DOCUMENTATION FOR ARTIFIAL INTELLIGENCE ASSIGNMENT-2

Question-1 Hill climbing informed searching algorithm.

INPUT-

Start state-
2 8 3
1 6 4
7 0 5

Goal state-
1 2 3
8 0 4
7 6 5

OUTPUT-

```
2 8 3
1 6 4
7 0 5
Node: 0
Depth: 0
Moves: []
--------------------------------------------------------------------------
1.sum of Manhattan distance of each tile from the goal position
2.number of tiles displaced from their destined position
1
[18]
2 8 3
1 0 4
7 6 5
Node: 1
Depth: 1
Moves: ['up']
-------------------------------------------------------------------------
2 8 3
1 6 4
0 7 5
Node: 2
Depth: 1
Moves: ['left']
-------------------------------------------------------------------------
2 8 3
1 6 4
7 5 0
Node: 3
Depth: 1
Moves: ['right']
-------------------------------------------------------------------------
[16, 16, 20]
2 0 3
1 8 4
7 6 5
Node: 4
Depth: 2
Moves: ['up', 'up']
```

```
-----------------------------------------------------------------------
2 8 3
0 1 4
7 6 5
Node: 5
Depth: 2
Moves: ['up', 'left']
-----------------------------------------------------------------------
2 8 3
1 4 0
7 6 5
Node: 6
Depth: 2
Moves: ['up', 'right']
-----------------------------------------------------------------------
[14, 14, 16]
0 2 3
1 8 4
7 6 5
Node: 7
Depth: 3
Moves: ['up', 'up', 'left']
-----------------------------------------------------------------------
2 3 0
1 8 4
7 6 5
Node: 8
Depth: 3
Moves: ['up', 'up', 'right']
------------------------------------------------------------------------
[12, 14]
1 2 3
0 8 4
7 6 5
Node: 9
Depth: 4
Moves: ['up', 'up', 'left', 'down']
------------------------------------------------------------------------
[12]
1 2 3
7 8 4
0 6 5
Node: 10
Depth: 5
Moves: ['up', 'up', 'left', 'down', 'down']
-----------------------------------------------------------------------
1 2 3
8 0 4
7 6 5
Node: 11
Depth: 5
Moves: ['up', 'up', 'left', 'down', 'right']
-----------------------------------------------------------------------
Success
Time: 0.014479875564575195
-----------------------------------------------------------------------
```

HILL CLIMBING -

Hill climbing is a inform search algorithm and it based on Greedy Local Search.

-> Local search: use single current state and move to neighboring states.
-> Are also useful for pure optimization problems. Find best state according to

some objective function.
-> State space landscape.
  ▪ Location (defined by state).
-> Elevation.
  ▪ Defined by the value of the heuristic function or objective function.
-> Elevation corresponds to cost.
  ▪ Global minimum (aim is to find the lowest valley).
-> Elevation corresponds to an objective function.
  ▪ Global maximum (aim is to find the highest peak).
-> Complete algorithm always finds a goal if one exists.
-> Optimal algorithm always finds global minimum/maximum.

HILL CLIMBING SEARCH

-> "is a loop that continuously moves in the direction of increasing value".
-> It terminates when a peak is reached.
-> No neighbor has higher value.
-> does not maintain any search tree.
-> Current node data structure records state and objective function value.
-> does not look ahead beyond the immediate neighbors of the current state.
-> chooses randomly among the set of best successors, if there is more than one.
-> Hill-climbing a.k.a. greedy local search.
-> Grabs a good neighbor state without thinking ahead about where to go next.
-> Makes very rapid progress towards a solution.
-> Quite easy to improve a bad state.
-> Some problem spaces are great for hill climbing and others are terrible.

ALGORITHM

```
function HILL-CLIMBING(problem) return a state that is a
   (global) maximum
   input: problem, a problem
   local variables: current, a node.
       neighbor, a node.
   Current <--MAKE-NODE(INITIAL-STATE[problem])
   loop do
       neighbor <-- a highest valued successor of current
       if VALUE [neighbor] ≤ VALUE[current] then return
   STATE[current]
      current <-- neighbor
```

ADVANTAGE

1. Uses very little memory.
2. Finds often reasonable solutions in large or infinite state spaces.

DRAWBACKS

1. Local Maxima
   (A) peaks that aren't the highest point in the space (below global maxima).

```
   (B) higher than each of its neighboring states.
2. Plateau: region where evaluation function is flat.
   (A) Flat local maximum: no uphill exit exists.
   (B) Shoulder: possible to make progress.
   (C) Could give the search algorithm no direction
       (random walk) for local maximum.
3. Ridges:
   (A) flat like a plateau, but with drop-offs to the sides;
       steps to the North, East, South and West may go
       down, but a combination of two steps (e.g. N, W) may
       go up.
   (B) results in a sequence of local maximum not
       connected to each other.
```

Question-2 Simulated Annealing informed search.

