ARTIFICIAL INTELLIGENCE

ROLL NO. : 18BCE191
DATE: 18/08/2021

Aim

Solve 8-puzzle problem using BFS

Code

```
#include<bits/stdc++.h>
#define FAST ios_base::sync_with_stdio(false);cin.tie();cout.tie();
#define FILE_READ_IN freopen("input2.txt","r",stdin);
#define FILE_READ_OUT freopen("output.txt","w",stdout);
using namespace std;
typedef long long ll;

int dx[4] = {1,0,-1,0};
int dy[4] = {0,1,0,-1};

bool isSolvable(vector<vector<int>>& a){
   int a_flat[9];
   for(int i=0;i<3;i++) for(int j=0;j<3;j++) a_flat[i*3+j]=a[i][j];
   int inversion=0;
   for(int i=0;i<8;i++){
       for(int j=i+1;j<9;j++){
           if(a_flat[i] && a_flat[j] && a_flat[i] > a_flat[j]){
               inversion++;
           }
       }
   }
   return inversion%2==0;
}
class State{
   private:
```

```
int state_id;
int x,y;
vector<vector<int>> a;
public:
State(vector<vector<int>> a, int state_id,int x,int  y){
    this->state_id = state_id;
    this->a = a;
    this->x = x;
    this->y = y;
}
int get_x(){return x;}
int get_y(){return y;}

int get_id(){
    return state_id;
}
vector<vector<int>> get_a(){
    return a;
}
bool isGoalState(){
    for(int i=0;i<a.size();i++){
        for(int j=0;j<a[i].size();j++){
            if(a[i][j] == 0) continue;
            if(a[i][j] == i*a.size()+j+1){
                continue;
            }
            else return 0;
        }
    }
    return 1;
}
void print(){
    cout<<"state_id: "<<state_id<<"\n";
    for(int i=0;i<a.size();i++){
        for(int j=0;j<a[i].size();j++){
            cout<<a[i][j]<<" ";
        }
        cout<<"\n";
```

```
        }
        cout<<"\n";
    }
};
void solve(vector<vector<int>>& a,int x,int y){
    State start(a,0,x,y);

    queue<State> q;
    q.push(start);
    int id=1;

    set<vector<vector<int>>> visited;
    visited.insert(start.get_a());

    while(!q.empty()){
        State s = q.front();
        q.pop();
        if(s.isGoalState()){
            s.print();
            return;
        }
        for(int i=0;i<4;i++){
            int nx = s.get_x()+dx[i];
            int ny = s.get_y()+dy[i];
            if(nx<0 || nx>=3 || ny<0 || ny>=3){
                continue;
            }
            vector<vector<int>> n_a = s.get_a();
            swap(n_a[nx][ny],n_a[s.get_x()][s.get_y()]);

            if(visited.count(n_a)) continue;
            State new_state(n_a,++id,nx,ny);
            q.push(new_state);
            visited.insert(n_a);

        }
    }
}
```

```cpp
int main(){
  #ifndef ONLINE_JUDGE
     FILE_READ_IN
     FILE_READ_OUT
  #endif
  int n=3;
  vector<vector<int>> a(n,vector<int>(n));
  int x=-1,y=-1;
  for(int i=0;i<n;i++){
      for(int j=0;j<n;j++){
          cin>>a[i][j];
          if(a[i][j] == 0){
              x=i,y=j;
          }
      }
  }
  if(!isSolvable(a)){
      cout<<"Not solvable\n";
  }
  else{
      solve(a,x,y);
  }

  return 0;
}
```

Input	Output
7 1 2 3 4 5 6 0 8	state_id: 101626 1 2 3 4 5 6 7 8 0

0 1 2 3 4 5 6 7 8	state_id: 83617 1 2 3 4 5 6 7 8 0
1 8 2 0 4 3 7 6 5	state_id: 412 1 2 3 4 5 6 7 8 0
1 2 3 4 5 6 8 7 0	Not solvable

** Note: state_id refers to the number of states that are explored to reach the goal state from the initial state.