PROJECT REPORT

On

**“Connect 4 Game”**

***Submitted By***

Ms. Pooja Sanjay Makhe

***Guided By***

Mr. Ratnesh K. Choudhary



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**S. B. JAIN INSTITUTE OF TECHNOLOGY MANAGEMENT AND RESEARCH, NAGPUR.**

(An Autonomous Institute, Affiliated to RTMNU, Nagpur)

**2021-2022**

**© S.B.J.I.T.M.R Nagpur 2022**

**S.B. JAIN INSTITUTE OF TECHNOLOGY MANAGEMENT AND RESEARCH, NAGPUR**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

SESSION 2021-2022

#### CERTIFICATE

This is to certify that the Project titled **“Connect 4 Game”** is a bonafide work of **Pooja Makhe** carried out for the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering in **Computer Science & Engineering.**

|  |  |  |
| --- | --- | --- |
| **Mr. Ratnesh K. Choudhary**  Assistant Professor |  | **Mr. Animesh Tayal**  Head of Department |

**INDEX**

|  |  |  |
| --- | --- | --- |
| CERTIFICATE | | i |
| INDEX | | ii |
| LIST OF FIGURES | | iii |
|  |  |  |
| CHAPTER 1 | INTRODUCTION | 1-3 |
| CHAPTER 2 | METHODOLOGY | 4 |
| CHAPTER 3 | TOOLS/PLATFORMS | 5-7 |
| CHAPTER 4 | DESIGN & IMPLEMENTATION | 8-14 |
|  | 4.1 ALGORITHM |  |
|  | 4.2 FLOWCHART |  |
|  | 4.3 SOURCE CODE |  |
| CHAPTER 5 | RESULT & DISCUSSION | 15-19 |
|  | 5.1 OUTPUT |  |
|  | 5.2 DISCUSSION |  |
|  | 5.3 APPLICATION |  |
| CHAPTER 6 | CONCLUSION | 20 |
|  | REFERENCES | 21 |

**LIST OF FIGURE**

|  |  |  |
| --- | --- | --- |
| **FIG. NO.** | **TITLE OF FIGURE** | **PAGE NO.** |
| 1.1 | Starting screen of this game | 1 |
| 1.2 | The physical structure | 2 |
| 4.2.1 | The flowchart | 9 |
| 5.1.1 | Winning position | 18 |
| 5.1.2 | Block position | 18 |
| 5.1.3 | Running position | 19 |

**CHAPTER 1**

**INTRODUCTION**

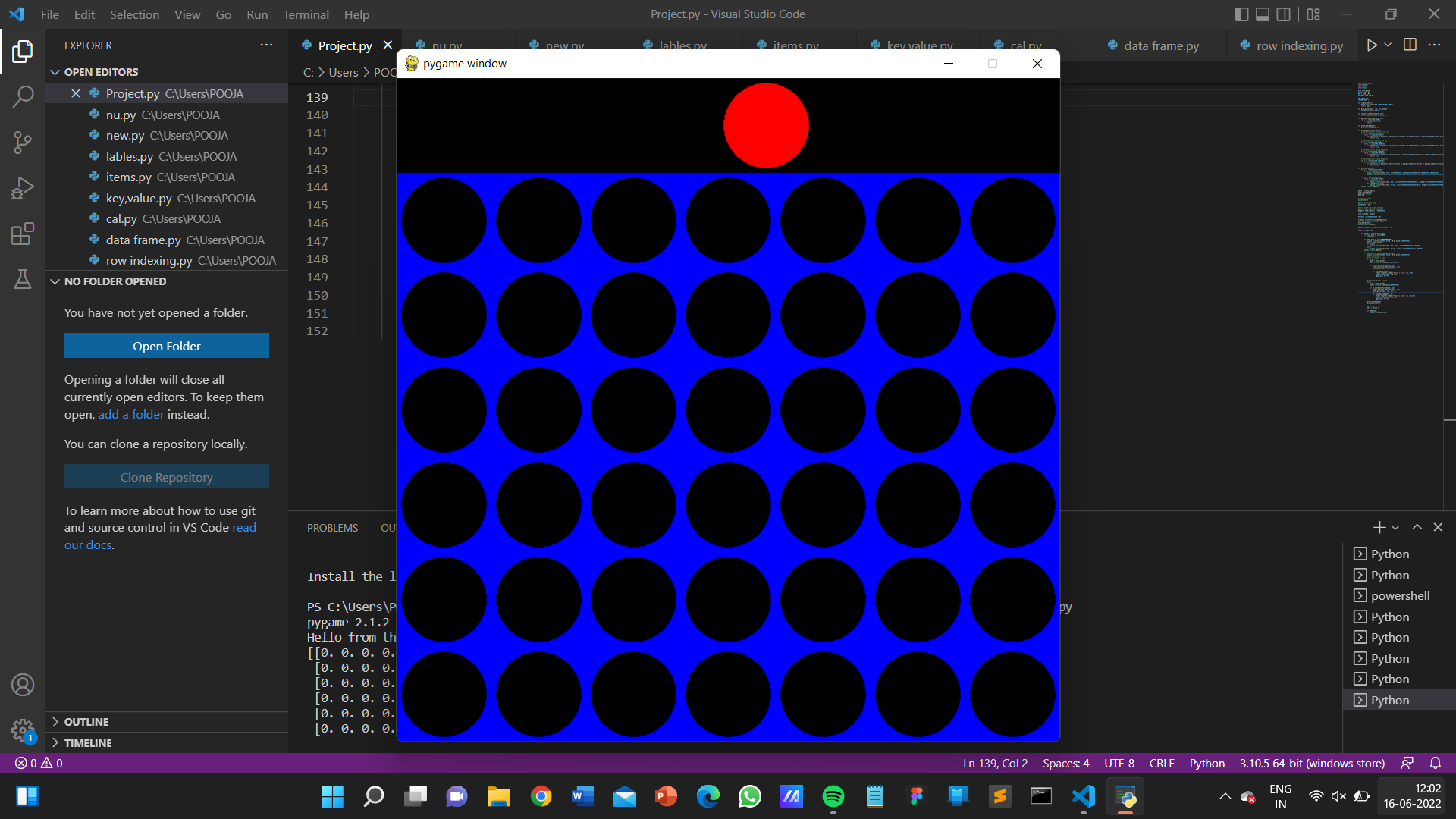
Before starting off, let us first understand what connect Four game actually is.

Connect4 game is also known as Four Up, Plot Four, Find Four, Captain’s Mistress, Four in a Row, Drop Four, and Gravitrips in the Soviet Union.It is a two-player connection board game, in which the players choose a color and then take turns dropping colored discs into a seven-column, six-row vertically suspended grid.

The pieces fall straight down, occupying the lowest available space within the column.

The objective of the game is to be the first to form a horizontal, vertical, or diagonal line of four of one’s own discs. Connect Four is a solved game.

This game is for to kill our boredom and its just a fun to play it.

**Fig 1.1** Starting screen of this game

The player one can win the game easily, but have to play wisely to win it and the player 2 have to compete against the player 1 to win the game.

Both player have to understand the rule that the one who first connect 4 dots in row or column or diagonally the one can win the game.

## **Rule variations**

There are many variations of Connect Four with differing game board sizes, game pieces, and gameplay rules. Many variations are popular with game theory and artificial intelligence research, rather than with physical game boards and gameplay by persons.

The most commonly-used Connect Four board size is 7 columns × 6 rows. Size variations include 5×4, 6×5, 8×7, 9×7, 10×7, 8×8, Infinite Connect-Four, and Cylinder-Infinite Connect-Four.

[](https://en.wikipedia.org/wiki/File:Connect_Four.jpg)

**Fig 1.2:** The Physical structure

A travel version of the Milton Bradley game.

Several versions of Hasbro's Connect Four physical gameboard make it easy to remove game pieces from the bottom one at a time. Along with traditional [gameplay](https://en.wikipedia.org/wiki/Gameplay), this feature allows for variations of the game. Some earlier game versions also included specially-marked discs, and cardboard column extenders, for additional variations to the game.

