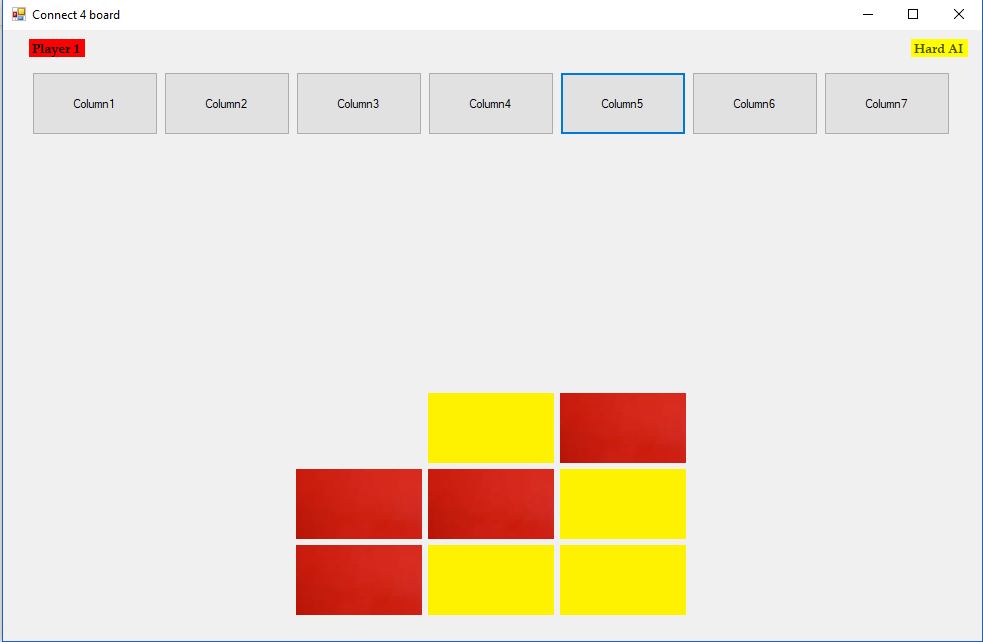
Strategically in connect 4 if you control the middle of the board then you have a higher chance of winning. Knowing this, the artificial intelligence is coded if the middle is clear and the surrounding areas then it will try and get as much of it as possible in the starting moves.



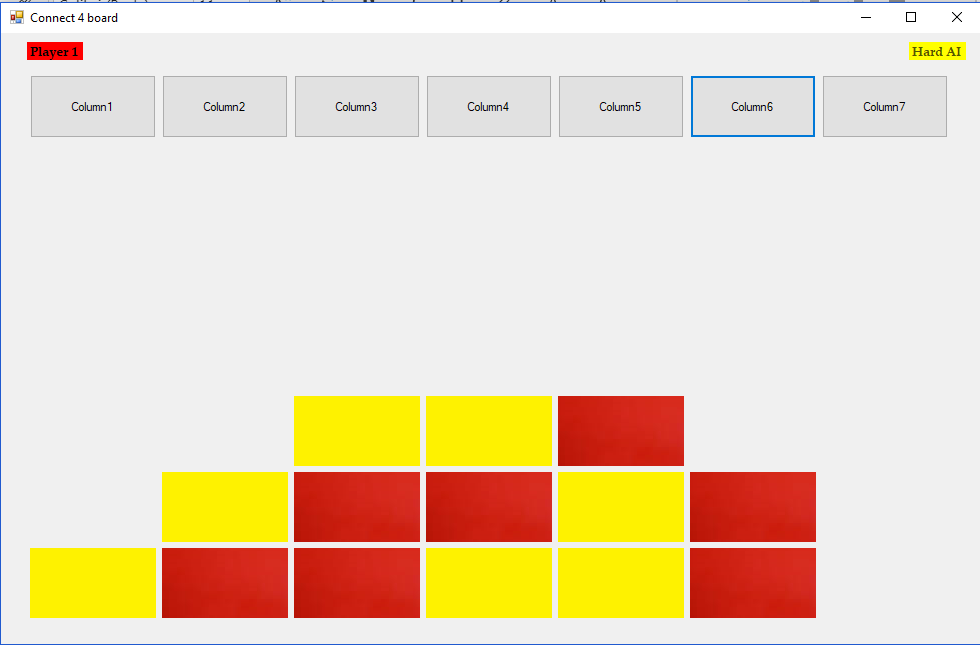
As I dropped a counter in column4 the artificial intelligence dropped theirs in column 5 to try to get equal mid domination.