INPUT-

Start state-
2 8 3
1 6 4
7 0 5

Goal state-
1 2 3
8 0 4
7 6 5

OUTPUT-

```
Started running simulated annealing:

goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 6, 4, 7, 0, 5]



Choosing random movement:E
state after move:[2, 8, 3, 1, 6, 4, 7, 5, 0]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 6, 4, 7, 5, 0]



Choosing random movement:W
state after move:[2, 8, 3, 1, 6, 4, 7, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 6, 4, 7, 0, 5]



Choosing random movement:N
state after move:[2, 8, 3, 1, 0, 4, 7, 6, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 0, 4, 7, 6, 5]

Choosing random movement:S
state after move:[2, 8, 3, 1, 6, 4, 7, 0, 5]
```

```
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 6, 4, 7, 0, 5]



Choosing random movement:N
state after move:[2, 8, 3, 1, 0, 4, 7, 6, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 0, 4, 7, 6, 5]



Choosing random movement:S
state after move:[2, 8, 3, 1, 6, 4, 7, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 6, 4, 7, 0, 5]



Choosing random movement:W
state after move:[2, 8, 3, 1, 6, 4, 0, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 6, 4, 0, 7, 5]



Choosing random movement:E
state after move:[2, 8, 3, 1, 6, 4, 7, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 6, 4, 7, 0, 5]



Choosing random movement:W
state after move:[2, 8, 3, 1, 6, 4, 0, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 1, 6, 4, 0, 7, 5]



Choosing random movement:N
state after move:[2, 8, 3, 0, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 0, 6, 4, 1, 7, 5]



Choosing random movement:N
state after move:[0, 8, 3, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[0, 8, 3, 2, 6, 4, 1, 7, 5]



Choosing random movement:E
state after move:[8, 0, 3, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 0, 3, 2, 6, 4, 1, 7, 5]



Choosing random movement:W
```

```
state after move:[0, 8, 3, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[0, 8, 3, 2, 6, 4, 1, 7, 5]



Choosing random movement:E
state after move:[8, 0, 3, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 0, 3, 2, 6, 4, 1, 7, 5]



Choosing random movement:E
state after move:[8, 3, 0, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 3, 0, 2, 6, 4, 1, 7, 5]



Choosing random movement:S
state after move:[8, 3, 4, 2, 6, 0, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 3, 4, 2, 6, 0, 1, 7, 5]



Choosing random movement:N
state after move:[8, 3, 0, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 3, 0, 2, 6, 4, 1, 7, 5]



Choosing random movement:W
state after move:[8, 0, 3, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 0, 3, 2, 6, 4, 1, 7, 5]



Choosing random movement:W
state after move:[0, 8, 3, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[0, 8, 3, 2, 6, 4, 1, 7, 5]



Choosing random movement:S
state after move:[2, 8, 3, 0, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[2, 8, 3, 0, 6, 4, 1, 7, 5]



Choosing random movement:N
state after move:[0, 8, 3, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[0, 8, 3, 2, 6, 4, 1, 7, 5]
```

```
Choosing random movement:E
state after move:[8, 0, 3, 2, 6, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 0, 3, 2, 6, 4, 1, 7, 5]



Choosing random movement:S
state after move:[8, 6, 3, 2, 0, 4, 1, 7, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 0, 4, 1, 7, 5]



Choosing random movement:S
state after move:[8, 6, 3, 2, 7, 4, 1, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 0, 5]



Choosing random movement:W
state after move:[8, 6, 3, 2, 7, 4, 0, 1, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 0, 1, 5]



Choosing random movement:E
state after move:[8, 6, 3, 2, 7, 4, 1, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 0, 5]



Choosing random movement:E
state after move:[8, 6, 3, 2, 7, 4, 1, 5, 0]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 5, 0]



Choosing random movement:N
state after move:[8, 6, 3, 2, 7, 0, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 0, 1, 5, 4]



Choosing random movement:N
state after move:[8, 6, 0, 2, 7, 3, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 0, 2, 7, 3, 1, 5, 4]



Choosing random movement:W
state after move:[8, 0, 6, 2, 7, 3, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 0, 6, 2, 7, 3, 1, 5, 4]
```

```
Choosing random movement:S
state after move:[8, 7, 6, 2, 0, 3, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 7, 6, 2, 0, 3, 1, 5, 4]



Choosing random movement:N
state after move:[8, 0, 6, 2, 7, 3, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 0, 6, 2, 7, 3, 1, 5, 4]



Choosing random movement:E
state after move:[8, 6, 0, 2, 7, 3, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 0, 2, 7, 3, 1, 5, 4]



Choosing random movement:S
state after move:[8, 6, 3, 2, 7, 0, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 0, 1, 5, 4]



Choosing random movement:W
state after move:[8, 6, 3, 2, 0, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 0, 7, 1, 5, 4]



Choosing random movement:W
state after move:[8, 6, 3, 0, 2, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 0, 2, 7, 1, 5, 4]



Choosing random movement:S
state after move:[8, 6, 3, 1, 2, 7, 0, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 1, 2, 7, 0, 5, 4]



Choosing random movement:N
state after move:[8, 6, 3, 0, 2, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 0, 2, 7, 1, 5, 4]



Choosing random movement:N
state after move:[0, 6, 3, 8, 2, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[0, 6, 3, 8, 2, 7, 1, 5, 4]
```

```
Choosing random movement:S
state after move:[8, 6, 3, 0, 2, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 0, 2, 7, 1, 5, 4]



Choosing random movement:E
state after move:[8, 6, 3, 2, 0, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 0, 7, 1, 5, 4]



Choosing random movement:N
state after move:[8, 0, 3, 2, 6, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 0, 3, 2, 6, 7, 1, 5, 4]



Choosing random movement:W
state after move:[0, 8, 3, 2, 6, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[0, 8, 3, 2, 6, 7, 1, 5, 4]



Choosing random movement:E
state after move:[8, 0, 3, 2, 6, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 0, 3, 2, 6, 7, 1, 5, 4]



Choosing random movement:S
state after move:[8, 6, 3, 2, 0, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 0, 7, 1, 5, 4]



Choosing random movement:E
state after move:[8, 6, 3, 2, 7, 0, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 0, 1, 5, 4]



Choosing random movement:W
state after move:[8, 6, 3, 2, 0, 7, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 0, 7, 1, 5, 4]



Choosing random movement:E
state after move:[8, 6, 3, 2, 7, 0, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 0, 1, 5, 4]
```

```
Choosing random movement:N
state after move:[8, 6, 0, 2, 7, 3, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 0, 2, 7, 3, 1, 5, 4]



Choosing random movement:S
state after move:[8, 6, 3, 2, 7, 0, 1, 5, 4]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 0, 1, 5, 4]



Choosing random movement:S
state after move:[8, 6, 3, 2, 7, 4, 1, 5, 0]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 5, 0]



Choosing random movement:W
state after move:[8, 6, 3, 2, 7, 4, 1, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 0, 5]



Choosing random movement:W
state after move:[8, 6, 3, 2, 7, 4, 0, 1, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 0, 1, 5]

Choosing random movement:E
state after move:[8, 6, 3, 2, 7, 4, 1, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 0, 5]



Choosing random movement:E
state after move:[8, 6, 3, 2, 7, 4, 1, 5, 0]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 5, 0]



Choosing random movement:W
state after move:[8, 6, 3, 2, 7, 4, 1, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 0, 5]



Choosing random movement:E
state after move:[8, 6, 3, 2, 7, 4, 1, 5, 0]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 5, 0]
```

```
Choosing random movement:W
state after move:[8, 6, 3, 2, 7, 4, 1, 0, 5]
goal:[1, 2, 3, 8, 0, 4, 7, 6, 5]
state:[8, 6, 3, 2, 7, 4, 1, 0, 5]
```

