AI

EL3

Practical 2

19BCE248

AIM: To solve 8 puzzle problem using dfs/bfs without using recursion

Code:

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.*;
import java.lang.*;
public class Solution{
static FastScanner sc = new FastScanner();
static StringBuffer as;
public static void solve(){
 int[][] dp=sc.read2dArray(3,3);
 Queue<int[][]> q=new LinkedList<>();
 Set<String> set=new HashSet<>();
 q.add(dp);
 set.add(getString(dp));
 outer:while (!q.isEmpty()) {
   int sz=q.size();
   while (sz-->0) {
     int[][] state=q.poll();
     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(copy);
             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(copy);

              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(copy);

              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(copy);
```

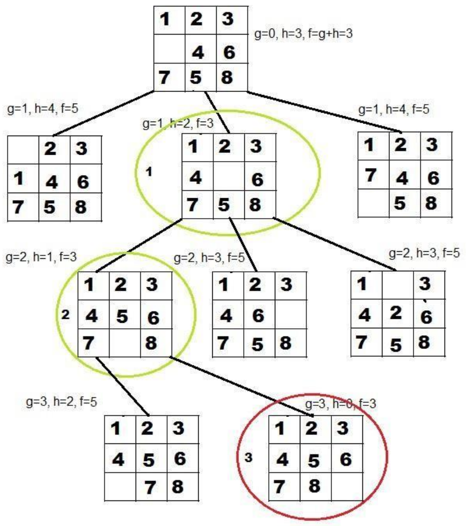
```
             set.add(getString(copy));
             }
             state[i][j+1]=right;
             state[i][j]=0;
           }
         }

       }
     }
   }
 }
}
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;
}
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++) {
```

```java
      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 main(String[] args) {
  solve();
}

static class FastScanner {
  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
  StringTokenizer st = new StringTokenizer("");
```

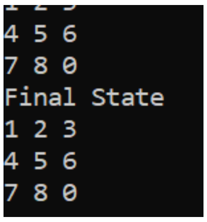
```java
String next() {
  while (!st.hasMoreTokens())
    try {
      st = new StringTokenizer(br.readLine());
    } catch (IOException e) {
      e.printStackTrace();
    }
  return st.nextToken();
}

int nextInt() {
  return Integer.parseInt(next());
}

int[] readArray(int n) {
  int[] a = new int[n];
  for (int i = 0; i < n; i++)
    a[i] = nextInt();
  return a;
}

long[] readLongArray(int n) {
  long[] a = new long[n];
  for (int i = 0; i < n; i++)
    a[i] = nextLong();
  return a;
}

int[][] read2dArray(int n, int m) {
  int arr[][] = new int[n][m];
```

```java
    for (int i = 0; i < n; i++) {
      for (int j = 0; j < m; j++) {
        arr[i][j] = nextInt();
      }
    }
    return arr;
  }

  ArrayList<Integer> readArrayList(int n) {
    ArrayList<Integer> arr = new ArrayList<Integer>();
    for (int i = 0; i < n; i++) {
      int a = nextInt();
      arr.add(a);
    }
    return arr;
  }

  long nextLong() {
    return Long.parseLong(next());
  }
}
```

State Space for Sample Input:



Output:



```
4 5 6
7 8 0
Final State
1 2 3
4 5 6
7 8 0
```