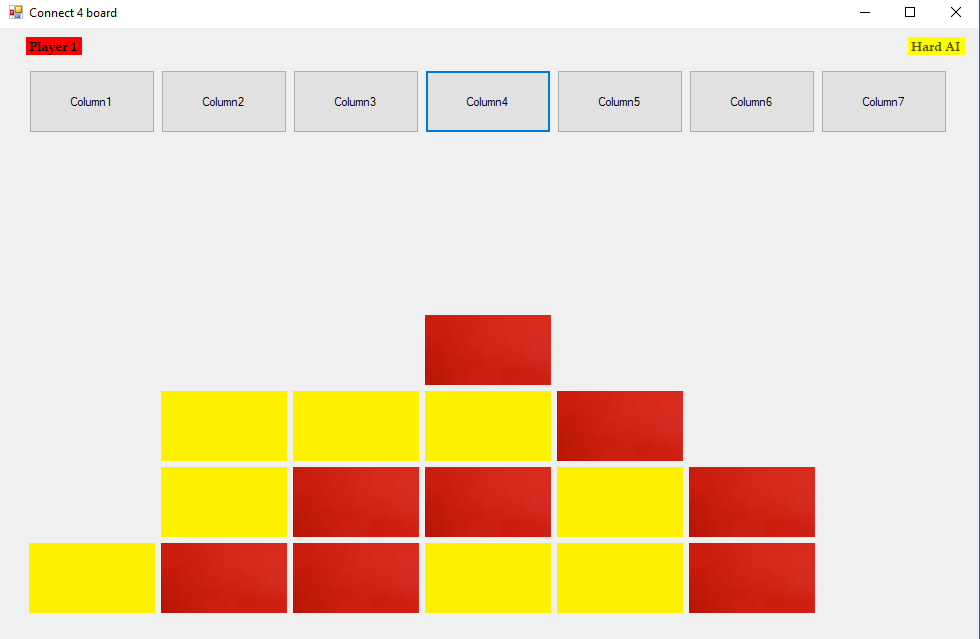
Once I dropped a counter in column3 the AI drops there's on column 5 so they maintain an equal slice of the middle to get a greater chance of winning.



