AI LAB 1:

Program Title: Tic Tac Toe game

Code :

```
import random


def check_win(board, player):
    # Check rows, columns, and diagonals for a win
    for row in board:
        if all(spot == player for spot in row):
            return True
    for col in range(3):
        if all(board[row][col] == player for row in range(3)):
            return True
    if all(board[i][i] == player for i in range(3)) or all(board[i][2 - i] == player for i in range(3)):
        return True
    return False


def display_board(board):
    for row in board:
        print(row)
    print()


def get_available_moves(board):
    return [(r, c) for r in range(3) for c in range(3) if board[r][c] == '-']


def bot_move(board):
    # Check if the bot can win in the next move
    for move in get_available_moves(board):
        r, c = move
        board[r][c] = 'O'
        if check_win(board, 'O'):
            print(f"Bot placed O at position: ({r + 1}, {c + 1})")
            display_board(board)
            return
        board[r][c] = '-'

    # Check if the player is about to win, and block them
    for move in get_available_moves(board):
        r, c = move
        board[r][c] = 'X'
        if check_win(board, 'X'):
            board[r][c] = 'O'
            print(f"Bot placed O at position: ({r + 1}, {c + 1}) to block the player")
            display_board(board)
            return
        board[r][c] = '-'

    # Otherwise, pick a random available move
    move = random.choice(get_available_moves(board))
    board[move[0]][move[1]] = 'O'
    print(f"Bot placed O at position: ({move[0] + 1}, {move[1] + 1})")
    display_board(board)

# Initial board setup
board = [['-', '-', '-'], ['-', '-', '-'], ['-', '-', '-']]
display_board(board)

xo = 1  # 1 for human, 0 for bot
flag = 0  # Flag to check for win or draw
```

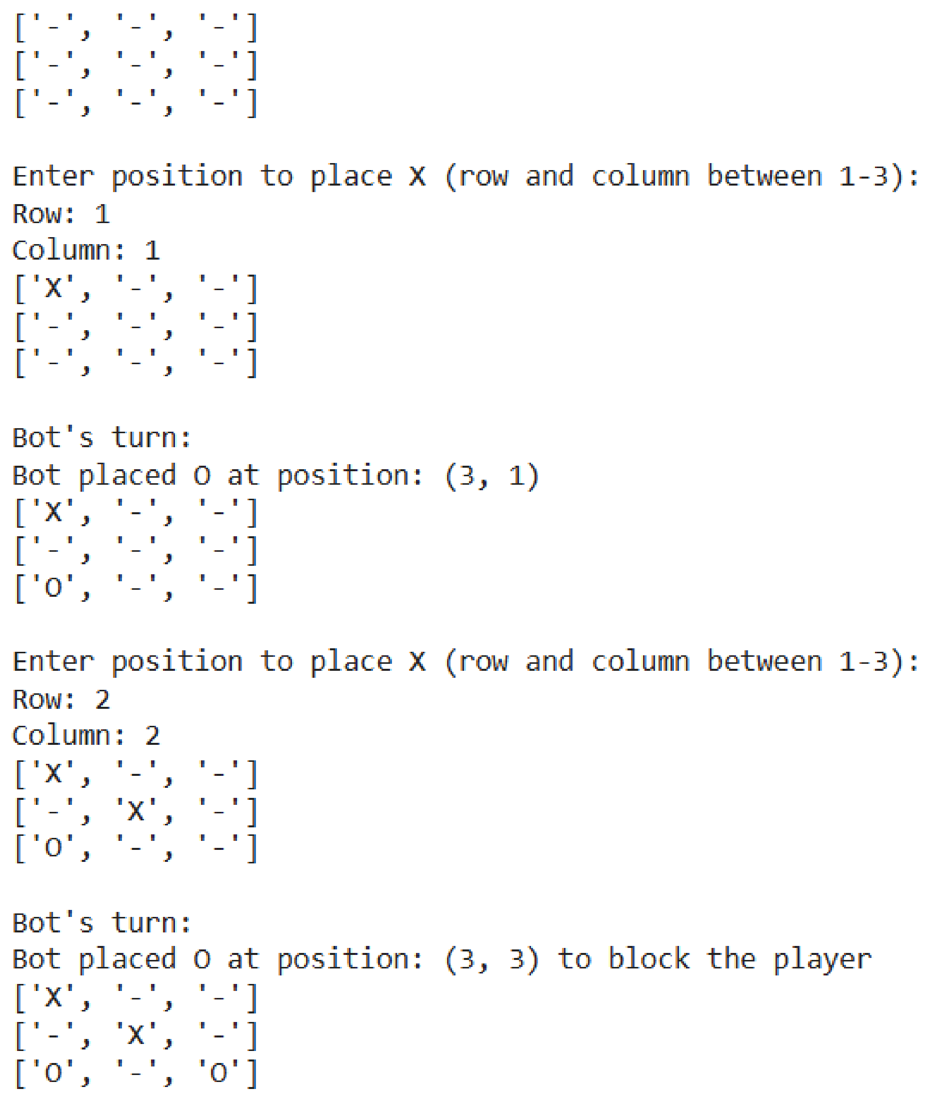
```
while '-' in board[0] or '-' in board[1] or '-' in board[2]:
    if xo == 1:  # Human's turn (X)
        print("Enter position to place X (row and column between 1-3):")
        try:
            x = int(input("Row: "))
            y = int(input("Column: "))
        except ValueError:
            print("Invalid input. Please enter numbers between 1 and 3.")
            continue

        if x > 3 or y > 3 or x < 1 or y < 1:
            print("Invalid position")
            continue

        if board[x - 1][y - 1] == '-':
            board[x - 1][y - 1] = 'X'
            display_board(board)
            if check_win(board, 'X'):
                print("X wins!")
                flag = 1
                break
            xo = 0  # Switch to bot's turn
        else:
            print("Invalid position")
            continue
    else:  # Bot's turn (O)
        print("Bot's turn:")
        bot_move(board)
        if check_win(board, 'O'):
            print("O (Bot) wins!")
            flag = 1
            break
        xo = 1  # Switch back to human's turn


if flag == 0:
    print("Draw")
print("Game Over")
```

