Practical 4

Artificial Intelligence

19BCE248

D2

AIM: Program to implement A * (for 8 puzzle problem)

Code:

```
import java.util.*;
class Prac4{
	static List<Pair> lst;
	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		lst=new ArrayList<Pair>();
		initialize();
		int n=3;
		int[][] dp=new int[n][n];
		for (int i=0;i<n;i++){
			for (int j=0;j<n;j++) {
				dp[i][j]=sc.nextInt();
			}
		}
		PriorityQueue<State> q=new PriorityQueue<>((s1,s2)->s1.val-s2.val);
		q.add(new State(dp,getHeuristic(dp)));
		Set<String> set=new HashSet<>();
		int level=0;
		set.add(getString(dp));
		outer:while (!q.isEmpty()) {
```

```
        // int sz=q.size();
        // while (sz-->0) {
         State st=q.poll();
         int[][] state=st.dp;
         for (int i=0;i<3;i++) {
           for (int j=0;j<3;j++) {
             System.out.print(state[i][j]+" ");
           }
           System.out.println("");
         }

if (isDone(state)) {
   System.out.println("Final State");
   for (int i=0;i<3;i++) {
     for (int j=0;j<3;j++) {
       System.out.print(state[i][j]+" ");
     }
     System.out.println("");
   }
  break outer;
}
for (int i=0;i<3;i++) {
  for (int j=0;j<3;j++) {
    if (state[i][j]==0) {
      if (i-1>=0) {
        int up=state[i-1][j];
        state[i][j]=up;
        state[i-1][j]=0;
```

```
                if (!set.contains(getString(state))) {
                int[][] copy =
Arrays.stream(state).map(int[]::clone).toArray(int[][]::new);
                q.add(new State(copy,getHeuristic(copy)+level));
                set.add(getString(copy));
                }
                state[i-1][j]=up;
                state[i][j]=0;
               }
               if (i+1<3) {
                int down=state[i+1][j];
                state[i][j]=down;
                state[i+1][j]=0;
                if (!set.contains(getString(state))) {
                int[][] copy =
Arrays.stream(state).map(int[]::clone).toArray(int[][]::new);
                q.add(new State(copy,getHeuristic(copy)+level));
                set.add(getString(copy));
                }
                state[i+1][j]=down;
                state[i][j]=0;
               }
               if (j-1>=0) {
                int left=state[i][j-1];
                state[i][j]=left;
                state[i][j-1]=0;
                if (!set.contains(getString(state))) {
                int[][] copy =
Arrays.stream(state).map(int[]::clone).toArray(int[][]::new);
```

```
                    q.add(new State(copy,getHeuristic(copy)+level));
                    set.add(getString(copy));
                    }
                    state[i][j-1]=left;
                    state[i][j]=0;
                  }
                  if (j+1<3) {
                    int right=state[i][j+1];
                    state[i][j]=right;
                    state[i][j+1]=0;
                    if (!set.contains(getString(state))) {
                    int[][] copy =
Arrays.stream(state).map(int[]::clone).toArray(int[][]::new);
                    q.add(new State(copy,getHeuristic(copy)+level));
                    set.add(getString(copy));
                    }
                    state[i][j+1]=right;
                    state[i][j]=0;
                   }
                 }


               }
              }
             }
            // }
            level++;
           }
           public static boolean isDone(int[][] dp){
```

```
 ArrayList<Integer> check=new ArrayList<>();
 for (int i=0;i<3;i++) {
  for (int j=0;j<3;j++) {
   check.add(dp[i][j]);
  }
 }

 if (check.get(0)==0) {
  for (int i=1;i<=8;i++) {
   if (check.get(i)!=i) {
    return false;
   }
  }
  return true;
 }
 if (check.get(8)==0) {
  for (int i=0;i<8;i++) {
   if (check.get(i)!=i+1) {
    return false;
   }
  }
  return true;
 }
 return false;
}

        public static void initialize(){
                for (int i=0;i<3;i++) {
```

```
                for(int j=0;j<3;j++){
                        lst.add(new Pair(i,j));
                }
        }
}
public static int getHeuristic(int[][] dp){
        int h=0;
        for (int i=0;i<3;i++) {
                for (int j=0;j<3;j++) {
                        if (dp[i][j]==0) {
                                continue;
                        }
                        int x=lst.get(dp[i][j]-1).x;
                        int y=lst.get(dp[i][j]-1).y;
                        h+=Math.abs(i-x)+Math.abs(i-y);
                }
        }
        return h;
}
public static String getString(int[][] dp){
String str="";
for (int i=0;i<3;i++) {
for (int j=0;j<3;j++) {
  str+=dp[i][j]+"";
}
}
return str;
}
```

```
}
class Pair{
	int x,y;
	Pair(int x,int y){
		this.x=x;
		this.y=y;
	}
}
class State{
	int[][] dp;
	int val;
	State(int[][] dp,int val){
		this.val=val;
		this.dp=dp;
	}
}
```

Output: