National University of Computer & Emerging Sciences
Karachi Campus

CONNECT-FOUR USING MINIMAX ALGORITHM WITH ALPHA-BETA PRUNING

PROJECT REPORT:

Artificial Intelligence [AI] Lab
Section: BCY-6B

Instructor: Miss Mehak Mazhar

Group Members:

22K-4796 Fakhur ul Din
22K-4718 Muhammad Hasnain
22K-4761 Ayesha Siddiqui

1. Introduction:

Connect Four is a two-player connection game where opponents take turns dropping colored discs into a vertically suspended grid. The objective is to form a horizontal, vertical, or diagonal line of four discs before the opponent does. This project implements a Python-based Connect Four game with an AI opponent using Minimax with Alpha-Beta Pruning. The game features: - Interactive GUI (Pygame) - Adjustable difficulty levels (Easy, Medium, Hard) - Multiple board sizes (6×7, 7×8, 8×9) - Animated disc drops - Sound effects - Win/draw detection

2. Project Objectives:

- Implement an **AI opponent** that can play Connect Four strategically.
- Use **Minimax algorithm with Alpha-Beta pruning** to improve AI efficiency.
- Allow **difficulty level customization** (Easy, Medium, Hard).
- Enable **board size selection** for varied gameplay experiences.
- Design a **responsive and engaging user interface** using Pygame

3. Game Description:

Original Game:

Connect Four is played on a 6x7 grid. Players alternate turns dropping colored discs into the grid. A player wins by connecting four discs consecutively. If all cells are filled without a winner, the game ends in a draw.

Features Added:

- **AI Opponent:** The game allows solo play against an intelligent bot.
- **Dynamic Board Sizes:** Users can choose from 6×7, 7×8, and 8×9 configurations.

- **Adjustable AI Depths:** Difficulty levels correspond to AI tree depth (2–6).
- **Intuitive Interface:** Includes gradient backgrounds, animations, and sound.
- **Optimized Evaluation Function:** Assigns scores to board states based on potential lines, blocking moves, and winning paths.

4. AI Methodology:

1) Minimax Algorithm

The Minimax algorithm is used to simulate all possible moves by both players and evaluate the outcomes. The AI assumes the human plays optimally and selects the best counterstrategy accordingly.

2) Alpha-Beta Pruning

To enhance performance, Alpha-Beta pruning reduces the number of nodes evaluated by cutting off branches that cannot influence the final decision. This significantly reduces the computational load.

3) Evaluation Heuristic

- Prioritizes moves that advance AI sequences (−1).
- Penalizes moves that allow opponent connections (+1).
- Evaluates rows, columns, and diagonals for possible 4-in-a-row opportunities.
- Balances offensive and defensive strategy based on the depth level.

5.Game Mechanics:

Rules:

- A player wins by connecting four of their discs consecutively.
- If the board is full with no winner, the game is a draw.
- In single-player mode, the human player always goes first.
- In two-player mode (extension idea), players alternate turns manually.

Gameplay Flow:

1. **Menu Screen:** Select board size and difficulty.
2. **Game Loop:** Players alternate turns; AI computes move using alpha-beta pruning.
3. **Win Check:** After each move, check for victory or draw.
4. **End Screen:** Show results with "Play Again" and "Quit" options.