### Pop-Out

*"Pop-Out" redirects here. For other uses, see*[*Pop Out (disambiguation)*](https://en.wikipedia.org/wiki/Pop_Out_(disambiguation))*.*

*Pop-Out* starts the same as traditional gameplay, with an empty board and players alternating turns placing their own colored discs into the board. During each turn, a player can either add another disc from the top, or if one has any discs of their own color on the bottom row, remove (or "pop out") a disc of one's own color from the bottom. Popping a disc out from the bottom drops every disc above it down one space, changing their relationship with the rest of the board and changing the possibilities for a connection. The first player to connect four of their discs horizontally, vertically, or diagonally wins the game.

### Pop 10

Before play begins, *Pop 10* is set up differently from the traditional game. Taking turns, each player places one of their own color discs into the slots filling up only the bottom row, then moving on to the next row until it is filled, and so forth until all rows have been filled.

Gameplay works by players taking turns removing a disc of one's own color through the bottom of the board. If the disc that was removed was part of a four-disc connection at the time of its removal, the player sets it aside out of play and immediately takes another turn. If it was not part of a "connect four", then it must be placed back on the board through a slot at the top into any open space in an alternate column (whenever possible) and the turn ends, switching to the other player. The first player to set aside ten discs of their color wins the game.

### Five-in-a-Row

The *Five-in-a-Row* variation for Connect Four is a game played on a 6 high, 9 wide grid. Two additional board columns, already filled with player pieces in an alternating pattern, are added to the left and right sides of the standard 6-by-7 game board. The game plays similarly to the original Connect Four, except players must now get five pieces in a row to win. This is still a 42-ply game since the two new columns added to the game represent twelve game pieces already played, before the start of a game.

### Power Up

### In this variation of Connect Four, players begin a game with one or more specially-marked "Power Checkers" game pieces, which each player may choose to play once per game. When playing a piece marked with an anvil icon, for example, the player may immediately pop out all pieces below it, leaving the anvil piece at the bottom row of the game board. Other marked game pieces include one with a wall icon, allowing a player to play a second consecutive non-winning turn with an unmarked piece; a "×2" icon, allowing for an unrestricted second turn with an unmarked piece; and a bomb icon, allowing a player to immediately pop out an opponent's piece.

**CHAPTER 2**

**METHODOLOGY**

Connect Four is a two-player game with perfect information for both sides, meaning that nothing is hidden from anyone. Connect Four also belongs to the classification of an adversarial, zero-sum game, since a player's advantage is an opponent's disadvantage.

One measure of complexity of the Connect Four game is the number of possible games board positions. For classic Connect Four played on a 7-column-wide, 6-row-high grid, there are 4,531,985,219,092 positions for all game boards populated with 0 to 42 pieces.

The game was first solved by James Dow Allen (October 1, 1988), and independently by Victor Allis (October 16, 1988). Allis describes a knowledge-based approach, with nine strategies, as a solution for Connect Four. Allen also describes winning strategies in his analysis of the game. At the time of the initial solutions for Connect Four, brute-force analysis was not deemed feasible given the game's complexity and the computer technology available at the time.

Connect Four has since been solved with brute-force methods, beginning with John Tromp's work in compiling an 8-ply database (February 4, 1995). The artificial intelligence algorithms able to strongly solve Connect Four are minimax or negamax, with optimizations that include alpha-beta pruning, move ordering, and transposition tables. The code for solving Connect Four with these methods is also the basis for the Fhourstones integer performance benchmark.

The solved conclusion for Connect Four is first-player-win. With perfect play, the first player can force a win, on or before the 41st move by starting in the middle column. The game is a theoretical draw when the first player starts in the columns adjacent to the center. For the edges of the game board, column 1 and 2 on left (or column 7 and 6 on right), the exact move-value score for first player start is loss on the 40th move, and loss on the 42nd move, respectively. In other words, by starting with the four outer columns, the first player allows the second player to force a win.

**CHAPTER 3**

**TOOLS/PLATFORMS**

**3.1 SOFTWARE REQUIREMENT**

1. **SOFTWARE:** Python3
2. **Libraries:** numpy, pygame, python sys, python math
3. **operating system:** windows 11
4. **IDE:** Visual code

**Python3**

**Python** is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020.

Python consistently ranks as one of the most popular programming languages.

**Libraries are as follows which are used in project:**

***1. NumPy module***

NumPy Library: NumPy stands for Numerical Python. NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. It is an open source project and you can use it freely. NumPy is a Python library that provides a simple yet powerful data structure: the n-dimensional array.

If you don’t have NumPy already pre-installed on your system, type the following command in your window’s cmd:

C:\Users\Admin> pip install numpy

When you call the statement import numpy as np , you are shortening the phrase “numpy” to “np” to make your code easier to read. It also helps to avoid namespace issues.

|  |
| --- |
| import numpy as np |

***2. Pygame module***

Pygame is a free and open-source cross-platform library for the development of multimedia applications like video games using Python.

It uses the Simple DirectMedia Layer library and several other popular libraries to abstract the most common functions, making writing these programs a more intuitive task.

If you don’t have Pygame already pre-installed on your system, type the following command in your window’s cmd:

|  |
| --- |
| C:\Users\Admin> pip install numpy |

***3. Python sys module***

The python sys module provides functions and variables which are used to manipulate different parts of the Python Runtime Environment. It lets us access system-specific parameters and functions. import sys. First, we have to import the sys module in our program before running any functions. sys.modules.

***4. Python math module***

Some of the most popular mathematical functions are defined in the math module. These include trigonometric functions, representation functions, logarithmic functions, angle conversion functions, etc. In addition, two mathematical constants are also defined in this module.

If you don’t have math already pre-installed on your system, type the following command in your window’s cmd:

|  |
| --- |
| C:\Users\Admin> pip install maths |

**Operating system:** Windows 11

**Windows 11** is the latest major release of Microsoft's Windows NT operating system, released in October 2021. It is a free upgrade to its predecessor, Windows 10 (2015), available for any Windows 10 devices that meet the new Windows 11 system requirements.

Windows 11 features major changes to the Windows shell influenced by the canceled Windows 10X, including a redesigned Start menu, the replacement of its "live tiles" with a separate "Widgets" panel on the taskbar, the ability to create tiled sets of windows that can be minimized and restored from the taskbar as a group, and new gaming technologies inherited from Xbox Series X and Series S such as Auto HDR and DirectStorage on compatible hardware. Internet Explorer (IE) has been replaced by the Chromium-based Microsoft Edge as the default web browser like its predecessor, Windows 10, and Microsoft Teams is integrated into the Windows shell. Microsoft also announced plans to allow more flexibility in software that can be distributed via Microsoft Store, and to support Android apps on Windows 11 (including a partnership with Amazon to make its app store available for the function).

**IDE:** Visual Code

**Visual Studio Code**, also commonly referred to as **VS Code**, is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

In the Stack Overflow 2021 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 70% of 82,000 respondents reporting that they use it.

**CHAPTER 4**

**DESIGN & IMPLEMENTATION**

**4.1 ALGORITHM**

**Step1:** Import the NumPy package as np. Then we will create a python function named create\_board( )**.** np.zeros( ) function

**Step2:** In the step 02, we make a few alterations and updates to the previous code.

We want the selection variable to actually drop a piece on the board. For that, the first thing we will do is create another three functions called def drop\_piece( ), def is\_valid\_location( ), def get\_next\_open\_row( ).

selection variable to column variable (col).

Syntax: np.flip(array, axis)

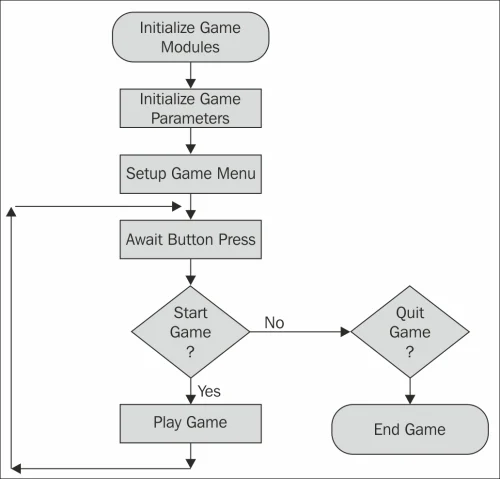
**Step3:** In step 03, we will create a game with a GUI and not just the one with the matrices. The above code along with the new modifications that we do will make the game look like an actual board game.

create\_board( ), drop\_piece( ), is\_valid\_location( ), get\_next\_open\_row( ) and print\_board( ).

 def draw\_board( ), pygame.draw is a module for drawing shapes.

**Step4:** stop

**4.2 FLOWCHART**

****

**Fig4.2.1** The Flow chart

**4.3 SOURCE CODE**

import numpy as np

import pygame

import sys

import math

BLUE = (0,0,255)

BLACK = (0,0,0)

RED = (255,0,0)

YELLOW = (255,255,0)

ROW\_COUNT = 6

COLUMN\_COUNT = 7

def create\_board():

    board = np.zeros((ROW\_COUNT,COLUMN\_COUNT))

    return board

def drop\_piece(board, row, col, piece):

    board[row][col] = piece

def is\_valid\_location(board, col):

    return board[ROW\_COUNT-1][col] == 0

def get\_next\_open\_row(board, col):

    for r in range(ROW\_COUNT):

        if board[r][col] == 0:

            return r

def print\_board(board):

    print(np.flip(board, 0))

def winning\_move(board, piece):

    # Check horizontal locations for win

    for c in range(COLUMN\_COUNT-3):

        for r in range(ROW\_COUNT):

            if board[r][c] == piece and board[r][c+1] == piece and board[r][c+2] == piece and board[r][c+3] == piece:

                return True

    # Check vertical locations for win

    for c in range(COLUMN\_COUNT):

        for r in range(ROW\_COUNT-3):

            if board[r][c] == piece and board[r+1][c] == piece and board[r+2][c] == piece and board[r+3][c] == piece:

                return True

    # Check positively sloped diaganols

    for c in range(COLUMN\_COUNT-3):

        for r in range(ROW\_COUNT-3):

            if board[r][c] == piece and board[r+1][c+1] == piece and board[r+2][c+2] == piece and board[r+3][c+3] == piece:

                return True

    # Check negatively sloped diaganols

    for c in range(COLUMN\_COUNT-3):

        for r in range(3, ROW\_COUNT):

            if board[r][c] == piece and board[r-1][c+1] == piece and board[r-2][c+2] == piece and board[r-3][c+3] == piece:

                return True

def draw\_board(board):

    for c in range(COLUMN\_COUNT):

        for r in range(ROW\_COUNT):

            pygame.draw.rect(screen, BLUE, (c\*SQUARESIZE, r\*SQUARESIZE+SQUARESIZE, SQUARESIZE, SQUARESIZE))

            pygame.draw.circle(screen, BLACK, (int(c\*SQUARESIZE+SQUARESIZE/2), int(r\*SQUARESIZE+SQUARESIZE+SQUARESIZE/2)), RADIUS)

    for c in range(COLUMN\_COUNT):

        for r in range(ROW\_COUNT):

            if board[r][c] == 1:

                pygame.draw.circle(screen, RED, (int(c\*SQUARESIZE+SQUARESIZE/2), height-int(r\*SQUARESIZE+SQUARESIZE/2)), RADIUS)

            elif board[r][c] == 2:

                pygame.draw.circle(screen, YELLOW, (int(c\*SQUARESIZE+SQUARESIZE/2), height-int(r\*SQUARESIZE+SQUARESIZE/2)), RADIUS)

    pygame.display.update()

board = create\_board()

print\_board(board)

game\_over = False

turn = 0

#initalize pygame

pygame.init()

#define our screen size

SQUARESIZE = 100

#define width and height of board

width = COLUMN\_COUNT \* SQUARESIZE

height = (ROW\_COUNT+1) \* SQUARESIZE

size = (width, height)

RADIUS = int(SQUARESIZE/2 - 5)

screen = pygame.display.set\_mode(size)

#Calling function draw\_board again

draw\_board(board)

pygame.display.update()

myfont = pygame.font.SysFont("monospace", 75)

while not game\_over:

    for event in pygame.event.get():

        if event.type == pygame.QUIT:

            sys.exit()

        if event.type == pygame.MOUSEMOTION:

            pygame.draw.rect(screen, BLACK, (0,0, width, SQUARESIZE))

            posx = event.pos[0]

            if turn == 0:

                pygame.draw.circle(screen, RED, (posx, int(SQUARESIZE/2)), RADIUS)

            else:

                pygame.draw.circle(screen, YELLOW, (posx, int(SQUARESIZE/2)), RADIUS)

        pygame.display.update()

        if event.type == pygame.MOUSEBUTTONDOWN:

            pygame.draw.rect(screen, BLACK, (0,0, width, SQUARESIZE))

