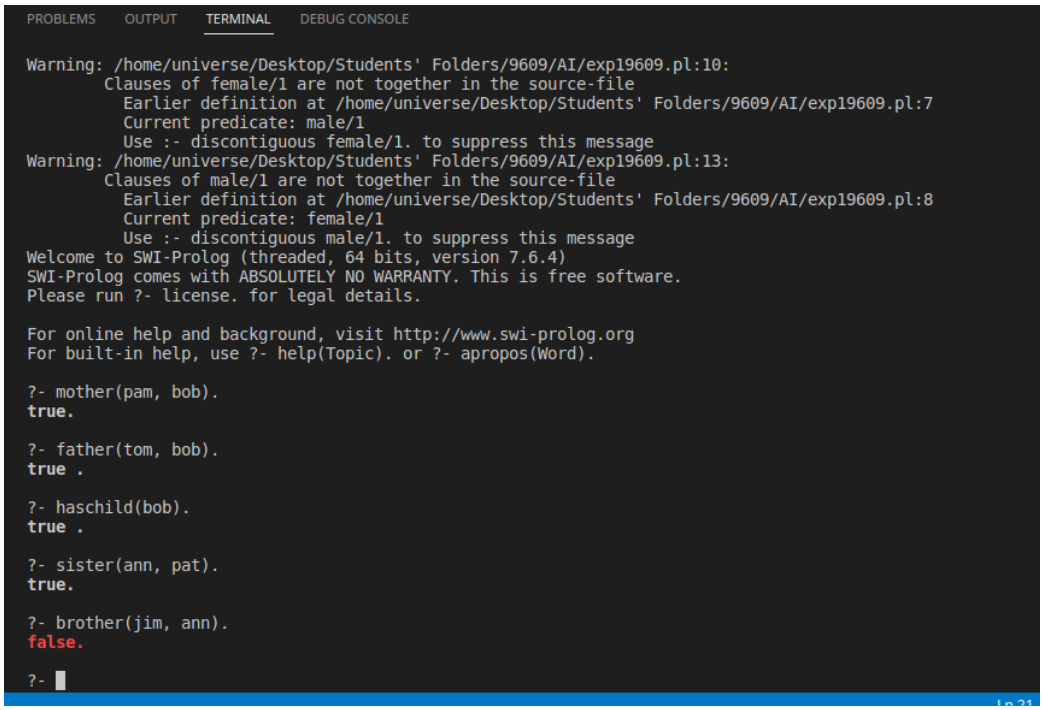
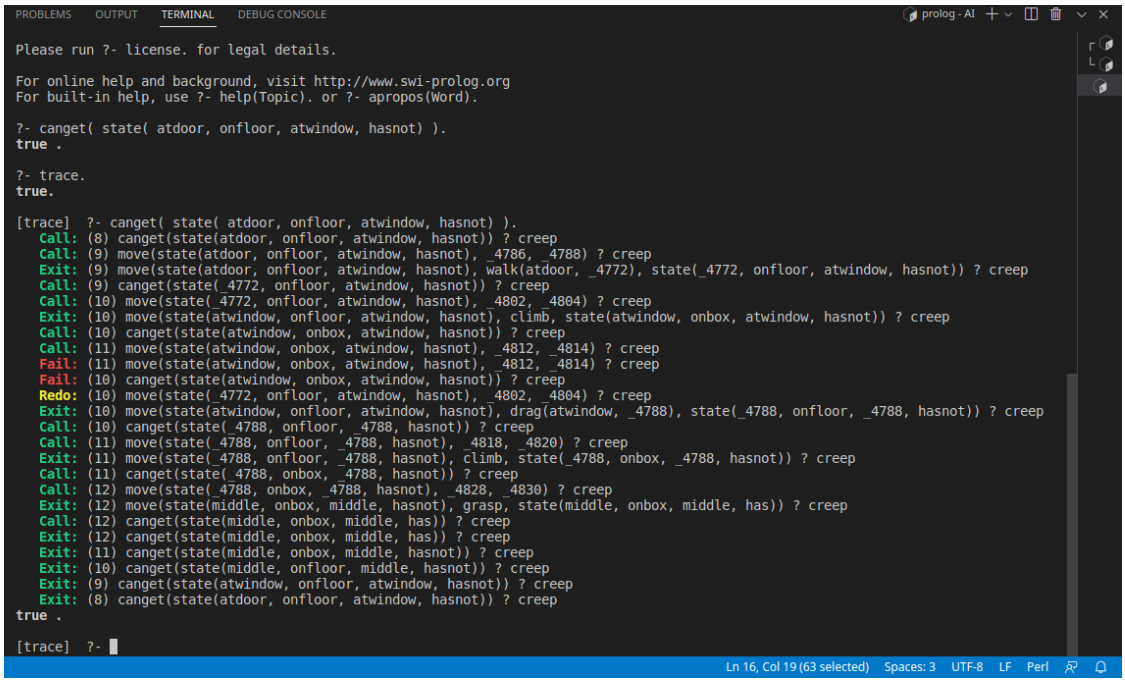
EXP 2:

```
parent(pam, bob). /*pam is parent of bob.*/
parent(tom, bob). /*tom is parent of bob.*/
parent(tom, liz). /*tom is parent of liz.*/
parent(bob, ann). /*bob is parent of ann.*/
parent(bob, pat). /*bob is parent of pat.*/
parent(pat, jim). /*pat is parent of jim.*/
female(pam). /*pam is female.*/
male(tom). /*tom is male.*/
male(bob). /*bob is male.*/
female(liz). /*liz is female.*/
female(pat). /*pat is female.*/
female(ann). /*ann is female.*/
male(jim). /*jim is male.*/

mother(X,Y):-parent(X,Y),female(X).
father(X,Y):-parent(X,Y),male(X).
haschild(X):-parent(X,_).
sister(X,Y):-parent(Z,X),parent(Z,Y),female(X),X\==Y.
brother(X,Y):-parent(Z,X),parent(Z,Y),male(X),X\==Y.
```



EXP 3:

```
move(state(middle,onbox,middle,hasnot),
   grasp,
   state(middle,onbox,middle,has)).
move(state(P,onfloor,P,H),
   climb,
   state(P,onbox,P,H)).
move(state(P1,onfloor,P1,H),
   drag(P1,P2),
   state(P2,onfloor,P2,H)).
move(state(P1,onfloor,B,H),
   walk(P1,P2),
   state(P2,onfloor,B,H)).
canget(state(_,_,_,has)).
canget(State1) :-
   move(State1,_,State2),
   canget(State2).
```



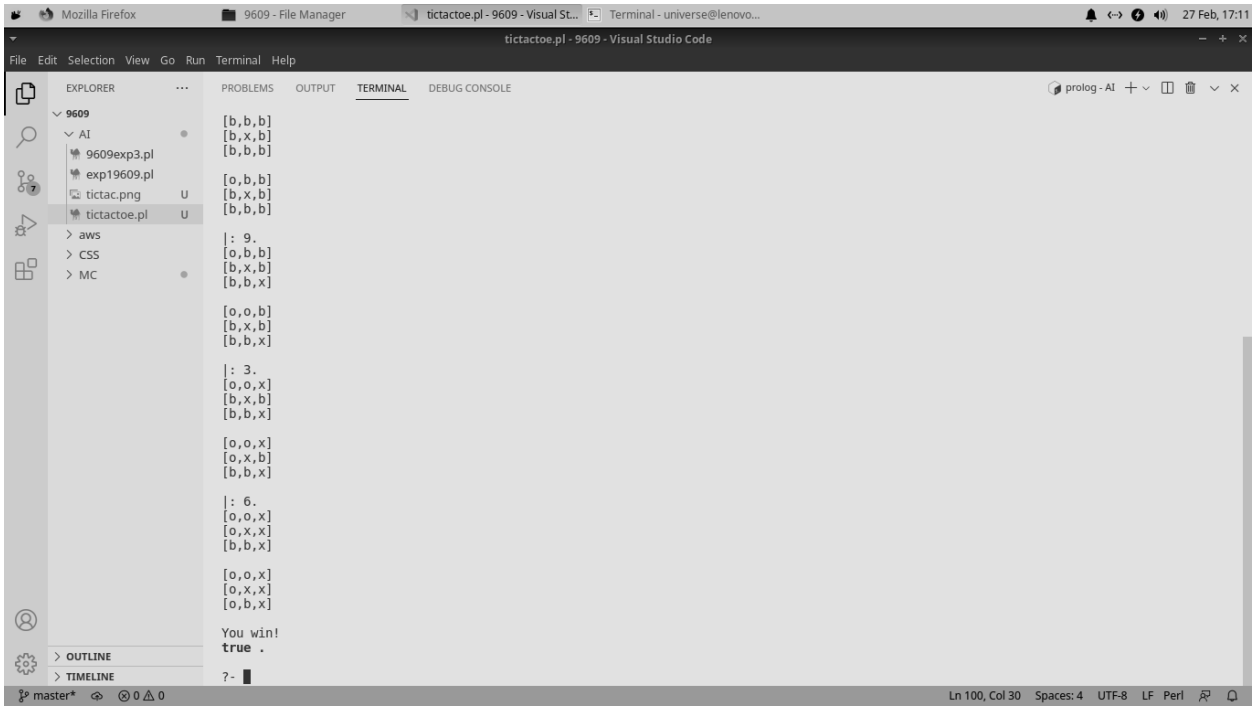
EXP 7:

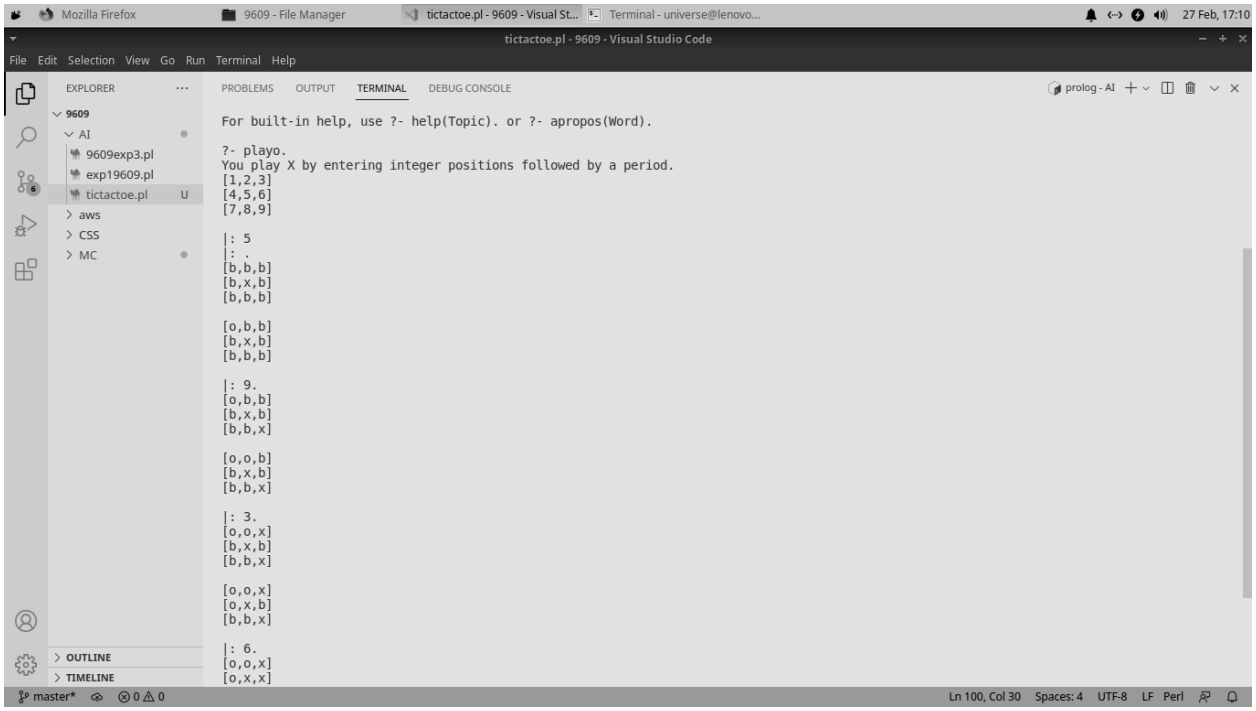
```
% A Tic-Tac-Toe program in Prolog.   S. Tanimoto, May 11, 2003.
% To play a game with the computer, type
% playo.
% To watch the computer play a game with itself, type
% selfgame.

% Predicates that define the winning conditions:

win(Board, Player) :- rowwin(Board, Player).
win(Board, Player) :- colwin(Board, Player).
win(Board, Player) :- diagwin(Board, Player).

rowwin(Board, Player) :- Board = [Player,Player,Player,_,_,_,_,_,_].
rowwin(Board, Player) :- Board = [_,_,_,Player,Player,Player,_,_,_].
rowwin(Board, Player) :- Board = [_,_,_,_,_,_,Player,Player,Player].

colwin(Board, Player) :- Board = [Player,_,_,Player,_,_,Player,_,_].
colwin(Board, Player) :- Board = [_,Player,_,_,Player,_,_,Player,_].
colwin(Board, Player) :- Board = [_,_,Player,_,_,Player,_,_,Player].

diagwin(Board, Player) :- Board = [Player,_,_,_,Player,_,_,_,Player].
diagwin(Board, Player) :- Board = [_,_,Player,_,Player,_,Player,_,_].

% Helping predicate for alternating play in a "self" game:

other(x,o).
other(o,x).

game(Board, Player) :- win(Board, Player), !, write([player, Player, wins]).
game(Board, Player) :-
  other(Player,Otherplayer),
  move(Board,Player,Newboard),
  !,
  display(Newboard),
  game(Newboard,Otherplayer).

move([b,B,C,D,E,F,G,H,I], Player, [Player,B,C,D,E,F,G,H,I]).
move([A,b,C,D,E,F,G,H,I], Player, [A,Player,C,D,E,F,G,H,I]).
move([A,B,b,D,E,F,G,H,I], Player, [A,B,Player,D,E,F,G,H,I]).
move([A,B,C,b,E,F,G,H,I], Player, [A,B,C,Player,E,F,G,H,I]).
move([A,B,C,D,b,F,G,H,I], Player, [A,B,C,D,Player,F,G,H,I]).
move([A,B,C,D,E,b,G,H,I], Player, [A,B,C,D,E,Player,G,H,I]).
```

```
move([A,B,C,D,E,F,b,H,I], Player, [A,B,C,D,E,F,Player,H,I]).
move([A,B,C,D,E,F,G,b,I], Player, [A,B,C,D,E,F,G,Player,I]).
move([A,B,C,D,E,F,G,H,b], Player, [A,B,C,D,E,F,G,H,Player]).

display([A,B,C,D,E,F,G,H,I]) :- write([A,B,C]),nl,write([D,E,F]),nl,
 write([G,H,I]),nl,nl.

selfgame :- game([b,b,b,b,b,b,b,b,b],x).

% Predicates to support playing a game with the user:

x_can_win_in_one(Board) :- move(Board, x, Newboard), win(Newboard, x).

% The predicate orespond generates the computer's (playing o) reponse
% from the current Board.

orespond(Board,Newboard) :-
  move(Board, o, Newboard),
  win(Newboard, o),
  !.
orespond(Board,Newboard) :-
  move(Board, o, Newboard),
  not(x_can_win_in_one(Newboard)).
orespond(Board,Newboard) :-
  move(Board, o, Newboard).
orespond(Board,Newboard) :-
  not(member(b,Board)),
  !,
  write('Cats game!'), nl,
  Newboard = Board.

% The following translates from an integer description
% of x's move to a board transformation.

xmove([b,B,C,D,E,F,G,H,I], 1, [x,B,C,D,E,F,G,H,I]).
xmove([A,b,C,D,E,F,G,H,I], 2, [A,x,C,D,E,F,G,H,I]).
xmove([A,B,b,D,E,F,G,H,I], 3, [A,B,x,D,E,F,G,H,I]).
xmove([A,B,C,b,E,F,G,H,I], 4, [A,B,C,x,E,F,G,H,I]).
xmove([A,B,C,D,b,F,G,H,I], 5, [A,B,C,D,x,F,G,H,I]).
xmove([A,B,C,D,E,b,G,H,I], 6, [A,B,C,D,E,x,G,H,I]).
xmove([A,B,C,D,E,F,b,H,I], 7, [A,B,C,D,E,F,x,H,I]).
xmove([A,B,C,D,E,F,G,b,I], 8, [A,B,C,D,E,F,G,x,I]).
xmove([A,B,C,D,E,F,G,H,b], 9, [A,B,C,D,E,F,G,H,x]).
xmove(Board, N, Board) :- write('Illegal move.'), nl.