Output:



```
['-', '-', '-']
['-', '-', '-']
['-', '-', '-']

Enter position to place X (row and column between 1-3):
Row: 1
Column: 1
['X', '-', '-']
['-', '-', '-']
['-', '-', '-']

Bot's turn:
Bot placed O at position: (3, 1)
['X', '-', '-']
['-', '-', '-']
['O', '-', '-']

Enter position to place X (row and column between 1-3):
Row: 2
Column: 2
['X', '-', '-']
['-', 'X', '-']
['O', '-', '-']

Bot's turn:
Bot placed O at position: (3, 3) to block the player
['X', '-', '-']
['-', 'X', '-']
['O', '-', 'O']
```

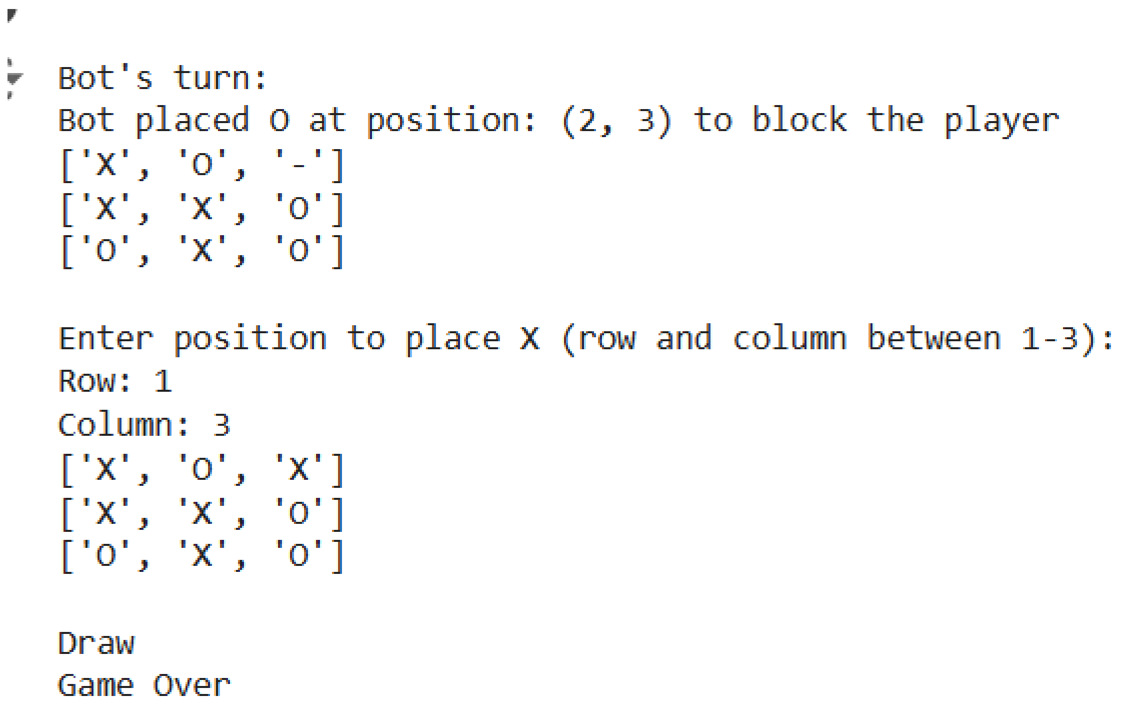
```
Enter position to place X (row and column between 1-3):
Row: 3
Column: 2
['X', '-', '-']
['-', 'X', '-']
['O', 'X', 'O']

Bot's turn:
Bot placed O at position: (1, 2) to block the player
['X', 'O', '-']
['-', 'X', '-']
['O', 'X', 'O']

Enter position to place X (row and column between 1-3):
Row: 2
Column: 1
['X', 'O', '-']
['X', 'X', '-']
['O', 'X', 'O']

Bot's turn:
Bot placed O at position: (2, 3) to block the player
['X', 'O', '-']
['X', 'X', 'O']
['O', 'X', 'O']

Enter position to place X (row and column between 1-3):
Row: 1
Column: 3
['X', 'O', 'X']
['X', 'X', 'O']
['O', 'X', 'O']
```

