AI Project 2 Report

1 Actions

All the actions are implemented under the predicate of ***'helper_goal'***
The predicate symbols are

1. **Ex**: denotes the X position of Ethan.
2. **Ey**: denotes the Y position of Ethan.
3. **L**: denotes the List of positions of soldiers.
4. **Sx**: denotes the X position of the Submarine.
5. **Sy**: denotes the Y position of the Submarine.
6. **C**: denotes the remaining Capacity of Ethan.
7. **Cp**: denotes the fixed Capacity of Ethan.
8. **S**: denotes the current State of taking actions.
9. **G**: denotes the Goal state.

The predicate is implemented 7 times.

1. **Goal State**: is when Ethan in Submarine, no soldiers left, ethan dropped all the soldiers in the submarine "C==Cp".
2. **move up**: decreasing Ex by one.
3. **move down**: increasing Ex by one.
4. **move left**: decreasing Ey by one.
5. **move right**: increasing Ey by one.
6. **carry**: is when C>0 and there is soldier in the current cell.
7. **drop**: is when Ethan in submarine and Ethan is carrying soldiers.

2 goal(S)

In the predicate we extract the Knowledge from KB.pl and then calling ***'helper_goal'***
with the following parameters.

1. Ex
2. Ey
3. L
4. Sx
5. Sy
6. C
7. C -> which denotes Cp in helper_goal
8. s0 -> which denotes S in helper_goal
9. S -> which denotes G in helper_goal

We are calling the predicate with the predicate 'call_with_depth_limit(Goal,L,R)' and we set the Limit to be 12 for the specific example, because if less it won't get the answer and if more it would repeat some actions like up-> down-> up->....
until it goes right. Also, we made sure that the R is not depth_limit_exceeded.

3 Running Examples

3.1 First one

- **KB**
  - ethan_loc(0,3).
  - members_loc([[1,0],[2,1]]).
  - submarine(3,2).
  - capacity(2).
- Result: S = result(drop, result(right, result(down, result(carry, result(right, result(down, result(carry, result(left, result(left, result(left, result(down, s0))))))))))

3.2 Second one

- **KB**
  - ethan_loc(1,1).
  - members_loc([[2,2]]).
  - submarine(3,3).

  - capacity(1).

- Result: S = result(drop, result(right, result(down, result(carry, result(right, result(down, s0))))))