```

```
% The 0-place predicate playo starts a game with the user.

playo :- explain, playfrom([b,b,b,b,b,b,b,b,b]).

explain :-
  write('You play X by entering integer positions followed by a period.'),
  nl,
  display([1,2,3,4,5,6,7,8,9]).

playfrom(Board) :- win(Board, x), write('You win!').
playfrom(Board) :- win(Board, o), write('I win!').
playfrom(Board) :- read(N),
  xmove(Board, N, Newboard),
  display(Newboard),
  orespond(Newboard, Newnewboard),
  display(Newnewboard),
  playfrom(Newnewboard).
```

OUTPUT:


Mozilla Firefox
9609 - File Manager
tictactoe.pl - 9609 - Visual St...
Terminal - universe@lenovo...
27 Feb, 17:11
tictactoe.pl - 9609 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
prolog - AI
9609
AI
9609exp3.pl
exp19609.pl
tictac.png U
tictactoe.pl U
aws
CSS
MC
[b,b,b]
[b,x,b]
[b,b,b]
[o,b,b]
[b,x,b]
[b,b,b]
|: 9.
[o,b,b]
[b,x,b]
[b,b,x]
[o,o,b]
[b,x,b]
[b,b,x]
|: 3.
[o,o,x]
[b,x,b]
[b,b,x]
[o,o,x]
[o,x,b]
[b,b,x]
|: 6.
[o,o,x]
[o,x,x]
[b,b,x]
[o,o,x]
[o,x,x]
[o,b,x]
You win!
true .
?-
OUTLINE
TIMELINE
master*
0 0
Ln 100, Col 30 Spaces: 4 UTF-8 LF Perl


Mozilla Firefox
9609 - File Manager
tictactoe.pl - 9609 - Visual St...
Terminal - universe@lenovo...
27 Feb, 17:10
tictactoe.pl - 9609 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
prolog - AI
9609
AI
9609exp3.pl
exp19609.pl
tictactoe.pl U
aws
CSS
MC
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- playo.
You play X by entering integer positions followed by a period.
[1,2,3]
[4,5,6]
[7,8,9]
|: 5
|: .
[b,b,b]
[b,x,b]
[b,b,b]
[o,b,b]
[b,x,b]
[b,b,b]
|: 9.
[o,b,b]
[b,x,b]
[b,b,x]
[o,o,b]
[b,x,b]
[b,b,x]
|: 3.
[o,o,x]
[b,x,b]
[b,b,x]
[o,o,x]
[o,x,b]
[b,b,x]
|: 6.
[o,o,x]
[o,x,x]
OUTLINE
TIMELINE
master*
0 0
Ln 100, Col 30 Spaces: 4 UTF-8 LF Perl

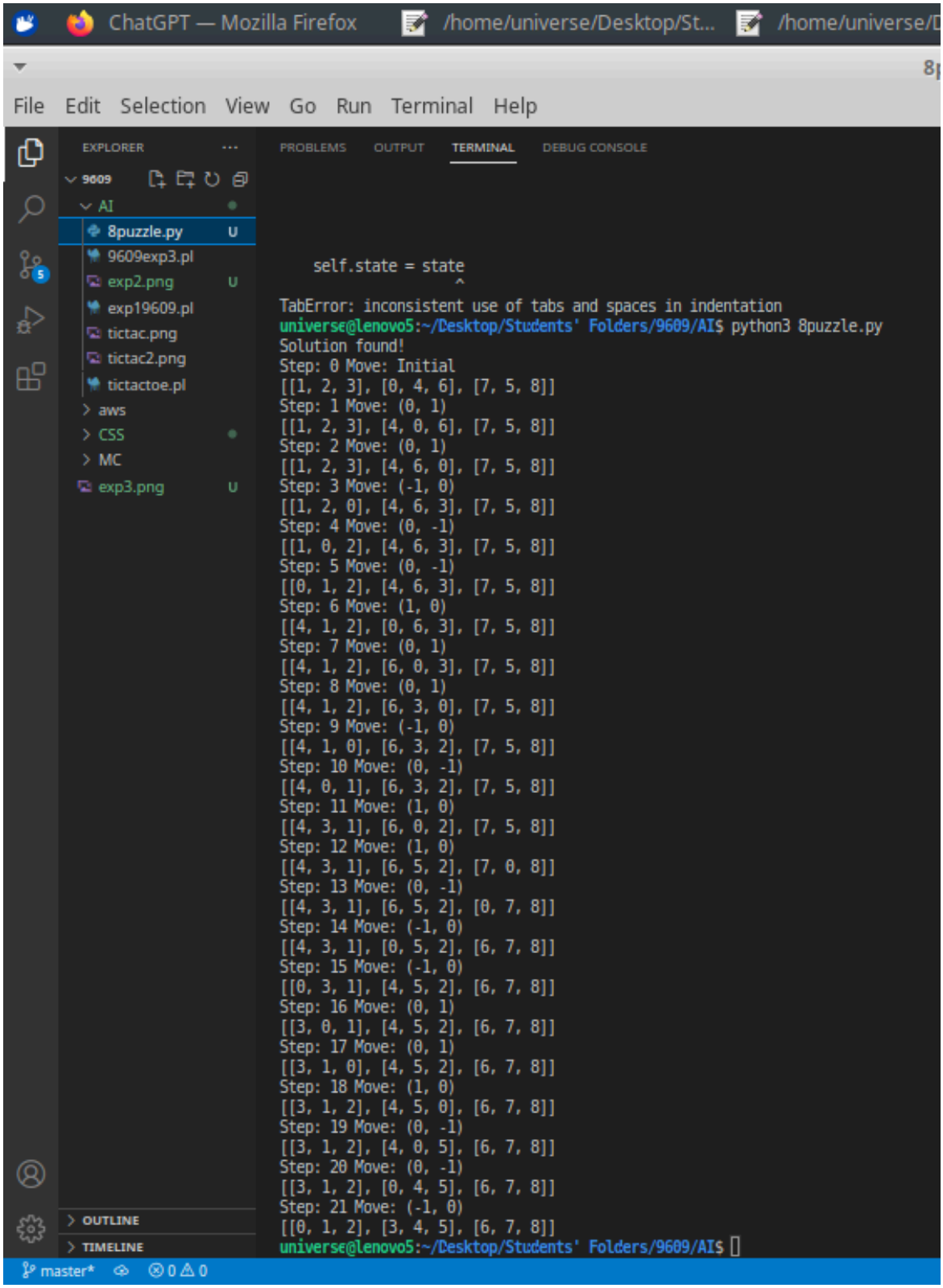
EXP 6:

```
import heapq

class PuzzleNode:
        def __init__(self, state, parent=None, move="Initial"):
        self.state = state
        self.parent = parent
        self.move = move
        self.depth = 0
        if parent:
        self.depth = parent.depth + 1

        def __lt__(self, other):
        return self.depth < other.depth

        def __eq__(self, other):
        return self.state == other.state

        def __hash__(self):
        return hash(str(self.state))

        def __str__(self):