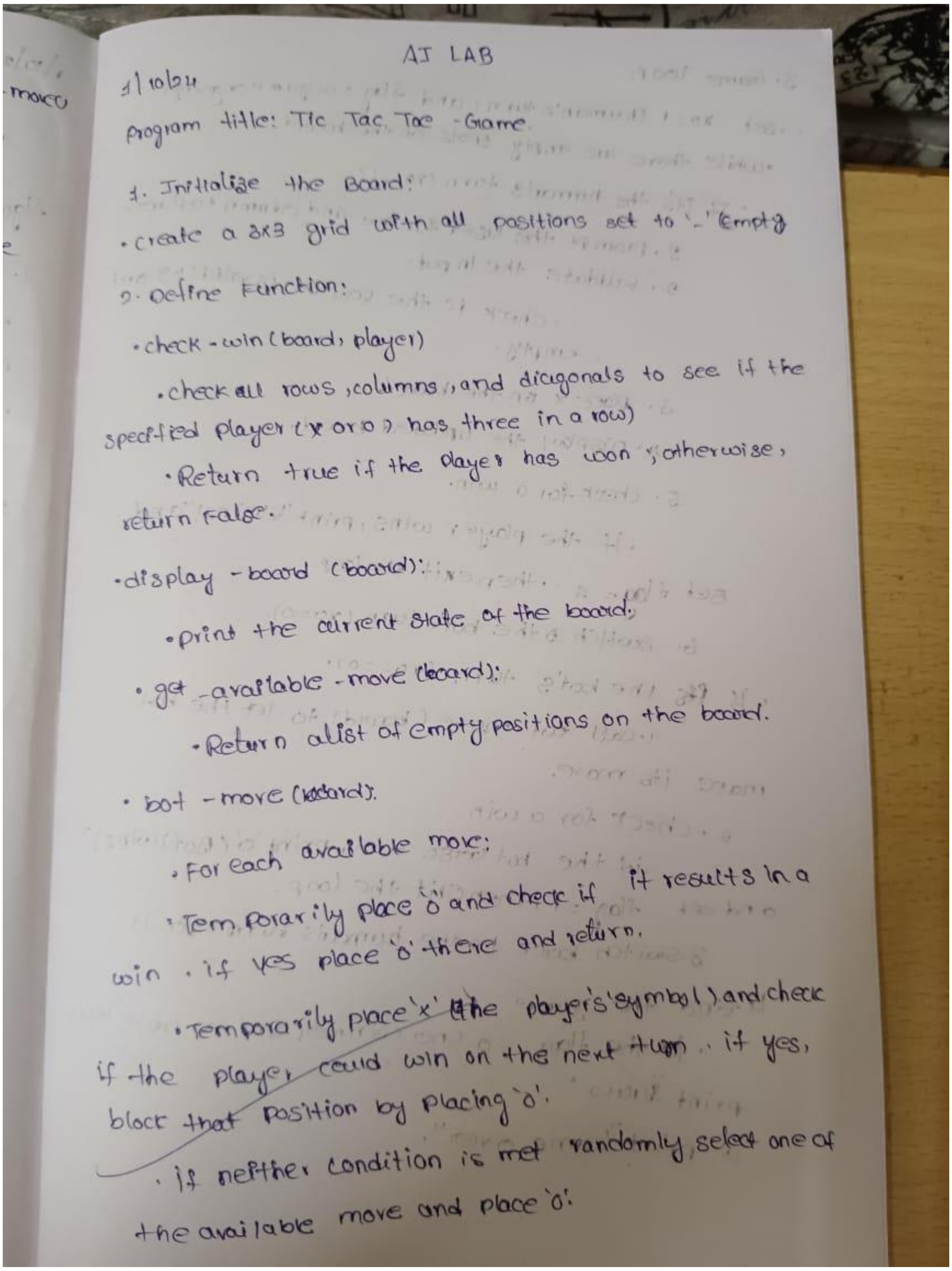


```
Bot's turn:
Bot placed O at position: (2, 3) to block the player
['X', 'O', '-']
['X', 'X', 'O']
['O', 'X', 'O']

Enter position to place X (row and column between 1-3):
Row: 1
Column: 3
['X', 'O', 'X']
['X', 'X', 'O']
['O', 'X', 'O']

Draw
Game Over
```