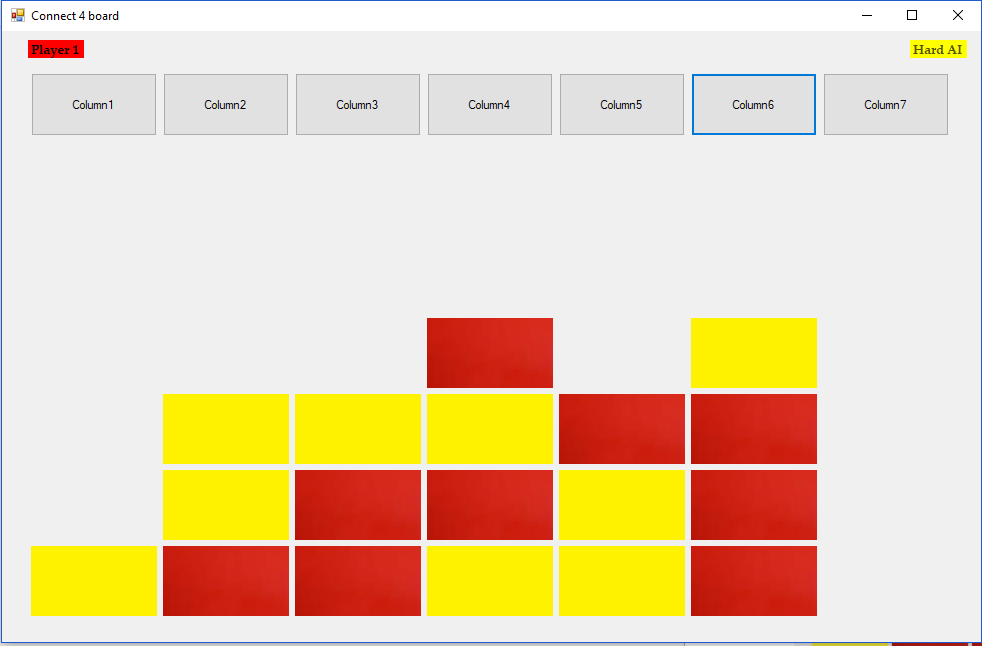
They maintain that middle conflict



Skipping a few turns now the artificial has swiftly got a chance to win diagonally by using my counters as stepping stones to it.



So as I countered the artificial intelligence they have not gotten a low enough roll in order to drop a counter on column7 which is a 1/30 chance of happening so this doesn’t often occur.



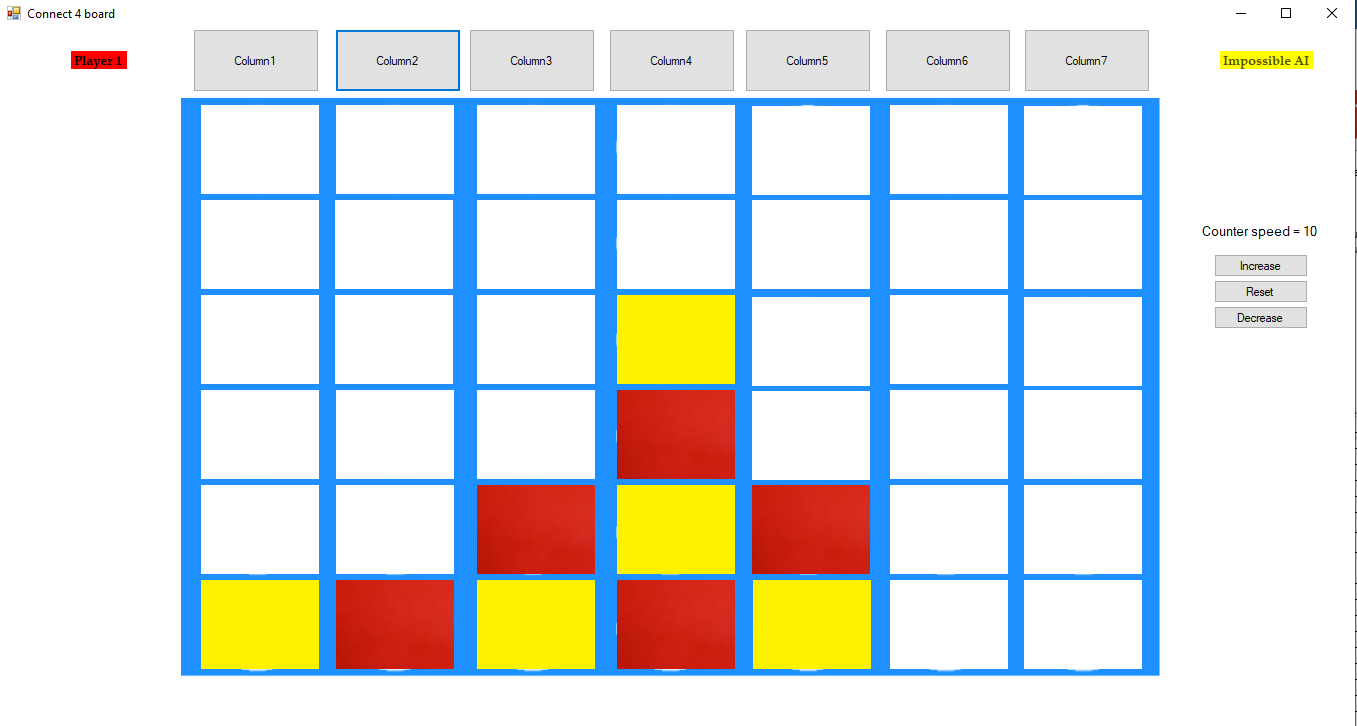
After getting a 3 in a row vertically in column 6 the artificial intelligence has prioritised that as I coded it to do the most seen 4 in a row first as you might have missed the diagonal 4 in a row. This game ended as a draw as I couldn’t present anything else I didn’t take any more screen shots



