**Program:**

import java.util.HashMap;

import java.util.LinkedList;

import java.util.Map;

public class EightPuzzleBFS {

String str = "";

String goal = "";

LinkedList <String> openList;

Map<String, Integer> levelDepth;

Map<String, String> stateHistory;

int nodes = 0;

int limit = 100;

int unique = -1;

int newValue;

int a;

String currState;

boolean solution = false;

EightPuzzleBFS(String str, String goal) {

openList = new LinkedList <String> ();

levelDepth = new HashMap<String, Integer>();

stateHistory = new HashMap<String, String>();

this.str = str;

this.goal = goal;

addToOpenList(str, null);

}

void doSearch () {

while (!openList.isEmpty()) {

currState = openList.removeFirst();

if (currState.equals(goal)) {

solution = true;

printSolution(currState);

break;

}

if (levelDepth.get(currState) == limit) {

solution = false;

printSolution(currState);

break;

}

else {

a = currState.indexOf("0");

while (a != 0 && a != 3 && a != 6) {

String nextState = currState.substring(0, a - 1) + "0" + currState.charAt(a - 1) + currState.substring(a + 1);

addToOpenList(nextState, currState);

nodes++;

break;

}

while (a != 0 && a != 1 && a != 2) {

String nextState = currState.substring(0, a - 3) + "0" + currState.substring(a - 2, a) + currState.charAt(a - 3) + currState.substring(a + 1);

addToOpenList(nextState, currState);

nodes++;

break;

}

while (a != 2 && a != 5 && a != 8) {

String nextState = currState.substring(0, a) + currState.charAt(a + 1) + "0" + currState.substring(a + 2);

addToOpenList(nextState, currState);

nodes++;

break;

}

while (a != 6 && a != 7 && a != 8) {

String nextState = currState.substring(0, a) + currState.substring(a + 3, a + 4) + currState.substring(a + 1, a + 3) + "0" + currState.substring(a + 4);

addToOpenList(nextState, currState);

nodes++;

break;

}

}

}

if (solution) {

System.out.println("Solution Exist");

} else {

System.out.println("Solution not yet found! My suggestion are:");

System.out.println("1. Try to increse level depth limit ");

System.out.println("2. Maybe it is physically impossible");

}

}

private void addToOpenList (String newState, String oldState) {

if (!levelDepth.containsKey(newState)) {

newValue = oldState == null ? 0 : levelDepth.get(oldState) + 1;

unique ++;

levelDepth.put(newState, newValue);

openList.add(newState);

stateHistory.put(newState, oldState);

}

}

void printSolution(String currState) {

if (solution) {

System.out.println("Solution found in " + levelDepth.get(currState) + " step(s)");

System.out.println("Node generated: " + nodes);

System.out.println("Unique Node generated: " + unique);

} else {

System.out.println("Solution not found!");

System.out.println("Depth Limit Reached!");

System.out.println("Node generated: " + nodes);

System.out.println("Unique Node generated: " + unique);

}

String traceState = currState;

while (traceState != null) {

System.out.println(traceState + " at " + levelDepth.get(traceState));

try {

for (int z = 0; z < 9; z++) {

System.out.print(" " + String.valueOf(traceState.charAt(z)) + " ");

if ((z + 1) % 3 == 0) {System.out.println();}

}

} catch (NullPointerException e) {}

traceState = stateHistory.get(traceState);

}

}

public static void main(String [] args) {

EightPuzzleBFS ep = new EightPuzzleBFS("283164075", "123804765");

ep.doSearch();

}

}

**Output**:

