**Mini Project Report**

**On**

**Connect 4 Game in Python**

**(**CSE V Semester Mini Project**)**

***2022- 2023***



**Submitted to: Submitted by:**

Ms. Preeti Chaudhary                         Maninder Singh

(CC-CSE-A-V-Sem) University Roll No.:2018464

GEHU, D. Dun CSE-A-V-Sem

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**GRAPHIC ERA HILL UNIVERSITY, DEHRADUN**

**CERTIFICATE**

Certified that Mr. Maninder (Roll No.-2018464) has developed mini project on “ Connect 4 game in Python” .This project carried out by student is their own work as best of my knowledge.

**Ms. Preeti Chaudhary**

**Class Co-ordinator**

**CSE-A-V-SEM**

**(CSE Department)**

**GEHU Dehradun**

**ACKNOWLEDGEMENT**

I would like to particularly thank my Class Co-ordinator **Ms. Preeti chaudhary** for his patience, support and encouragement throughout the completion of this project.

       At last, but not the least I greatly indebted to all other persons who directly or indirectly helped me during this course.

**Maninder singh**

**Univ. Roll No.- 2018464**

**B.Tech CSE-A-V- Sem**

**Session: 2022-2023**

**GEHU, Dehradun**

# Overview

In this project, I will get a chance to learn some important fundamentals of programming and game design and use them to build the famous multiplayer game Connect4.

Connect 4 is a famous game that comes in various variants. The gist of the game is to get four coins in a straight line in any direction (top to bottom, left to right, or diagonals) before your opponent does the same.

# Technologies Used

1. **Pygame**

**Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language.**

1. **Numpy**

**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. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python.**

**Logical Part**

ROWS, COLUMNS = 6, 7

*def* didWin(*board*, *col*, *row*):

    player = board[row][col]

    # Horizontal Check

    count = 0

    for r in range(COLUMNS):

        if player == board[row][r]:

            count += 1

        else:

            count = 0

        if count >= 4:

            return True

    # Vertical Check

    count = 0

    for r in range(ROWS):

        if player == board[r][col]:

            count += 1

        else:

            count = 0

        if count >= 4:

            return True

    # Diagonal Check

    for r in range(3, 6):

        for c in range(3, 7):

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

                return True

    for r in range(3, 6):

        for c in range(0, 4):

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

                return True

*def* set\_piece(*board*, *col*, *row*, *num*):

    board[row][col] = num

*def* check\_col(*board*, *col*):

    if (board[0][col] == 0):

        return True

    else:

        return False

*def* get\_row(*board*, *col*):

    r = ROWS-1

    while r >= 0:

        if board[r][col] == 0:

            return r

        r -= 1

**UI Part**

import pygame

import logic

import sys

pygame.init()

COLUMNS = logic.COLUMNS

ROWS = logic.ROWS

BLUE, BLACK, RED, YELLOW = (0, 0, 255), (0, 0, 0), (255, 0, 0), (255, 255, 0)

SQUARESIZE = 100

WIDTH, HEIGHT = COLUMNS\*SQUARESIZE, (ROWS+1)\*SQUARESIZE

SIZE = (WIDTH, HEIGHT)

RADIUS = int(SQUARESIZE/2 - 5)

WINDOW = pygame.display.set\_mode(SIZE)

myfont = pygame.font.SysFont('ariel', 75)

*def* draw\_window(*board*):

    for r in range(1, ROWS+1):

        for c in range(COLUMNS):

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

            if *board*[r-1][c] == 0:

                pygame.draw.circle(

                    WINDOW, BLACK, (c\*SQUARESIZE + int(SQUARESIZE/2), r\*SQUARESIZE + int(SQUARESIZE/2)), RADIUS)

            elif *board*[r-1][c] == 1:

                pygame.draw.circle(

                    WINDOW, RED, (c\*SQUARESIZE + int(SQUARESIZE/2), r\*SQUARESIZE + int(SQUARESIZE/2)), RADIUS)

            elif *board*[r-1][c] == 2:

                pygame.draw.circle(

                    WINDOW, YELLOW, (c\*SQUARESIZE + int(SQUARESIZE/2), r\*SQUARESIZE + int(SQUARESIZE/2)), RADIUS)

    pygame.display.update()

*def* Player1Won():

    pygame.draw.rect(WINDOW, BLACK, (0, 0, WIDTH, SQUARESIZE))

    lable = myfont.render("Player 1 WINS!!", 1, RED)

    WINDOW.blit(lable, (40, 10))

    print("Player 1 Win")

*def* Player2Won():

    pygame.draw.rect(WINDOW, BLACK, (0, 0, WIDTH, SQUARESIZE))

    lable = myfont.render("Player 2 WINS!!", 1, YELLOW)

