

Laporan Praktikum Kecerdasan Buatan

Semester Ganjil 2024/2025

Jurusan Teknik Elektro
Program Studi S1 Teknik Elektro
Fakultas Teknik
Universitas Tidar

Praktikum ke- : 5
Judul Praktikum : Teori Game Dalam Kecerdasan Buatan

Nama : Jerico Christianto
NIM : 2220501082

“Laporan praktikum ini saya kerjakan dan selesaikan dengan sebaik-baiknya tanpa melakukan tindak kecurangan. Apabila di kemudian hari ditemukan adanya kecurangan pada laporan ini, maka saya bersedia menerima konsekuensinya.”

Tertanda,



Jerico Christianto

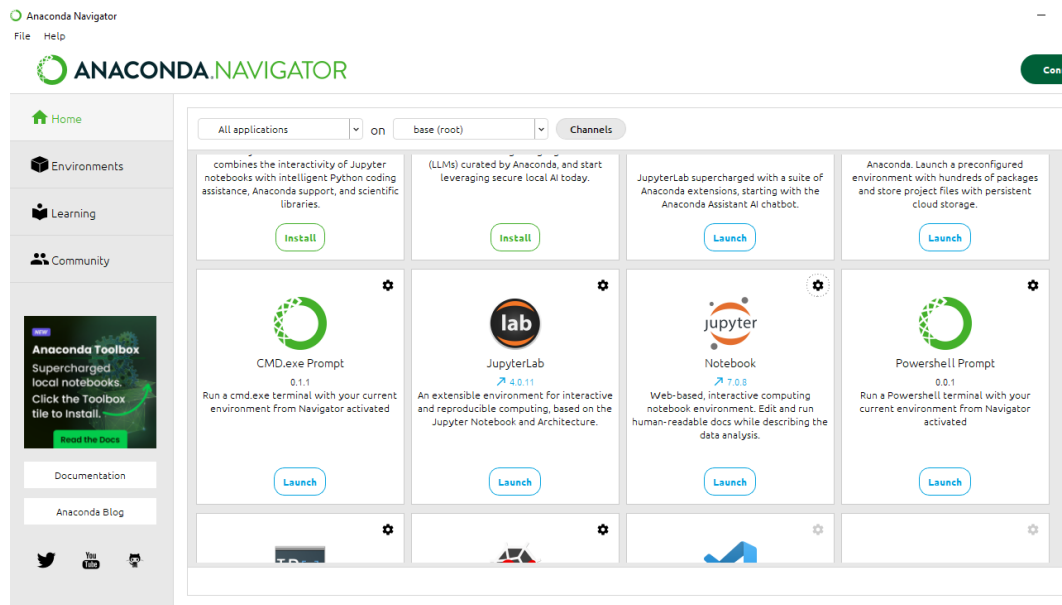
2220501082

A. Tujuan Praktikum

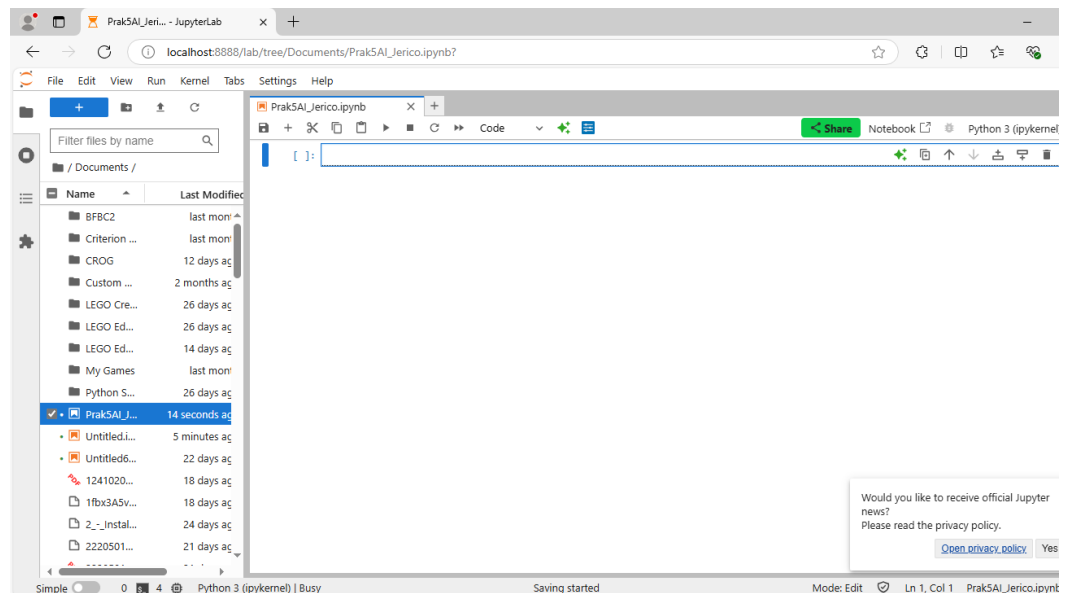
1. Memahami konsep dasar teori game dalam kecerdasan buatan
2. Menerapkan algoritma tentang teori game dalam kecerdasan buatan

