Assignment for Midterm

CSE-0408 Summer 2021

Ashraf uddin Mamun - [UG02-44-17-044]

Department of Computer Science and Engineering
State University of Bangladesh (SUB)
Dhaka, Bangladesh
mamun336929@gmail.com

Assignment 01: 8 puzzle problems in C++

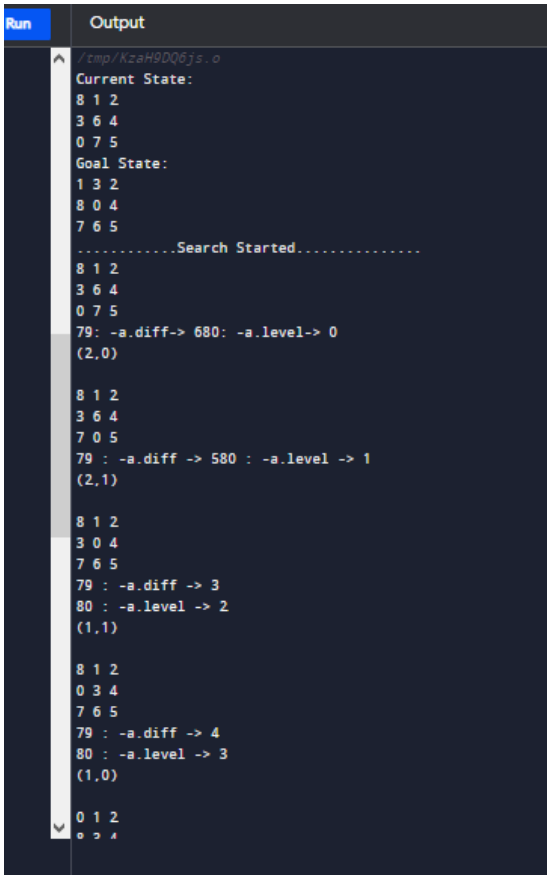
```
#include<bits/stdc++.h>
using namespace std;
#define D(x) cerr<<__LINE__<<" : "<<#x<<" -> "<<x<<endl
#define rep(i,j) for(int i = 0; i < 3; i++) for(int j = 0; j < 3; j++)
#define PII pair < int, int >
typedef vector<vector<int>> vec2D;

const int MAX = 1e5+7;
int t=1, n, m, l, k, tc;

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

vec2D init{
    {8, 1, 2},
    {3, 6, 4},
    {0, 7, 5}
};
vec2D goal{
```

