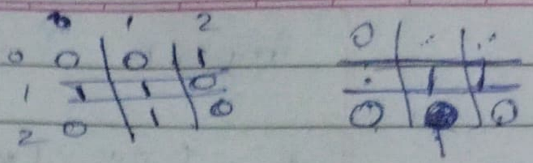
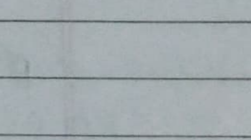
DATE: 24/9/2024 PAGE:

Tic Tac Toe



- Initialize 2D list with empty symbol (underscore)
- Initialize a map {0: [0,8] ___ }
- Initialize step = 0
- ~~for~~ Input user symbol
- other symbol for bot

```
                                    U  C
                                   (1, 2)
turn = Randomly choose user or bot & display.
• for i = 0 to 8
    if (step%2 == 0) or turn = 1)
        input (" ~~two~~ cells 0 to 8)
        ~~check~~ print (matrix)
          ~~random(  )~~
          check(sym)      step ++
    ~~elif ( turn==2  or~~


check(sym)
    if two of the winning condition satisfies ~~of~~
  put in third place

    else if random()
          Check for computer win()
    else if      draw    step==9
```



```
0 : [[0,1,2], [0,3,6], [0,4 7]]
1 : [[0,1,2], [1,4,7],]
2 : [[0,1,2], [2,4,6], [2,5,8])
3 : [
import random

d = [["-"]*3 for _ in range(3)]
turn = {0:[0,0], 1:[0,1], 2:[0,2], 3:[1,0],
        4:[1,1], 5:[1,2], 6:[2,0], 7:[2,1],
        8:[2,2]}

r0 = [0,1,2]; r1 = [3,4,5]; r2 = [6,7,8],
c0 = [0,3,6]; c1 = [1,4,7]; c2 = [2,5,8],
d1 = [0,4,8]; d2 = [2,4,6]

avoid win_turns = { 0:[r0, r1, r0], 1: [r0, c1],
    2:[r0, c2, d2], 3:[r0, c1], 4:[d1, d2, c1, r1],
    5:[r2, c1], 6:[c0, r2, d2], 7:[c2, r1],
    8:[c2, r2, d1] }

p1 = input(" player symbol (o or x)").strip().
                                         lower()

if p1 == "x":
    p2 = "o"
else:
    p2 = "x"

def player_move():
    while True:
        move = int(input("Enter your move
                         (0-8): "))
        if move in turn and d[turn[move][0]]
                 [turn[move][1]] == "-":
            d[turn[move][0]] [turn[move][1]] = p1
            break
```