B. Langkah Praktikum

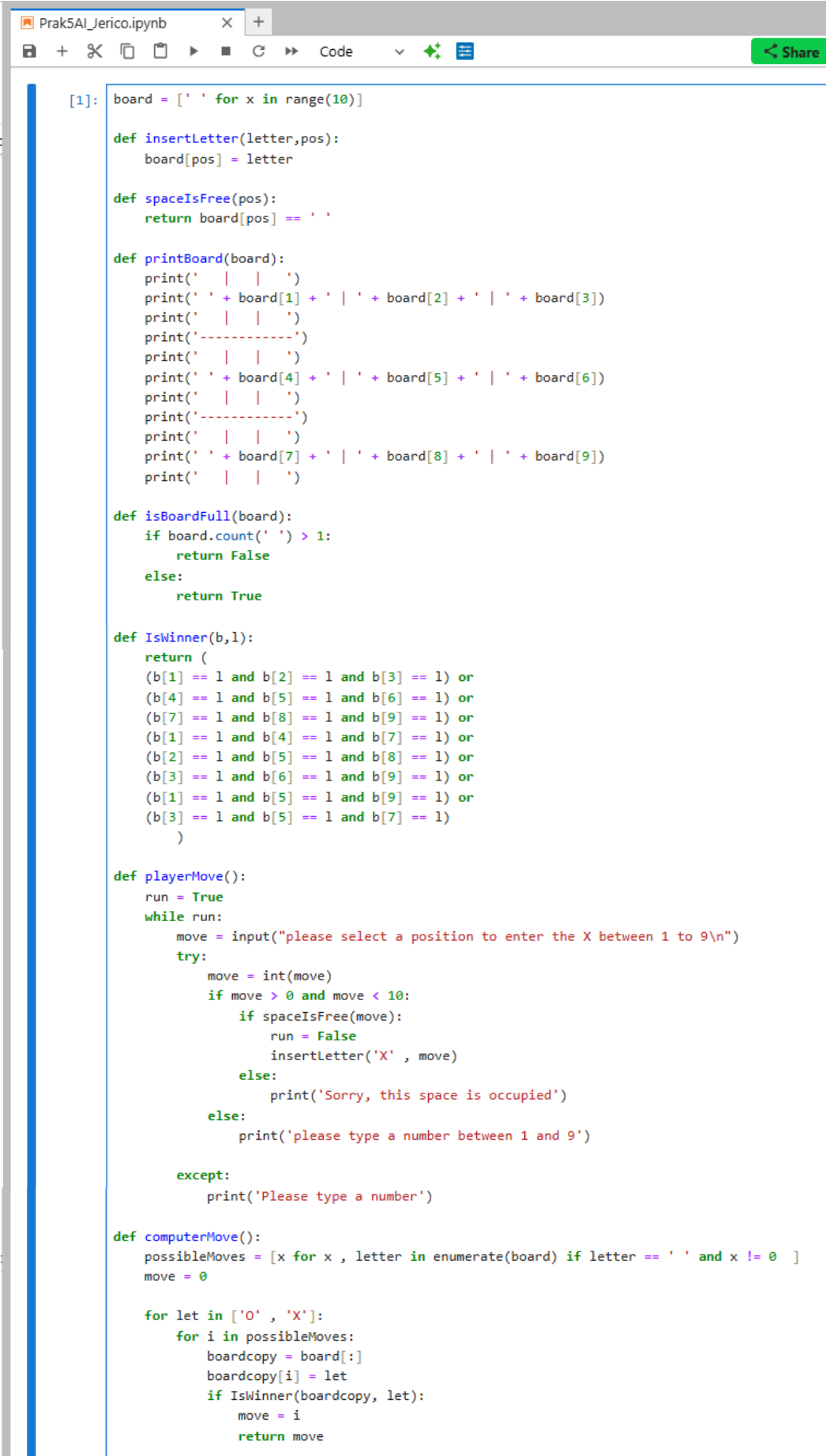
1. Membuka Anaconda Navigator



2. Menjalankan JupyterLab dan membuat file baru dengan nama “Praktik5AI_Jerico”



3. Menuliskan kode program seperti berikut



```
[1]: board = [' ' for x in range(10)]

def insertLetter(letter,pos):
    board[pos] = letter

def spaceIsFree(pos):
    return board[pos] == ' '

def printBoard(board):
    print(' | | ')
    print(' ' + board[1] + ' | ' + board[2] + ' | ' + board[3])
    print(' | | ')
    print('-----')
    print(' | | ')
    print(' ' + board[4] + ' | ' + board[5] + ' | ' + board[6])
    print(' | | ')
    print('-----')
    print(' | | ')
    print(' ' + board[7] + ' | ' + board[8] + ' | ' + board[9])
    print(' | | ')

def isBoardFull(board):
    if board.count(' ') > 1:
        return False
    else:
        return True

def IsWinner(b,l):
    return (
        (b[1] == l and b[2] == l and b[3] == l) or
        (b[4] == l and b[5] == l and b[6] == l) or
        (b[7] == l and b[8] == l and b[9] == l) or
        (b[1] == l and b[4] == l and b[7] == l) or
        (b[2] == l and b[5] == l and b[8] == l) or
        (b[3] == l and b[6] == l and b[9] == l) or
        (b[1] == l and b[5] == l and b[9] == l) or
        (b[3] == l and b[5] == l and b[7] == l)
    )

def playerMove():
    run = True
    while run:
        move = input("please select a position to enter the X between 1 to 9\n")
        try:
            move = int(move)
            if move > 0 and move < 10:
                if spaceIsFree(move):
                    run = False
                    insertLetter('X', move)
                else:
                    print('Sorry, this space is occupied')
            else:
                print('please type a number between 1 and 9')
        except:
            print('Please type a number')

def computerMove():
    possibleMoves = [x for x, letter in enumerate(board) if letter == ' ' and x != 0 ]
    move = 0

    for let in ['O', 'X']:
        for i in possibleMoves:
            boardcopy = board[:]
            boardcopy[i] = let
            if IsWinner(boardcopy, let):
                move = i
                return move
```

```

cornersOpen = []
for i in possibleMoves:
    if i in [1, 3, 7, 9]:
        cornersOpen.append(i)

if len(cornersOpen) > 0:
    move = selectRandom(cornersOpen)
    return move

if 5 in possibleMoves:
    move = 5
    return move

edgesOpen = []
for i in possibleMoves:
    if i in [2, 4, 6, 8]:
        edgesOpen.append(i)

if len(edgesOpen) > 0:
    move = selectRandom(edgesOpen)
    return move

def selectRandom(li):
    import random
    ln = len(li)
    r = random.randrange(0, ln)
    return li[r]

def main():
    print("Welcome to the game!")
    printBoard(board)

    while not(isBoardFull(board)):
        if not(IsWinner(board, 'O')):
            playerMove()
            printBoard(board)
        else:
            print("sorry you loose!")
            break

        if not(IsWinner(board, 'X')):
            move = computerMove()
            if move == 0:
                print(" ")
            else:
                insertLetter('O', move)
                print('computer placed an o on position', move, ':')
                printBoard(board)
        else:
            print("you win!")
            break

    if isBoardFull(board):
        print("Tie game")

while True:
    x = input("Do you want to play? Press y for yes or n for no (y/n)\n")
    if x.lower() == 'y':
        board = [' ' for x in range(10)]
        print('-----')
        main()
    else:
        break