Algorithm:



AI LAB

1|10|24

Program title: Tic Tac Toe - Game

1. Initialize the Board:

- create a 3x3 grid with all positions set to '-' Empty

2. Define Function:

- check - win (board, player)
    - check all rows, columns, and diagonals to see if the specified player (x or o) has three in a row)
    - Return true if the player has won; otherwise, return False.
- display - board (board):
    - print the current state of the board.
- get - available - move (board):
    - Return a list of empty positions on the board.
- bot - move (board):
    - For each available move:
    - Temporarily place 'o' and check if it results in a win. if yes place 'o' there and return.
    - Temporarily place 'x' (the player's symbol) and check if the player could win on the next turn. if yes, block that position by placing 'o'.
    - if neither condition is met randomly select one of the available move and place 'o'.

```
3. Game loop:
 . set xo=1 (human's turn), and flag =0(game ongoing).
 .while there are empty spots on the board.
   . If it's the human's turn (xo==1)
       1. prompt the user for row and column to place x
       2. validate the input:
              . check if the position is valid (1-3 and
            empty.
       3. place x on the board.
       4. Display the updated board.
       5. check for a win.
              . if the player wins, print "x wins!" and,
     set flag = ~~1~~ , then exit the loop.
       6. switch to the bot's turn (xo=0).
   . If it's the bot's turn (xo==0).
          1. call bot-move (board) to let the bot
   make its move.
     2. check for a win.
          . if the bot ~~wise~~ win, print "O (Bot) wins)"
   and set flag =1, then exit the loop.
     3. switch back to the human's turn (xo=1)
4. End the game
   . if the flag ==0 (no winner and no more left),
   print "Draw".
   . print "Game over".
```

```
output:

['_', '_', '_']
['_', '_', '_']
['_', '_', '_']

Enter position to
  Row: 2
  Column: 2

[ _, _, _,
  _, x, _
  _, _, _ ]

Bot's turn.

    (3,3)

[ _ _ _ ]
[ _ x _ ]
[ _ _ O ]

Humanturn (

[ _   _ ]
[ _ x _ ]
[ x _ O ]

Bot's turn. (

[ _ _ O ]
[ _ x _ ]
[ x _ O ]

Humantur

[ _ _ O ]
[ _ x x
[ x _ O
```

output:

```
['_', '_', '_']
['_', '_', '_']
['_', '_', '_']
```

Enter position to place x (row and column between 1-3):

```
Row: 2
Column: 2

[ -, -, -,
  -, x, -
  -, -, - ]
```

Bot's turn.

```
(3,3)

[ - - - ]
[ - x - ]
[ - - 0 ]
```

Human turn (3,1)

```
[ -   - ]
[ - x - ]
[ x - 0 ]
```

Bot's turn. (1,3)

```
[ - - 0 ]
[ - x - ]
[ x - 0 ]
```

Human turn.

```
[ - - 0 ]
[ - x x ]
[ x - 0 ]
```

Bot turn (3, 2)

```
[ x     0 ]
[ 0  x  x ]
[ x  0  0 ]
```

Human turn
(1, 2)

```
[ x  x  0
  0  x  x
  x  0  0 ]
```

Draw
Game Draw.

[illegible]
1/10/24