            #print(event.pos)

            # Ask for Player 1 Input

            if turn == 0:

                posx = event.pos[0]

                col = int(math.floor(posx/SQUARESIZE))

                if is\_valid\_location(board, col):

                    row = get\_next\_open\_row(board, col)

                    drop\_piece(board, row, col, 1)

                    if winning\_move(board, 1):

                        label = myfont.render("Player 1 wins!!", 1, RED)

                        screen.blit(label, (40,10))

                        game\_over = True

            # # Ask for Player 2 Input

            else:

                posx = event.pos[0]

                col = int(math.floor(posx/SQUARESIZE))

                if is\_valid\_location(board, col):

                    row = get\_next\_open\_row(board, col)

                    drop\_piece(board, row, col, 2)

                    if winning\_move(board, 2):

                        label = myfont.render("Player 2 wins!!", 1, YELLOW)

                        screen.blit(label, (40,10))

                        game\_over = True

            print\_board(board)

            draw\_board(board)

            turn += 1

            turn = turn % 2

            if game\_over:

                pygame.time.wait(3000)

**CHAPTER 5**

**RESULT & DISCUSSION**

**5.1 OUTPUT**

PS C:\Users\POOJA> & C:/Users/POOJA/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/Users/POOJA/Project.py

pygame 2.1.2 (SDL 2.0.18, Python 3.10.4)

Hello from the pygame community. https://www.pygame.org/contribute.html

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 2. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]

[0. 0. 0. 2. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]

[0. 0. 0. 2. 0. 0. 0.]

[0. 0. 0. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]

[0. 0. 0. 2. 0. 0. 0.]

[0. 0. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]

[0. 0. 0. 2. 0. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]

[0. 0. 1. 2. 0. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 2. 1. 0. 0. 0.]

[0. 0. 1. 2. 0. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 2. 1. 0. 0. 0.]

[0. 1. 1. 2. 0. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 2. 2. 1. 0. 0. 0.]

[0. 1. 1. 2. 0. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]

[0. 2. 2. 1. 0. 0. 0.]

[0. 1. 1. 2. 0. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 1. 0. 0. 0.]

[0. 2. 2. 1. 0. 0. 0.]

[0. 1. 1. 2. 2. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 1. 1. 0. 0. 0.]

[0. 2. 2. 1. 0. 0. 0.]

[0. 1. 1. 2. 2. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

[[0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0.]

[0. 2. 1. 1. 0. 0. 0.]

[0. 2. 2. 1. 0. 0. 0.]