```

C. Hasil Praktikum

Praktikum kecerdasan buatan kali ini yaitu menerapkan konsep kecerdasan buatan ke dalam sebuah game, yaitu board game Tic Tac Toe

X	O	O
O	X	X
O	O	X

Permainan Tic Tac Toe pada gambar diatas menggunakan matriks 3x3. Ada dua pemain, satu menggunakan X dan satu lagi menggunakan O. Untuk menang, seorang pemain harus membuat garis lurus (horizontal, vertikal, atau diagonal) penuh dengan tanda mereka. Gambar diatas pemain X menang karena berhasil membuat garis diagonal penuh dengan X.

Kondisi yang ada pada permainan ini yaitu :

- Kondisi Awal : Permainan belum dimulai. Semua kotak pada papan masih kosong.
- Kondisi Dimulai : Salah satu pemain menempatkan tanda pertamanya pada salah satu kotak kosong.
- Kondisi Seri : Semua kotak telah terisi, tetapi tidak ada pemain yang berhasil membuat garis lurus penuh dengan tanda mereka.
- Kondisi Menang Pemain X : Ada satu baris, kolom, atau diagonal yang seluruhnya terisi dengan tanda X.
- Kondisi Menang Pemain O : Ada satu baris, kolom, atau diagonal yang seluruhnya terisi dengan tanda O.

Adapun proses berjalannya program yaitu :

1. Membuat Matriks untuk Tic Tac Toe

```
[1]: board = [' ' for x in range(10)]

def insertLetter(letter,pos):
    board[pos] = letter

def spaceIsFree(pos):
    return board[pos] == ' '

def printBoard(board):
    print(' | | ')
    print(' ' + board[1] + ' | ' + board[2] + ' | ' + board[3])
    print(' | | ')
    print('-----')
    print(' | | ')
    print(' ' + board[4] + ' | ' + board[5] + ' | ' + board[6])
    print(' | | ')
    print('-----')
    print(' | | ')
    print(' ' + board[7] + ' | ' + board[8] + ' | ' + board[9])
    print(' | | ')
```

Matriks yang dibuat yaitu berupa matriks 3x3.

2. Memeriksa apakah board sudah penuh atau belum

```
def isBoardFull(board):
    if board.count(' ') > 1:
        return False
    else:
        return True
```

Fungsi ini memeriksa apakah papan permainan sudah penuh atau belum. Jika masih ada kotak kosong, kode akan mengatakan "benar" (True) dan permainan bisa dimulai. Tapi kalau semua kotak sudah terisi, kode akan bilang "salah" (False). Jadi, fungsi ini dipakai untuk memastikan bahwa permainan belum dimulai sebelum kita mulai bermain

3. Fungsi untuk menentukan pemenang

```
def IsWinner(b,l):
    return (
        (b[1] == 1 and b[2] == 1 and b[3] == 1) or
        (b[4] == 1 and b[5] == 1 and b[6] == 1) or
        (b[7] == 1 and b[8] == 1 and b[9] == 1) or
        (b[1] == 1 and b[4] == 1 and b[7] == 1) or
        (b[2] == 1 and b[5] == 1 and b[8] == 1) or
        (b[3] == 1 and b[6] == 1 and b[9] == 1) or
        (b[1] == 1 and b[5] == 1 and b[9] == 1) or
        (b[3] == 1 and b[5] == 1 and b[7] == 1)
    )
```

Fungsi diatas merupakan fungsi untuk menentukan pemenang. Pemenang dalam game ini yaitu yang dapat mengisi salah satu baris, kolom, atau diagonal. Angka-angka diatas yaitu semua kemungkinan yang memungkinkan pemain untuk memenangkan game.