Simulated Annealing-

-> Hill climbing that does not make downhill move is incomplete.
-> Pure random walk: choosing successor from a list is complete but inefficient.
-> Solution: hill climbing + random walk =simulated annealing to ensure both efficiency and completeness.

Use a more complex Evaluation Function:
-> Do sometimes accept candidates with higher cost to escape from local optimum.
- Idea: but gradually decrease their size and frequency.

-> Adapt the parameters of this evaluation function during execution.
-> Based upon the analogy with the simulation of the annealing of solids.

Others Names-

1. Monte Carlo Annealing
2. Statistical Cooling
3. Probabilistic Hill Climbing
4. Stochastic Relaxation
5. Probabilistic Exchange Algorithm

Analogy-

->Slowly cool down a heated solid, so that all particles arrange in the ground energy state.
-> At each temperature wait until the solid reaches its thermal equilibrium.
-> Probability of being in a state with energy E :
Pr { E = E } = 1/Z(T) . exp (-E / k B .T)

E Energy
T Temperature
kB Boltzmann constant
Z(T) Normalization factor (temperature dependent)

_ *At a fixed temperature T :*
- *Perturb (randomly) the current state to a new state*
- *ΔE is the difference in energy between new and current state.*
- *If ΔE < 0 (new state is lower), accept new state as current state.*
- *If ΔE >= 0 , accept new state with probability Pr (accepted) = exp (- ΔE / k B .T).*
- *Eventually the systems evolves into thermal equilibrium*

at temperature T.

- *When equilibrium is reached, temperature T can be lowered and the process can be repeated.*

Difference between Hill Climbing and Simulated Annealing

-> *Unlike hill climbing, it does not always pick the best move.*
-> *SA picks the move randomly.*
-> *If situation improves then accept the move.*
-> *Otherwise, accept the move with some probability.*
-> *Probability decreases as the temperature goes down.*
-> *Probability decreases exponentially with the badness of the move.*
-> *Bad moves are more likely to be allowed at the start when temperature is high, and they are unlike when temperature decreases.*