CS 411 - Artificial Intelligence I
Assignment 4
Department of Computer Science, University of Illinois at Chicago

Write a program which performs iterative deepening depth first search (Figure 3.12, AIMA 4th edition) to find the solution to any given board position for 15 puzzle

Input
The input should be given in the form of a sequence of numbered tiles for initial board configuration, '0' indicating the empty space.

Output

1. Moves
2. Number of Nodes expanded
3. Time Taken
4. Memory Used

Pseudocode

Figure 3.12

```
function ITERATIVE-DEEPENING-SEARCH(problem) returns a solution node or failure
  for depth = 0 to ∞ do
    result ← DEPTH-LIMITED-SEARCH(problem, depth)
    if result ≠ cutoff then return result

function DEPTH-LIMITED-SEARCH(problem, ℓ) returns a node or failure or cutoff
  frontier ← a LIFO queue (stack) with NODE(problem.INITIAL) as an element
  result ← failure
  while not IS-EMPTY(frontier) do
    node ← POP(frontier)
    if problem.IS-GOAL(node.STATE) then return node
    if DEPTH(node) > ℓ then
      result ← cutoff
    else if not IS-CYCLE(node) do
      for each child in EXPAND(problem, node) do
        add child to frontier
  return result
```

Iterative deepening and depth-limited tree-like search. Iterative deepening repeatedly applies depth-limited search with increasing limits. It returns one of three different types of values: either a solution node; or *failure*, when it has exhausted all nodes and proved there is no solution at any depth; or *cutoff*, to mean there might be a solution at a deeper depth than ℓ. This is a tree-like search algorithm that does not keep track of *reached* states, and thus uses much less memory than best-first search, but runs the risk of visiting the same state multiple times on different paths. Also, if the IS-CYCLE check does not check *all* cycles, then the algorithm may get caught in a loop.

Submission

Please upload the code to blackboard (and to gradescope if you are using the starter code)

- Source Code
- Readme.txt including instruction to run the code, include version of compiler you are using (e.g. java 1.8, c++11)

Hint

Algorithm 3.12 mentions a function is_cycle(node) to check for cycle in the graph. You can traverse a current node all the way to the root node using pointer to the parent and keep track of seen node in a hash set. If the current node is same as one its ancestor, then cycle is detected.

Programming Language

You can choose from C++, Java, Python.

Use of starter code is optional. You can choose not to use the starter code and still complete the assignment. It will be manually graded and will take longer.

If you pass all test cases, you can expect to get a maximum of 10 points. The other 15 points are from manual grading.

Rubric

Implement Iterative deepening depth first search => 10

Print the moves to reach the solution => 3

Print number of nodes expanded => 3

Print total memory usage => 3

Print total time taken => 3

Coding style, comments, readme instruction => 3