4. Fungsi untuk mengatur pergerakan pemain

```
def playerMove():
    run = True
    while run:
        move = input("please select a position to enter the X between 1 to 9\n")
        try:
            move = int(move)
            if move > 0 and move < 10:
                if spaceIsFree(move):
                    run = False
                    insertLetter('X', move)
                else:
                    print('Sorry, this space is occupied')
            else:
                print('please type a number between 1 and 9')
        except:
            print('Please type a number')
```

Fungsi diatas yaitu fungsi untuk mengatur pergerakan pemain. Pergerakan pemain pada intinya yaitu mengisi matriks dari nomor sampai 9. Jika posisi telah terisi maka akan menampilkan output bahwa posisi itu telah terisi seperti di bawah ini :

```
please select a position to enter the X between 1 to 9
1
Sorry, this space is occupied
please select a position to enter the X between 1 to 9
↑↓ for history. Search history with c-↑/c-↓
```

5. Fungsi untuk mengatur pergerakan computer

```
def computerMove():
    possibleMoves = [x for x, letter in enumerate(board) if letter == ' ' and x != 0]
    move = 0

    for let in ['O', 'X']:
        for i in possibleMoves:
            boardcopy = board[:]
            boardcopy[i] = let
            if IsWinner(boardcopy, let):
                move = i
                return move

    cornersOpen = []
    for i in possibleMoves:
        if i in [1, 3, 7, 9]:
            cornersOpen.append(i)

    if len(cornersOpen) > 0:
        move = selectRandom(cornersOpen)
        return move

    if 5 in possibleMoves:
        move = 5
        return move

    edgesOpen = []
    for i in possibleMoves:
        if i in [2, 4, 6, 8]:
            edgesOpen.append(i)

    if len(edgesOpen) > 0:
        move = selectRandom(edgesOpen)
        return move

    if len(possibleMoves) > 0:
        move = selectRandom(edgesOpen)
        return move
    return move
```

Fungsi diatas digunakan untuk mengatur pergerakan komputer. Inti dari kode program ini adalah memeriksa setiap sudut/corner, sisi/edges, dan pusat yang berada pada angka 5. Jika ada sudut yang kosong, komputer lebih cenderung memilih sudut tersebut. Jika pusat papan kosong, komputer akan memilih pusat. Jika tidak ada sudut atau pusat yang kosong, komputer akan memilih sisi.

6. Fungsi untuk memilih posisi random dari pergerakan komputer

```
def selectRandom(li):
    import random
    ln = len(li)
    r = random.randrange(0,ln)
    return li[r]
```

Fungsi diatas digunakan untuk memilih posisi random dari pergerakan pemain komputer. Fungsi ini dapat digunakan untuk membuat komputer mengambil keputusan secara acak.

7. Fungsi untuk menjalankan permainan

```
def main():
    print("Welcome to the game!")
    printBoard(board)

    while not(isBoardFull(board)):
        if not(IsWinner(board, 'O')):
            playerMove()
            printBoard(board)
        else:
            print("sorry you loose!")
            break

        if not(IsWinner(board, 'X')):
            move = computerMove()
            if move == 0:
                print(" ")
            else:
                insertLetter('O', move)
                print('computer placed an o on position', move, ':')
                printBoard(board)
        else:
            print("you win!")
            break

    if isBoardFull(board):
        print("Tie game")

while True:
    x = input("Do you want to play? Press y for yes or n for no (y/n)\n")
    if x.lower() == 'y':
        board = [' ' for x in range(10)]
        print('-----')
        main()
    else:
        break
```

Fungsi diatas yaitu fungsi utama untuk menjalankan permainan.