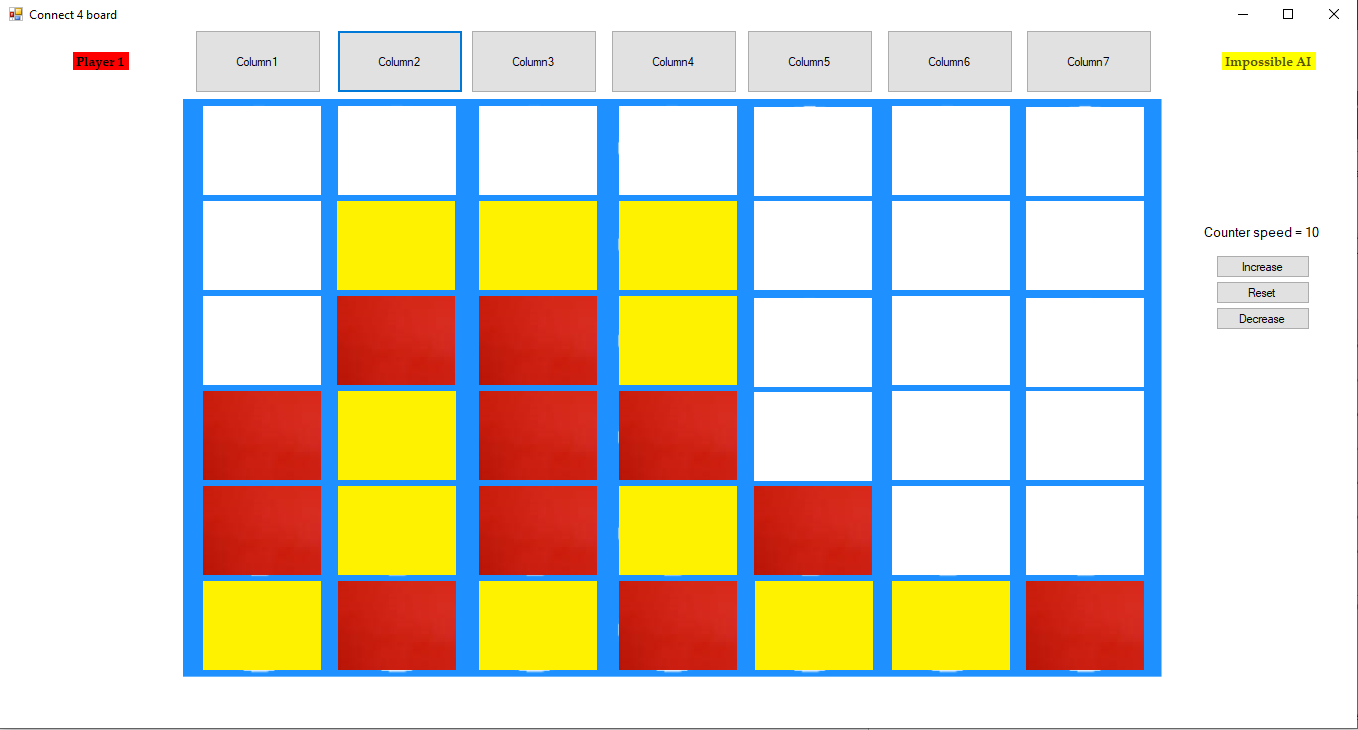
The only example I could get of the artificial intelligence preventing a 5 in a row doing all other solutions until it was forced to drop its counter in column 2 giving red player the win.

(After graphics added)

Testing for the prediction algorithm (Using impossible AI as it doesn’t rely on RNG to run through the code. The coding is the same to hard hence why there’s no individual impossible difficulty



The AI will avoid dropping any counters in column 5 due to me being able to win so if we play on it will be apparent that they aren’t dropping any counter there.



In column 1 the AI has a potential 4 in a row horizontally but isn’t gonna place their counter into column 1 first unless they must. As you can see the after the first screen shot the counters in column 5 haven’t progressed at all due to my chance of winning. Theres no further screen shots of this game due to me losing 2 counters later as I missed there left diagonal placing a counter in column 6 and column 5 to win disregarding my chance to win. So this is some evidence of this function working.

