jupyter.org/try-jupyter/notebooks/index.html?path=notebooks%2F+minimax.ipynb

jupyterlite minimax Last Checkpoint: 1 minute ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python (Pyodide)

[3]:
```
import math

# Print the Tic-Tac-Toe board
def print_board(board):
    for i in range(0, 9, 3):
        print(" | ".join(board[i:i+3]))
    print("\n")

# Check for winner
def check_winner(board):
    winning_combinations = [
        [board[0], board[1], board[2]],
        [board[3], board[4], board[5]],
        [board[6], board[7], board[8]],
        [board[0], board[3], board[6]],
        [board[1], board[4], board[7]],
        [board[2], board[5], board[8]],
        [board[0], board[4], board[8]],
        [board[2], board[4], board[6]]
    ]
    if ["X", "X", "X"] in winning_combinations:
        return "X"
    elif ["O", "O", "O"] in winning_combinations:
        return "O"
    return None
```

jupyter.org/try-jupyter/notebooks/index.html?path=notebooks%2F+minimax.ipynb

jupyterlite minimax Last Checkpoint: 2 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python (Pyodide)

```python
# Minimax algorithm
def minimax(board, depth, is_maximizing):
    winner = check_winner(board)
    if winner == "X":
        return -10 + depth
    elif winner == "O":
        return 10 - depth
    elif "." not in board:
        return 0  # Tie

    if is_maximizing:
        best_score = -math.inf
        for i in range(len(board)):
            if board[i] == ".":
                board[i] = "O"
                score = minimax(board, depth + 1, False)
                board[i] = "."  # Undo move
                best_score = max(score, best_score)
        return best_score
    else:
        best_score = math.inf
        for i in range(len(board)):
            if board[i] == ".":
                board[i] = "X"
                score = minimax(board, depth + 1, True)
                board[i] = "."  # Undo move
```

JupyterLite | minimax | depth first search | Microsoft Copilot: Your AI com

jupyter.org/try-jupyter/notebooks/index.html?path=notebooks%2F+minimax.ipynb

Click to go back, hold to see history

jupyterlite minimax Last Checkpoint: 2 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab

Python (Pyodide)

```
                score = minimax(board, depth + 1, True)
                board[i] = "."  # Undo move
                best_score = min(score, best_score)
        return best_score

# Find the best move for AI ('O')
def best_move(board):
    best_score = -math.inf
    move = -1
    for i in range(len(board)):
        if board[i] == ".":
            board[i] = "O"
            score = minimax(board, 0, False)
            board[i] = "."  # Undo move
            if score > best_score:
                best_score = score
                move = i
    return move

# Example board setup (with some pre-filled moves)
board = ["X", "O", "X",
         "O", "X", ".",
         ".", "O", "X"]

print("Current Board:")
print_board(board)
```

Search

ENG IN

18:39 10-06-2025

JupyterLite | minimax | depth first search | Microsoft Copilot: Your AI com

jupyter.org/try-jupyter/notebooks/index.html?path=notebooks%2F+minimax.ipynb

jupyterlite minimax Last Checkpoint: 2 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python (Pyodide)

```
print("Current Board:")
print_board(board)

# Get AI's best move
move_index = best_move(board)
print(f"Best move for AI (O): Position {move_index}")

# Apply the best move and display updated board
if move_index != -1:
    board[move_index] = "O"

print("Board After Best Move:")
print_board(board)
```

```
Current Board:
X | O | X
O | X | .
. | O | X


Best move for AI (O): Position 5
Board After Best Move:
X | O | X
O | X | O
. | O | X
```

Search

ENG IN

18:39 10-06-2025