8. Contoh saat bermain :

a. Posisi Menang atau Win

```

      |
      |
please select a position to enter the X between 1 to 9
5
|
|
X | O | X
|
|
-----
|
|
O | X |
|
|
-----
|
|
X |   | O
|
|
you win!
Do you want to play? Press y for yes or n for no (y/n)
↑↓ for history. Search history with c-↑/c-↓
```

b. Posisi Seri atau Tie

```

      |
      |
please select a position to enter the X between 1 to 9
7
|
|
X | O | O
|
|
-----
|
|
O | X | X
|
|
-----
|
|
X | X | O
|
|
Tie game
Do you want to play? Press y for yes or n for no (y/n)
↑↓ for history. Search history with c-↑/c-↓
```

c. Posisi Kalah atau Lose

```

computer placed an o on position 7 :
  x |   | o
  ---
  x | o | 
  ---
  o |   | x
  ---
sorry you loose!
Do you want to play? Press y for yes or n for no (y/n)
↑↓ for history. Search history with c-↑/c-↓

```

D. Kendala Praktikum

Terdapat error kode program saat permainan dimulai dan hasilnya seri atau tie :

```

please select a position to enter the X between 1 to 9
7
  x |   | 
  ---
  o | x | x
  ---
  x | x | o
  ---

```

```

-----
TypeError                                Traceback (most recent call last)
Cell In[1], line 130
    128 board = [' ' for x in range(10)]
    129 print('-----')
--> 130 main()
    131 else:
    132     break

Cell In[1], line 115, in main()
    113 print(" ")
    114 else:
--> 115 insertLetter('O' , move)
    116 print('computer placed an o on position' , move , ':')
    117 printBoard(board)

Cell In[1], line 4, in insertLetter(letter, pos)
      3 def insertLetter(letter,pos):
----> 4     board[pos] = letter

TypeError: list indices must be integers or slices, not NoneType

```

Solusinya yaitu dengan memodifikasi kode program pada fungsi `computerMove()` dengan menambahkan baris kode seperti di bawah

```

if len(possibleMoves) > 0:
    move = selectRandom(edgesOpen)
    return move
return move

```

Jika tidak ada kode pemeriksaan `if len(possibleMoves) > 0`, program akan tetap mencoba menjalankan baris `move = selectRandom(edgesOpen)`. Namun, karena `possibleMoves` kosong, fungsi `selectRandom` tidak akan menemukan elemen untuk dipilih dan akan mengembalikan `None`. Ketika program mencoba menggunakan `None`

sebagai indeks untuk mengakses elemen dalam list, maka akan muncul error list indices must be integers or slices, not NoneType.

Hasilnya saat permainan hasilnya seri, maka program masih bisa dilanjutkan kembali

```

please select a position to enter the X between 1 to 9
7
  |  |  |
X | X | O
  |  |  |
-----
  |  |  |
O | X | X
  |  |  |
-----
  |  |  |
X | O | O
  |  |  |

Tie game
Do you want to play? Press y for yes or n for no (y/n)

```

E. Studi Kasus

Modifikasi kode program tersebut sehingga menampilkan board berukuran 4 × 4 seperti berikut:

X	O	O	X
X	O	O	X
X	O	O	X
X	O	O	X

Kode Program :

```

[9]: board = [' ' for x in range(17)]

def insertLetter(letter, pos):
    board[pos] = letter

def spaceIsFree(pos):
    return board[pos] == ' '

def printBoard(board):
    print('  |  |  |  ')
    print(' ' + board[1] + ' | ' + board[2] + ' | ' + board[3] + ' | ' + board[4])
    print('  |  |  |  ')
    print('-----')
    print('  |  |  |  ')
    print(' ' + board[5] + ' | ' + board[6] + ' | ' + board[7] + ' | ' + board[8])
    print('  |  |  |  ')
    print('-----')
    print('  |  |  |  ')
    print(' ' + board[9] + ' | ' + board[10] + ' | ' + board[11] + ' | ' + board[12])
    print('  |  |  |  ')
    print('-----')
    print('  |  |  |  ')
    print(' ' + board[13] + ' | ' + board[14] + ' | ' + board[15] + ' | ' + board[16])
    print('  |  |  |  ')