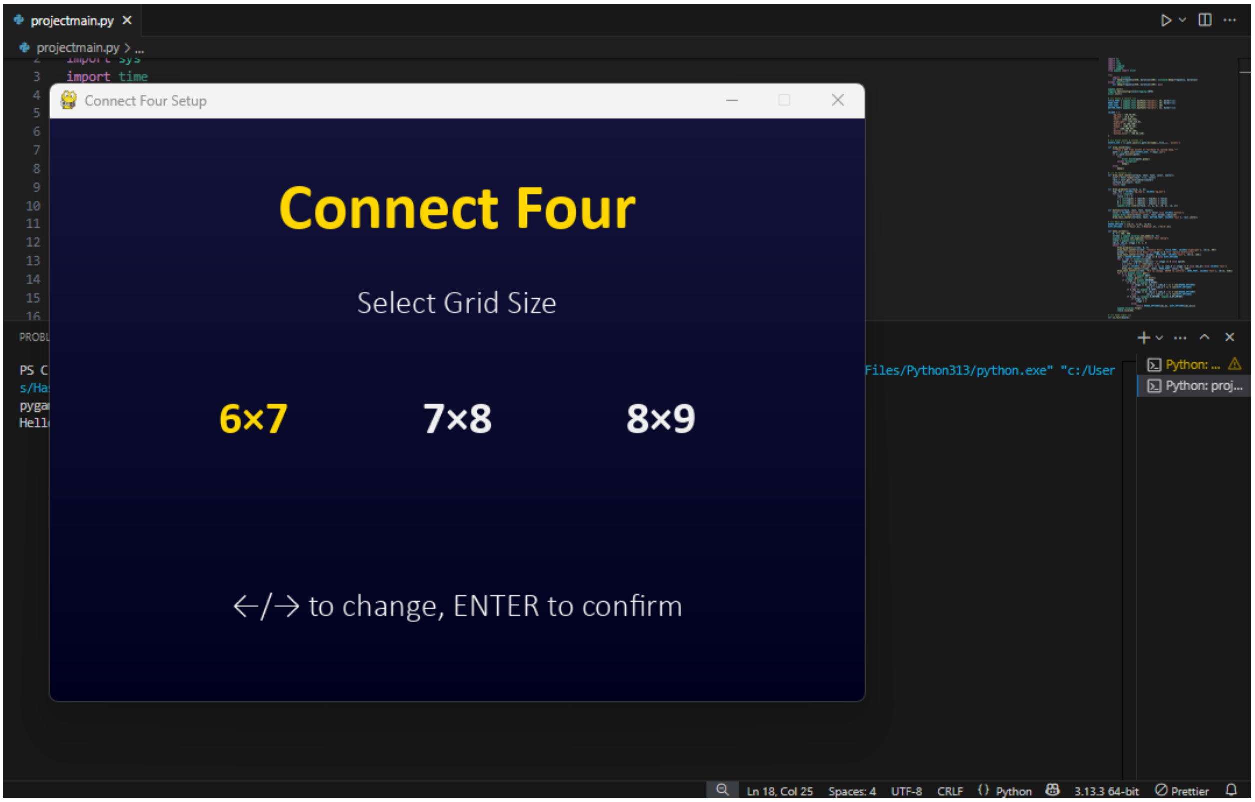
6.Technologies Used:

Component	Tool / Library
Programming Language	Python
GUI & Graphics	Pygame
Sound Effects	Pygame.mixer / winsound
AI Algorithm	Minimax + Alpha-Beta Pruning

7.Implementation Timeline:

Week	Task
1–2	Game design and UI planning
3–4	Minimax and Alpha-Beta pruning implementation
5–6	Core gameplay logic, win-checking, animations
7	AI integration and menu customization
8	Final testing, debugging, and report writing

8.Working Screenshots:



```
import sys
import time
```

```
PS C
s/Ha                                    Files/Python313/python.exe" "c:/User
pyga
Hell
```