        return str(self.state)

        def get_blank_position(self):
        for i, row in enumerate(self.state):
        if 0 in row:
                return (i, row.index(0))

        def expand(self):
        blank_pos = self.get_blank_position()
        children = []
        moves = [(0, 1), (0, -1), (1, 0), (-1, 0)]  # right, left, down, up

        for move in moves:
        new_row, new_col = blank_pos[0] + move[0], blank_pos[1] + move[1]
        if 0 <= new_row < 3 and 0 <= new_col < 3:
                new_state = [row[:] for row in self.state]
                new_state[blank_pos[0]][blank_pos[1]] = new_state[new_row][new_col]
                new_state[new_row][new_col] = 0
                children.append(PuzzleNode(new_state, self, move))
        return children
```

```
        def is_goal(self):
        return self.state == [[0, 1, 2],
                              [3, 4, 5],
                              [6, 7, 8]]

        def heuristic(self):
        # Manhattan distance heuristic
        distance = 0
        for i in range(3):
        for j in range(3):
                if self.state[i][j] != 0:
                row, col = divmod(self.state[i][j], 3)
                distance += abs(row - i) + abs(col - j)
        return distance + self.depth


def a_star_search(initial_state):
        initial_node = PuzzleNode(initial_state)
        frontier = []
        heapq.heappush(frontier, initial_node)

        explored = set()

        while frontier:
        current_node = heapq.heappop(frontier)

        if current_node.is_goal():
        return current_node

        explored.add(current_node)

        for child in current_node.expand():
        if child not in explored:
                heapq.heappush(frontier, child)

        return None


def print_solution(solution_node):
        path = []
        current_node = solution_node
        while current_node:
        path.append(current_node)
        current_node = current_node.parent
```

```
        path.reverse()

        for i, node in enumerate(path):
        print("Step:", i, "Move:", node.move)
        print(node.state)


if __name__ == "__main__":
        initial_state = [[1, 2, 3], [0, 4, 6], [7, 5, 8]]
        solution_node = a_star_search(initial_state)
        if solution_node:
        print("Solution found!")