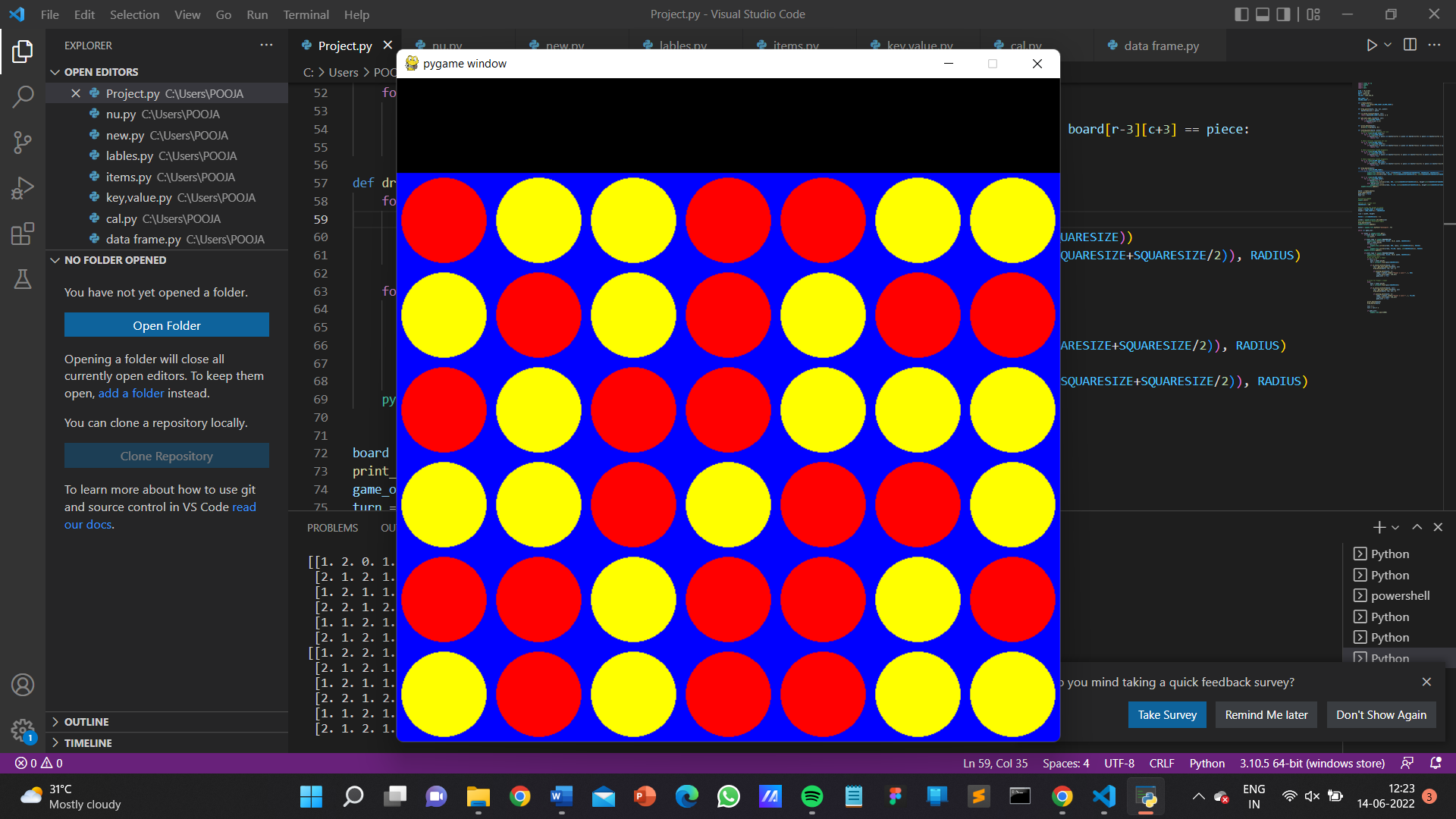
[0. 1. 1. 2. 2. 0. 0.]

[0. 2. 1. 1. 2. 0. 0.]]