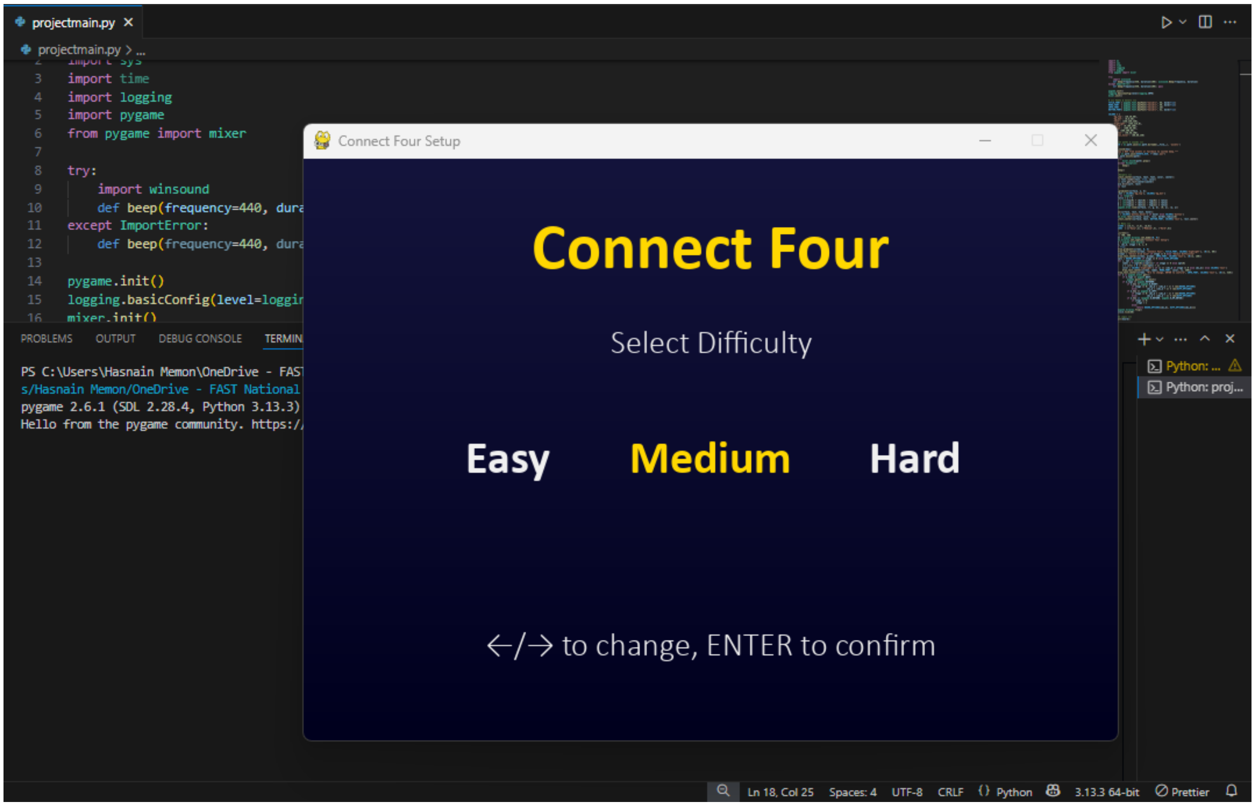
Connect Four Setup

Connect Four

Select Grid Size

6×7 7×8 8×9

←/→ to change, ENTER to confirm



```
import sys
import time
import logging
import pygame
from pygame import mixer

try:
    import winsound
    def beep(frequency=440, dura
except ImportError:
    def beep(frequency=440, dura

pygame.init()
logging.basicConfig(level=loggir
mixer.init()
```