```

```

def isBoardFull(board):
    if board.count(' ') > 1:
        return False

def IsWinner(b, l):
    return (
        (b[1] == l and b[2] == l and b[3] == l and b[4] == l) or
        (b[5] == l and b[6] == l and b[7] == l and b[8] == l) or
        (b[9] == l and b[10] == l and b[11] == l and b[12] == l) or
        (b[13] == l and b[14] == l and b[15] == l and b[16] == l) or
        (b[1] == l and b[5] == l and b[9] == l and b[13] == l) or
        (b[2] == l and b[6] == l and b[10] == l and b[14] == l) or
        (b[3] == l and b[7] == l and b[11] == l and b[15] == l) or
        (b[4] == l and b[8] == l and b[12] == l and b[16] == l) or
        (b[1] == l and b[6] == l and b[11] == l and b[16] == l) or
        (b[4] == l and b[7] == l and b[10] == l and b[13] == l)
    )

def playerMove():
    run = True
    while run:
        move = input("Please select a position to enter the X between 1 to 16\n")
        try:
            move = int(move)
            if move > 0 and move < 17:
                if spaceIsFree(move):
                    run = False
                    insertLetter('X', move)
                else:
                    print('Sorry, this space is occupied')
            else:
                print('Please type a number between 1 and 16')
        except:
            print('Please type a number')

def computerMove():
    possibleMoves = [x for x, letter in enumerate(board) if letter == ' ' and x != 0]
    move = 0

    for let in ['O', 'X']:
        for i in possibleMoves:
            boardcopy = board[:]
            boardcopy[i] = let
            if IsWinner(boardcopy, let):
                move = i
                return move

    cornersOpen = []
    for i in possibleMoves:
        if i in [1, 4, 13, 16]:
            cornersOpen.append(i)

    if len(cornersOpen) > 0:
        move = selectRandom(cornersOpen)
        return move

    if 5 in possibleMoves:
        move = 5
        return move

```

```

edgesOpen = []
for i in possibleMoves:
    if i in [2, 3, 6, 7, 8, 9, 10, 11, 12, 14, 15]:
        edgesOpen.append(i)

if len(edgesOpen) > 0:
    move = selectRandom(edgesOpen)
    return move

def selectRandom(li):
    import random
    ln = len(li)
    r = random.randrange(0, ln)
    return li[r]

def main():
    print("Welcome to the game!")
    printBoard(board)

    while not(isBoardFull(board)):
        if not(IsWinner(board, 'O')):
            playerMove()
            printBoard(board)
        else:
            print("Sorry, you lose!")
            break

        if not(IsWinner(board, 'X')):
            move = computerMove()
            if move == 0:
                print(" ")
            else:
                insertLetter('O', move)
                print('Computer placed an O on position', move, ':')
                printBoard(board)
        else:
            print("You win!")
            break

    if isBoardFull(board):
        print("Tie game")

while True:
    x = input("Do you want to play? Press y for yes or n for no (y/n)\n")
    if x.lower() == 'y':
        board = [' ' for x in range(17)]
        print('-----')
        main()
    else:
        break

```

Pada studi kasus ini sebagian besar kode program sama dengan praktikum yang telah dilakukan, hanya dimodifikasi pada fungsi tertentu seperti matriks menjadi 4x4, lalu kemungkinan yang memungkinkan pemain untuk menang ditambah, dan possible move ditambah.

Hasil :

```

Computer placed an O on position 12 :
  x | x | o | o
  ---
  x | x | x | o
  ---
  x | o | o | o
  ---
  o | x | o | x
Tie game
Do you want to play? Press y for yes or n for no (y/n)

```

Gambar diatas merupakan contoh dari permainan Tic Tac Toe 4x4 dan hasilnya seri atau tie