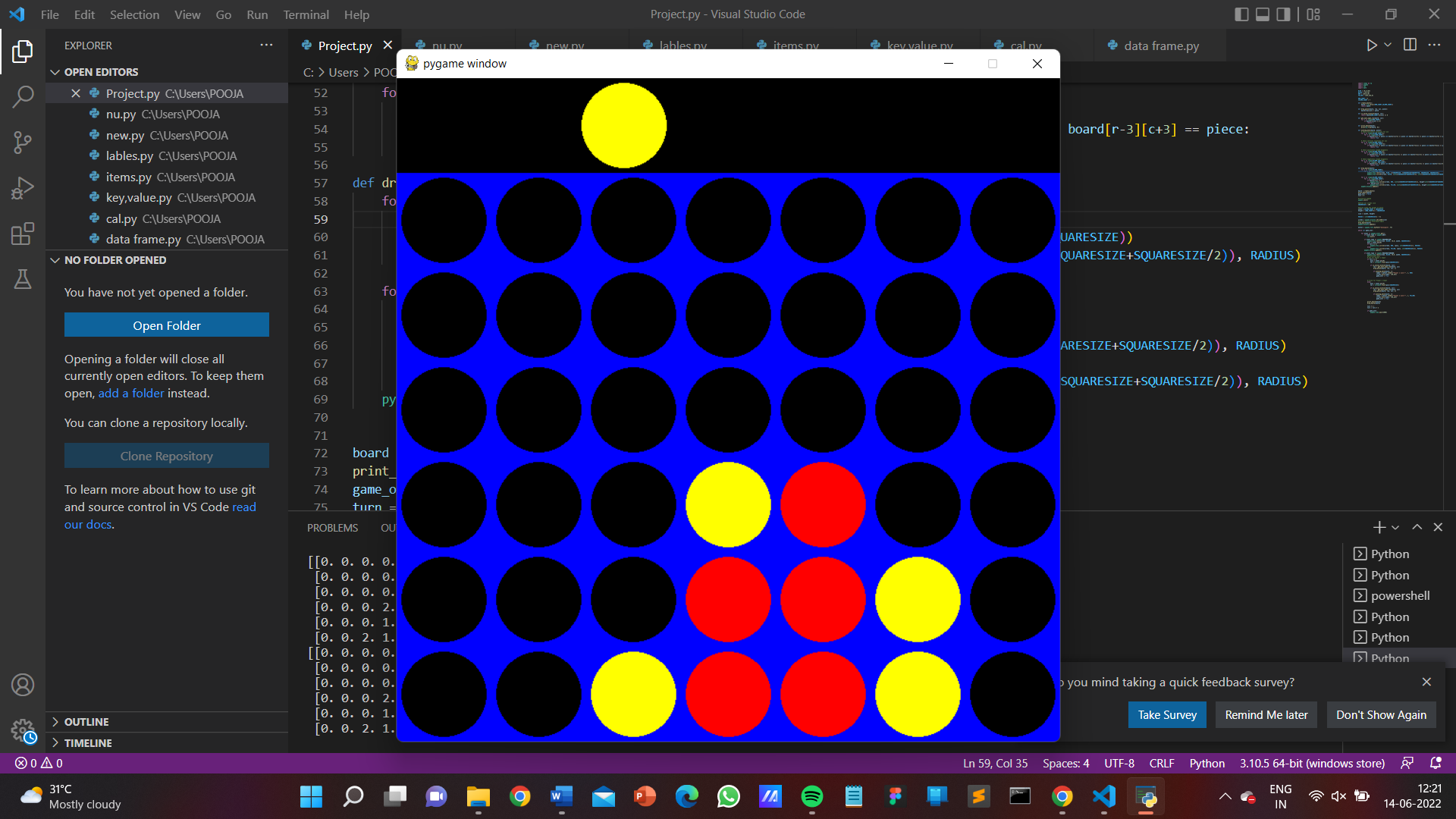
PS C:\Users\POOJA>

****

**Fig 5.1.1:** Winning Position

****

**Fig 5.1.2:** Blocked Position

****

**Fig 5.1.3:** Running State

**5.2 DISCUSSION**

In fig 5.1.1 we get the result of the player 2 wins by playing the game. In this fig player 2 wins the game easily as he played it wisely.

In fig 5.1.2 we can see that the both players play tough and no one is able to win the game. So in this fig we get to know that both player have the great IQ level and they played the very well for their win.

In fig 5.1.3 is the playing image of the game. The both players are playing the game.

**5.3 APPLICATION**

Connect4 game is also known as Four Up, Plot Four, Find Four, Captain's Mistress, Four in a Row, Drop Four, and Gravitrips in the Soviet Union. It is a two-player connection board game, in which the players choose a color and then take turns dropping colored discs into a seven-column, six-row vertically suspended grid.

**CHAPTER 6**

**CONCLUSION**

I have designed and developed the smart game “**CONNECT 4**” by using the python language. The game can be played by any age group like children and adults. It can use for entertain purpose. It can increase the IQ level of the human and also increase the thinking ability of the person who will play the game.

It can be also played on online platform.

**REFERENCE**

* Askpython.com
* Geeksforgonics.com
* The Complete Reference PYTHON by Brown Published by Mc Graw Hill

**Notes:**

**1. Chapter Title –** Front size = 16, Front Type = Times New Roman, Alignment = Centered,

Bold.

(Example – INTRODUCTION)

**2. Heading –** Front size = 14, Front Type = Times New Roman, Alignment = Left, Bold.

(Example – 4.1 ALGORITHM)

**3. Sub Heading -** Front size = 12, Front Type = Times New Roman, Alignment = Left, Bold.

(Example – Module)

**4. Content -** Front size = 12, Front Type = Times New Roman, Alignment = Left.

(Example – Write down proper information about you project)

**5. Figure –** Give proper name to your figure and screenshot.

Front size = 12, Front Type = Times New Roman, Alignment = Centered.

Fig. Chapter no. name of figure/screenshot

(Example – Fig. 4 Flowchart of Tic-Tac-Toe)