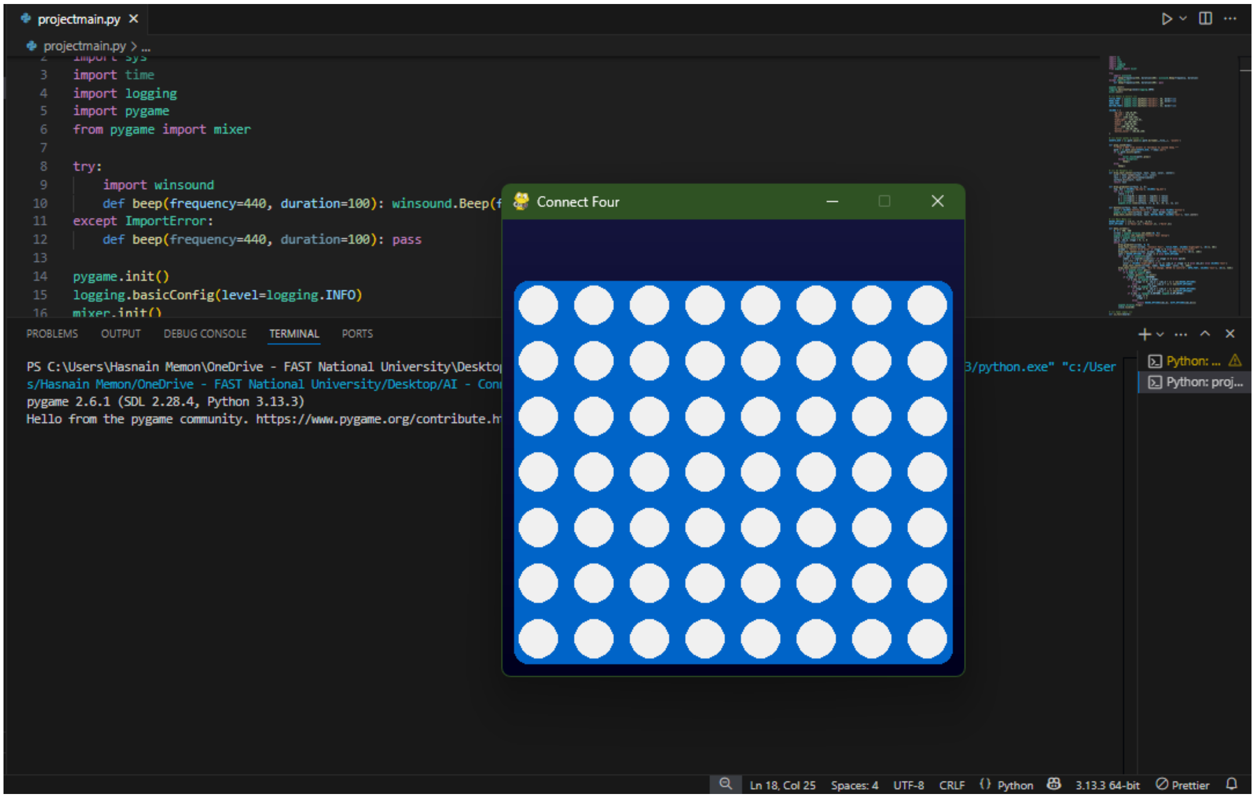
Connect Four

Select Difficulty

Easy Medium Hard

←/→ to change, ENTER to confirm

```
PS C:\Users\Hasnain Memon\OneDrive - FAST
s/Hasnain Memon/OneDrive - FAST National
pygame 2.6.1 (SDL 2.28.4, Python 3.13.3)
Hello from the pygame community. https:/
```