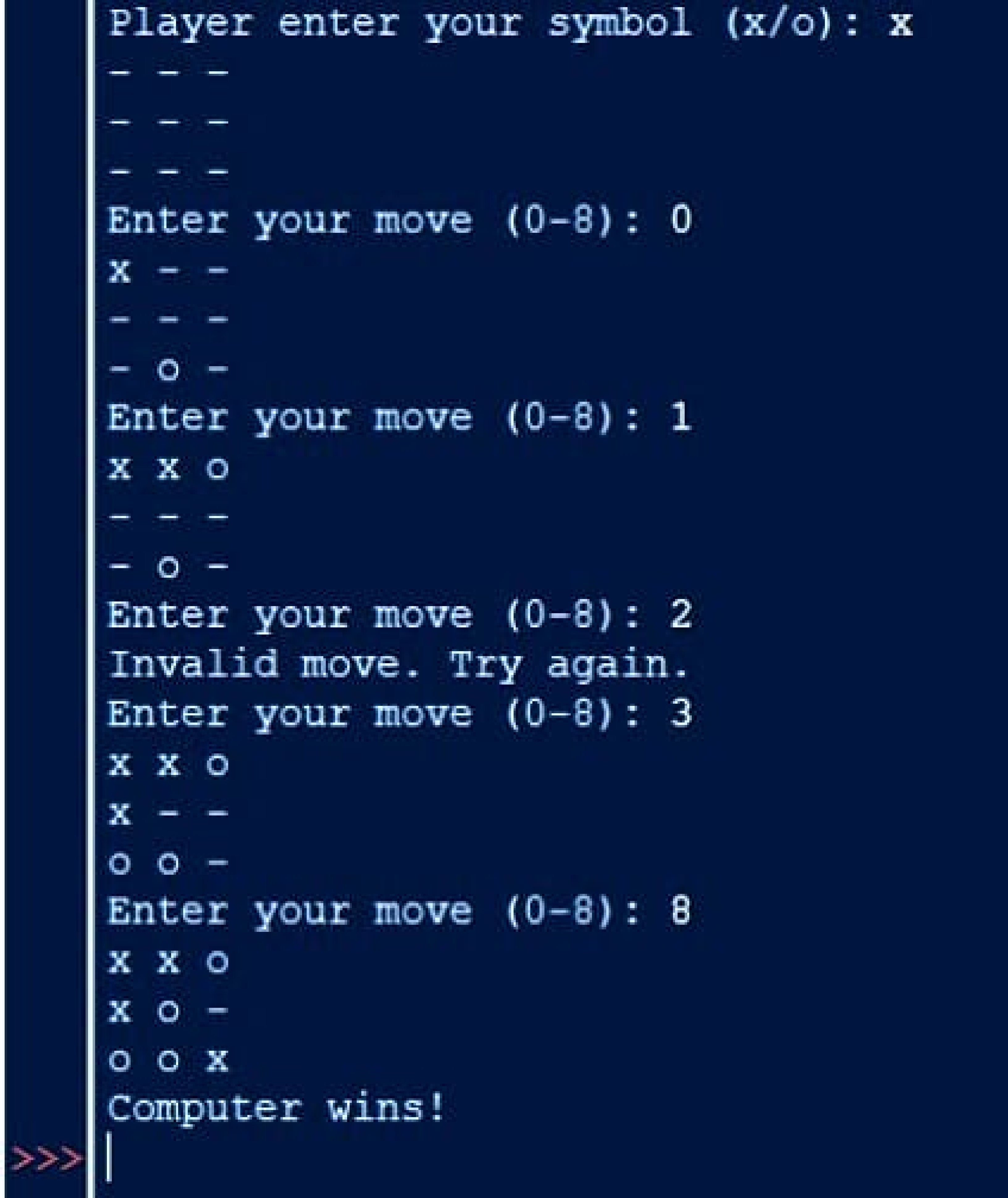
DATE: PAGE:

```
    else:
        print("Invalid")


def computer_move():
    available_moves = [ ]
    for i in range(9):
        if d[turn[i][0]][turn[i][1]] == "-":
            available_moves.append(i)
        if not available_moves:
            return

        for move in available_moves:
            if move in avoid_win_turns:
                possible_moves = avoid_window_
                                        turns[move]

            filtered_moves = [m for sublist
                in possible_moves for m
                    in sublist if m in
                        available_moves]
                if filtered_moves:
                    computer_choice =
                        random.choice(filtered_
                                       moves)
                  d[turn[computer_choice][0]]
                    [turn[computer_choice[1]] = p2
                  return


def print_board():
    for row in d:
        print(" ".join(row))
```

DATE: PAGE:

```
for  i  in range(5):
  print_board()
  player_more()
  if  any(row.count(p1) == 3  for row in
                          d)  or
      any(col.count(p1) == 3  for col in
          zip(*d))  or \

      d[0][0] == d[1][1] == d[2][2] == p1 or
      d[0][2] == d[1][1] == d[2][0] == p1:
        print_board()
        print("player  wins!")
        break.

     computer_more()
     if  any(row.count(p2)  == 3  for row in d) or \
       any(col.count(p2) == 3  for col in zip(*d)) or
       d[0][0] == d[1][1] == d[2][2] == p2  or \
       d[0][2] == d[1][1] == d[2][0] == p2:
          print board()
          print("computer  wins!")
          break

       else:

          print_board()
          print(" Draw
```

[illegible]



```
Player enter your symbol (x/o): x
- - -
- - -
- - -
Enter your move (0-8): 0
x - -
- - -
- o -
Enter your move (0-8): 1
x x o
- - -
- o -
Enter your move (0-8): 2
Invalid move. Try again.
Enter your move (0-8): 3
x x o
x - -
o o -
Enter your move (0-8): 8
x x o
x o -
o o x
Computer wins!
>>> |
```