        print_solution(solution_node)
        else:
        print("No solution found!")
```

Output:


ChatGPT — Mozilla Firefox
/home/universe/Desktop/St...
/home/universe/D
File Edit Selection View Go Run Terminal Help
EXPLORER
9609
AI
8puzzle.py U
9609exp3.pl
exp2.png U
exp19609.pl
tictac.png
tictac2.png
tictactoe.pl
aws
CSS
MC
exp3.png U
OUTLINE
TIMELINE
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
self.state = state
^
TabError: inconsistent use of tabs and spaces in indentation
universe@lenovo5:~/Desktop/Students' Folders/9609/AI$ python3 8puzzle.py
Solution found!
Step: 0 Move: Initial
[[1, 2, 3], [0, 4, 6], [7, 5, 8]]
Step: 1 Move: (0, 1)
[[1, 2, 3], [4, 0, 6], [7, 5, 8]]
Step: 2 Move: (0, 1)
[[1, 2, 3], [4, 6, 0], [7, 5, 8]]
Step: 3 Move: (-1, 0)
[[1, 2, 0], [4, 6, 3], [7, 5, 8]]
Step: 4 Move: (0, -1)
[[1, 0, 2], [4, 6, 3], [7, 5, 8]]
Step: 5 Move: (0, -1)
[[0, 1, 2], [4, 6, 3], [7, 5, 8]]
Step: 6 Move: (1, 0)
[[4, 1, 2], [0, 6, 3], [7, 5, 8]]
Step: 7 Move: (0, 1)
[[4, 1, 2], [6, 0, 3], [7, 5, 8]]
Step: 8 Move: (0, 1)
[[4, 1, 2], [6, 3, 0], [7, 5, 8]]
Step: 9 Move: (-1, 0)
[[4, 1, 0], [6, 3, 2], [7, 5, 8]]
Step: 10 Move: (0, -1)
[[4, 0, 1], [6, 3, 2], [7, 5, 8]]
Step: 11 Move: (1, 0)
[[4, 3, 1], [6, 0, 2], [7, 5, 8]]
Step: 12 Move: (1, 0)
[[4, 3, 1], [6, 5, 2], [7, 0, 8]]
Step: 13 Move: (0, -1)
[[4, 3, 1], [6, 5, 2], [0, 7, 8]]
Step: 14 Move: (-1, 0)
[[4, 3, 1], [0, 5, 2], [6, 7, 8]]
Step: 15 Move: (-1, 0)
[[0, 3, 1], [4, 5, 2], [6, 7, 8]]
Step: 16 Move: (0, 1)
[[3, 0, 1], [4, 5, 2], [6, 7, 8]]
Step: 17 Move: (0, 1)
[[3, 1, 0], [4, 5, 2], [6, 7, 8]]
Step: 18 Move: (1, 0)
[[3, 1, 2], [4, 5, 0], [6, 7, 8]]
Step: 19 Move: (0, -1)
[[3, 1, 2], [4, 0, 5], [6, 7, 8]]
Step: 20 Move: (0, -1)
[[3, 1, 2], [0, 4, 5], [6, 7, 8]]
Step: 21 Move: (-1, 0)
[[0, 1, 2], [3, 4, 5], [6, 7, 8]]
universe@lenovo5:~/Desktop/Students' Folders/9609/AI$
master*
0 0