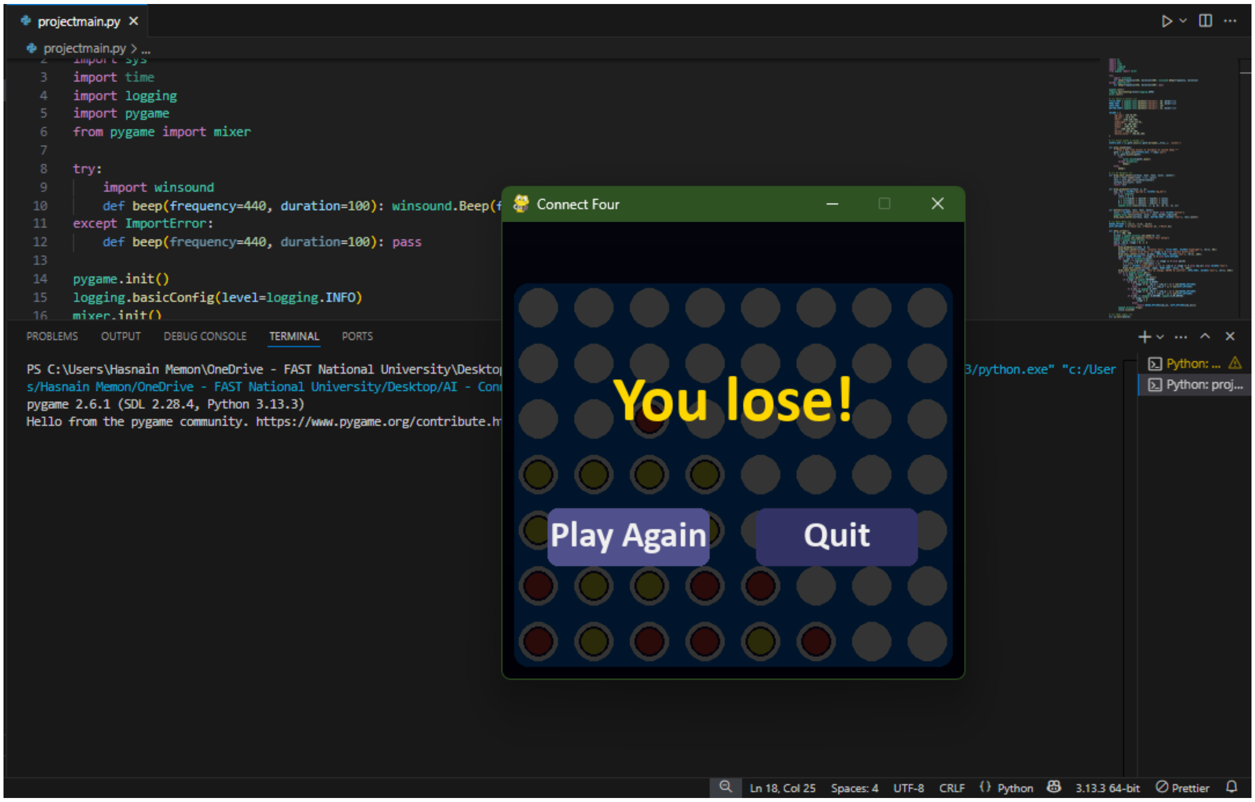


```
import sys
import time
import logging
import pygame
from pygame import mixer

try:
    import winsound
    def beep(frequency=440, duration=100): winsound.Beep(f
except ImportError:
    def beep(frequency=440, duration=100): pass

pygame.init()
logging.basicConfig(level=logging.INFO)
mixer.init()
```

```
PS C:\Users\Hasnain Memon\OneDrive - FAST National University\Desktop                3/python.exe" "c:/User
s/Hasnain Memon/OneDrive - FAST National University/Desktop/AI - Con
pygame 2.6.1 (SDL 2.28.4, Python 3.13.3)
Hello from the pygame community. https://www.pygame.org/contribute.h
```



```python
import sys
import time
import logging
import pygame
from pygame import mixer

try:
    import winsound
    def beep(frequency=440, duration=100): winsound.Beep(f
except ImportError:
    def beep(frequency=440, duration=100): pass

pygame.init()
logging.basicConfig(level=logging.INFO)
mixer.init()
```

```
PS C:\Users\Hasnain Memon\OneDrive - FAST National University\Deskto     3/python.exe" "c:/User
s/Hasnain Memon/OneDrive - FAST National University/Desktop/AI - Con
pygame 2.6.1 (SDL 2.28.4, Python 3.13.3)
Hello from the pygame community. https://www.pygame.org/contribute.h
```

You lose!

9.Conclusion:

This project successfully demonstrates the integration of AI search techniques into a classic board game, enhancing it with usability features like animations, sound, and customization. Through the Minimax algorithm with Alpha-Beta pruning, the AI opponent makes calculated moves efficiently, offering a challenging experience to players of all skill levels.

The code is modular, expandable (e.g., to two-player or online modes), and optimized for performance. This project show cases how traditional games can be revitalized using fundamental AI principles.