Assignment-5

1. Write a Prolog program to implement the Water Jug Problem:
You are given two jugs: a 4-gallon jug and a 3-gallon jug. Neither has any measuring markers on it. There is a tap that can be used to fill the jugs with water. How can you fill a 4-gallon jug with exactly 2 gallons of water?

Production Rules:-

- **R1: (x,y) --> (4,y) if x < 4**
- **R2: (x,y) --> (x,3) if y < 3**
- **R3: (x,y) --> (x-d,y) if x > 0**
- **R4: (x,y) --> (x,y-d) if y > 0**
- **R5: (x,y) --> (0,y) if x > 0**
- **R6: (x,y) --> (x,0) if y > 0**
- **R7: (x,y) --> (4,y-(4-x)) if x+y >= 4 and y > 0**
- **R8: (x,y) --> (x-(3-y),y) if x+y >= 3 and x > 0**
- **R9: (x,y) --> (x+y,0) if x+y =< 4 and y > 0**
- **R10: (x,y) --> (0,x+y) if x+y =< 3 and x > 0**

code.pl

```
% Define goal state
goal((2,_)).

% Define possible moves (production rules)
move((X,Y), (4,Y)) :- X < 4. % R1: Fill 4-gallon jug
move((X,Y), (X,3)) :- Y < 3. % R2: Fill 3-gallon jug
move((X,Y), (0,Y)) :- X > 0. % R5: Empty 4-gallon jug
move((X,Y), (X,0)) :- Y > 0. % R6: Empty 3-gallon jug
move((X,Y), (4, Y1)) :- X + Y >= 4, Y > 0, Y1 is Y - (4 - X). % R7: Pour Y -> X until
X full
move((X,Y), (X1, 3)) :- X + Y >= 3, X > 0, X1 is X - (3 - Y). % R8: Pour X -> Y until
Y full
move((X,Y), (X1,0)) :- X + Y =< 4, Y > 0, X1 is X + Y. % R9: Pour Y -> X completely
move((X,Y), (0,Y1)) :- X + Y =< 3, X > 0, Y1 is X + Y. % R10: Pour X -> Y completely

% Search algorithm (DFS)
path(State, Goal, Visited, [State|Path]) :-
    move(State, NextState),
    \+ member(NextState, Visited),
    path(NextState, Goal, [NextState|Visited], Path).

path(State, State, _, [State]).
```

```
% Solve from initial state (0,0)
solve :-
    path((0,0), (2,_), [(0,0)], Path),
    print_path(Path).

% Print solution steps
print_path([]).
print_path([H|T]) :-
    write(H), nl,
    print_path(T).
```

```
debargha@HP-Pavilion:~/MTech/CS1051/PKG/Assignment-5$ swipl code.pl
Welcome to SWI-Prolog (threaded, 64 bits, version 9.2.9)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [code].
true.

?- solve.
0,0
4,0
4,3
0,3
3,0
3,3
4,2
0,2
2,0
2,3
true .
```