    WINDOW.blit(lable, (40, 10))

    print("Player 2 Win")

*def* Motion(*posx*, *turn*):

    pygame.draw.rect(WINDOW, BLACK, (0, 0, WIDTH, SQUARESIZE))

    if *turn* == 1:

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

    else:

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

    pygame.display.update()

**Driver Code**

import pygame

import numpy as np

import sys

from logic import \*

import logic

from UI import \*

COLUMNS = logic.COLUMNS

ROWS = logic.ROWS

board = np.zeros((ROWS, COLUMNS))

game\_over = False

turn = 1

pygame.init()

clock = pygame.time.Clock()

draw\_window(board)

while not game\_over:

    clock.tick(60)

    for event in pygame.event.get():

        if event.type == pygame.QUIT:

            sys.exit()

        if event.type == pygame.MOUSEMOTION:

            Motion(event.pos[0], turn)

        if event.type == pygame.MOUSEBUTTONDOWN:

            if turn == 1:

                col = int(event.pos[0]/SQUARESIZE)

                if (check\_col(board, col) == False):

                    print("Column filled try again!")

                    continue

                row = get\_row(board, col)

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

                if didWin(board, col, row):

                    Player1Won()

                    game\_over = True

                turn += 1

            else:

                col = int(event.pos[0]/SQUARESIZE)

                if (check\_col(board, col) == False):

                    print("Column filled try again!")

                    continue

                row = get\_row(board, col)

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

                if didWin(board, col, row):

                    Player2Won()

                    game\_over = True

                turn -= 1

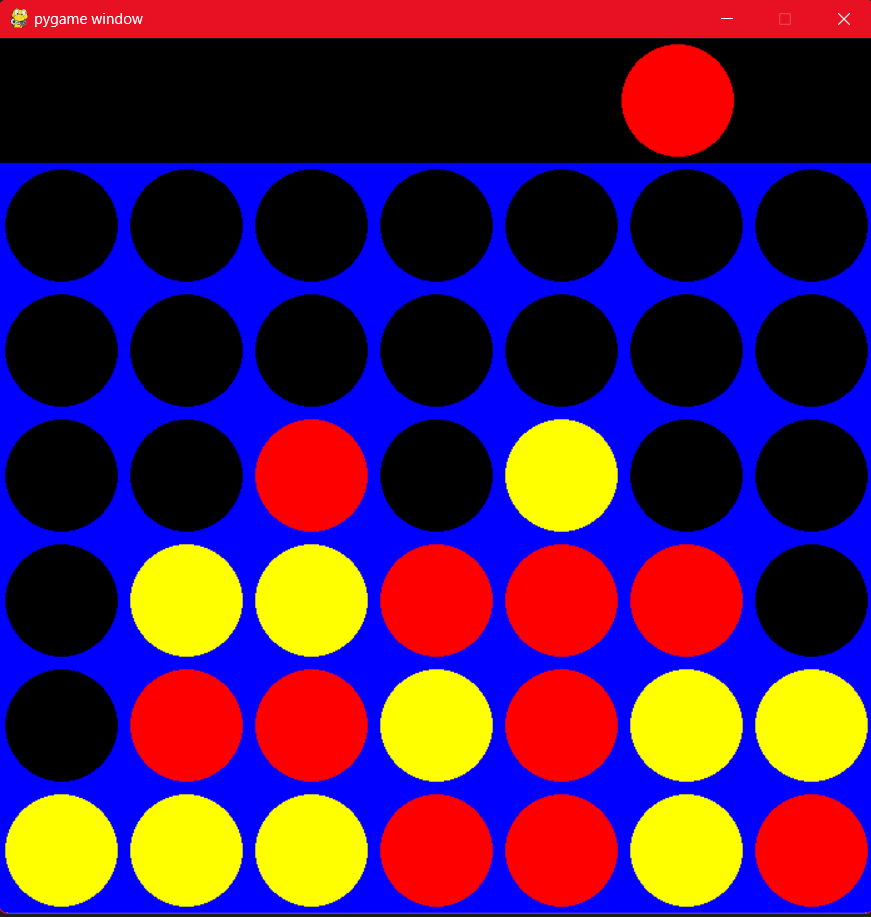
            draw\_window(board)

            print(board)

        if game\_over == True:

            pygame.time.wait(3000)

**Screen Shot**

****

**Future Goals**

The future goals of the project are mentioned below:

1. **Genre** –Will also integrate a functionality to play with computer bot.
2. **UI** -Will improvise on the design of the game.
3. **Chat Room –** will also have the chatroom within the playing window.

# Reference Resources

* 1. **FreeCodeCamp.**
  2. **Crio.Do**