After exhaustive testing we can say that the hard artificial intelligence functions as intended

|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game | **YES** |
| Inputted data is in correct data type e.g only characters for names | **YES** |
| Allows user to input into database | **YES** |
| Database can retrieve relevant information and pass it to program | **YES** |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form | **YES** |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) | **YES** |
| ‘Register’ button inputs data within the database and displays it | **YES** |
| User can select Artificial intelligences difficulty | **YES** |
| ‘Drop counter’ button allows the user to drop the counter | **YES** |
| Winning conditions can be met and is viable | **YES** |
| Artificial intelligence is present within the game | **YES** |
| Artificial intelligence with simple coding works | **YES** |
| Medium difficult artificial intelligence is functioning | **YES** |
| Hard difficulty artificial intelligence is functioning | **YES** |
| The game terminates after the user is finished with program | **YES** |

### Success criteria

|  |  |
| --- | --- |
| Things to check | Does it function? |
| Login button searches within the database and moves to game | **YES** |
| Inputted data is in correct data type e.g only characters for names | **YES** |
| Allows user to input into database | **YES** |
| Database can retrieve relevant information and pass it to program | **YES** |
| ‘Register now’ button redirects user to register form | **YES** |
| ‘Back’ button returns user to login screen from the register form | **YES** |
| Allow the user to enter details in register form (If anything is duplicate data in database it will mention it) | **YES** |
| ‘Register’ button inputs data within the database and displays it | **YES** |
| User can select Artificial intelligences difficulty | **YES** |
| ‘Drop counter’ button allows the user to drop the counter | **YES** |
| Winning conditions can be met and is viable | **YES** |
| Artificial intelligence is present within the game | **YES** |
| Artificial intelligence with simple coding works | **YES** |
| Medium difficult artificial intelligence is functioning | **YES** |
| Hard difficulty artificial intelligence is functioning | **YES** |
| The game terminates after the user is finished with program | **YES** |

The success criteria is fully complete and tested for any errors.

## Feedback

### Stakeholders

I sent over my finalised program to the stakeholders in which they responded with

Joanne Jackson: “This is perfect in teaching my pupils about artificial intelligence and a simple user interface where anybody using the program can have a easy time playing the game. The database to store all the student's information is simplistic and works as intended allowing a better management of the users.”

Overall the stakeholders are satisfied with the final product, pleased with the level of difficulty within the ranging categories.

### Clients

Questions:

1. Do you find the program useful?
2. Do you find the artificial intelligence difficult?
3. Do you like the login system?
4. What would you improve?
5. Would you use it again?

Answers:

**Jack:**

1. I found the program to be enjoyable and useful to use when it came to learn more about artificial intelligence with replay ability as the ranging difficulties gave me a new challenge to beat them.
2. I found the easy AI to be very easy to defeat as it seemed they just placed counters anywhere for no reason sometimes beating them in 4 turns. The medium AI really gave me a challenge as I thought it would be simple as the easy AI posed no threat, but I found I had to think more to win with them missing sometimes which I found good representation to real life. The Hard AI was difficult to beat as anything you would do it would immediately counter and if you missed anything then you would lose instantly so you couldn’t make a mistake.
3. The login system did its function and I saw nothing that you could’ve added to make it any better.
4. I would improve the games graphics as I found them very weird to the game of connect 4 and didn’t really fit it, finding it hard sometimes to find a 4 in a row diagonally.
5. I would as I want to beat my friends score on the hard difficulty as he is currently number 1 so having that leaderboard has challenged me to beat the ai in as

## Future endeavours

As my clients are keen in constant development on this project as they believe “It will provide a better understanding and appreciation of artificial intelligence” stating that there are graphical enhancements to be made and the a better development of artificial intelligence can always be made so continuation of the project are planned. The motion with the object can be altered in order to function with the artificial intelligence and moving object. The game is robust in functions so it could us with some implementations of features, one idea was that after getting a 4 in a row the game wouldn’t end, instead the counters above it would drop down to, by however many slots and the player which got the 4 in a row would score some points, receiving different amounts depending on the difficulty of the 4/5 in a row.