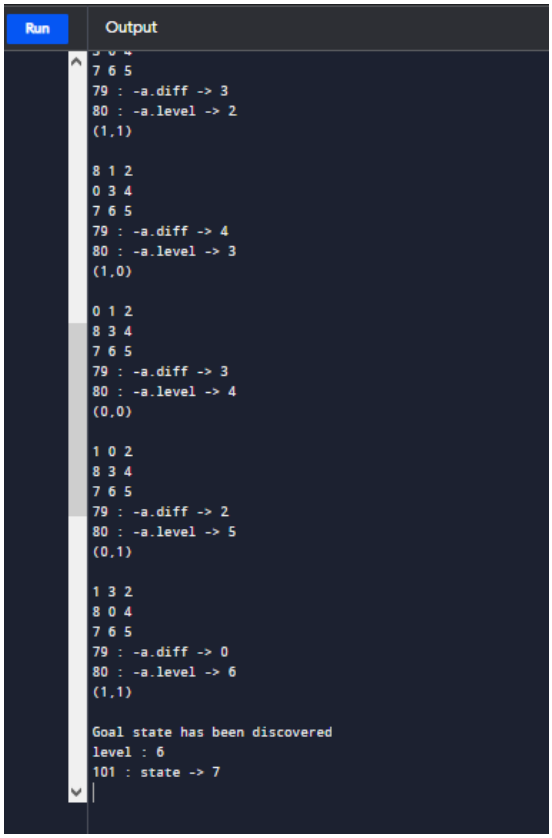
```
    {1, 3, 2},
    {8, 0, 4},
    {7, 6, 5}
};
//vec2D init{
//    {1, 2, 3},
//    {8, 6, 0},
//    {7, 5, 4}
//};
//vec2D goal{
//    {1, 2, 3},
//    {8, 0, 4},
//    {7, 6, 5}
//};
//vec2D init{
//    {1, 3, 2},
//    {4, 0, 7},
//    {6, 5, 8}
//};
//vec2D goal{
//    {0, 2, 4},
//    {1, 3, 8},
//    {6, 5, 7}
//};

struct Box {
    vec2D mat{ { 0,0,0 },{ 0,0,0},{ 0,0,0} };
    int diff, level;
    int x, y;
    int lastx, lasty;
```

```
    Box(vec2D a,int b = 0, int c = 0, PII p = {0,0}, PII q =
{0,0}) {
        rep(i,j) mat[i][j] = a[i][j];
        diff = b;
        level = c;
        x = p.first;
        y = p.second;
        lastx = q.first;
        lasty = q.second;
    }
};

bool operator < (Box A, Box B) {
    if(A.diff == B.diff) return A.level < B.level;
    return A.diff < B.diff;
}

int isEqual(vec2D a, vec2D b) {
    int ret(0);
    rep(i,j) if (a[i][j] != b[i][j]) ret--;
    return ret;
}

bool check(int i, int j) {
    return i>=0 and i<3 and j>=0 and j<3;
}
void print(Box a) {
    rep(i,j)
    cout << a.mat[i][j] << (j == 2 ? "\n" : " ");
    D(-a.diff);
    D(-a.level);
```

```
    cout << "(" << a.x << "," << a.y <<")\n\n";
}

void dijkstra(int x, int y) {
    map < vec2D,  bool > mp;
    priority_queue < Box > PQ;
    int nD = isEqual(init, goal);
    Box src = {init, nD, 0, {x,y}, {-1,-1}};
    PQ.push(src);
    int state = 0;
    while(!PQ.empty()) {
        state++;
        Box now = PQ.top();
        PQ.pop();
        print(now);
        if(!now.diff) {
            puts("Goal state has been discovered");
            cout << "level : " << -now.level << "\n";
            D(state);
            break;
        }
        if(mp[now.mat]) continue;
        mp[now.mat] = true;
        for(int i = 0; i < 4; i++) {
            int xx = now.x + dx[i];
            int yy = now.y + dy[i];
            if(check(xx, yy)) {
                if(now.lastx == xx and now.lasty == yy)
continue;
                Box temp = now;
```

```
                swap(temp.mat[temp.x][temp.y],
temp.mat[xx][yy]);
                temp.diff = isEqual(temp.mat, goal);
                temp.level = now.level - 1;
                temp.x = xx;
                temp.y = yy;
                temp.lastx = now.x;
                temp.lasty = now.y;
                PQ.push(temp);
            }
        }
    }
}


signed main() {
    puts("Current State:");
    rep(i,j) cout << init[i][j] << (j == 2 ? "\n" : " ");
    puts("");
    puts("Goal State:");
    rep(i,j) cout << goal[i][j] << (j == 2 ? "\n" : " ");
    puts("\n...........Search Started.............\n");
    rep(i,j) if(!init[i][j]) dijkstra(i,j);
    return 0;
}
```


Run
Output
/tmp/KzaH9DQ6js.o
Current State:
8 1 2
3 6 4
0 7 5
Goal State:
1 3 2
8 0 4
7 6 5
.....Search Started.....
8 1 2
3 6 4
0 7 5
79: -a.diff-> 680: -a.level-> 0
(2,0)
8 1 2
3 6 4
7 0 5
79 : -a.diff -> 580 : -a.level -> 1
(2,1)
8 1 2
3 0 4
7 6 5
79 : -a.diff -> 3
80 : -a.level -> 2
(1,1)
8 1 2
0 3 4
7 6 5
79 : -a.diff -> 4
80 : -a.level -> 3
(1,0)
0 1 2


Run
Output
7 6 5
79 : -a.diff -> 3
80 : -a.level -> 2
(1,1)
8 1 2
0 3 4
7 6 5
79 : -a.diff -> 4
80 : -a.level -> 3
(1,0)
0 1 2
8 3 4
7 6 5
79 : -a.diff -> 3
80 : -a.level -> 4
(0,0)
1 0 2
8 3 4
7 6 5
79 : -a.diff -> 2
80 : -a.level -> 5
(0,1)
1 3 2
8 0 4
7 6 5
79 : -a.diff -> 0
80 : -a.level -> 6
(1,1)
Goal state has been discovered
level : 6
101 : state -> 7

Assignment 02: BFS code in Python

```
from queue import PriorityQueue
v = 14
graph = [[] for i in range(v)]


def best_first_search(source, target, n):
     visited = [0] * n
     visited = True
     pq = PriorityQueue()
     pq.put((0, source))
     while pq.empty() == False:
          u = pq.get()[1]
          print(u, end=" ")
          if u == target:
               break

          for v, c in graph[u]:
               if visited[v] == False:
                    visited[v] = True
                    pq.put((c, v))
     print()


def addedge(x, y, cost):
     graph[x].append((y, cost))
     graph[y].append((x, cost))

addedge(0, 1, 3)
```

```
addedge(0, 2, 6)
addedge(0, 3, 5)
addedge(1, 4, 9)
addedge(1, 5, 8)
addedge(2, 6, 12)
addedge(2, 7, 14)
addedge(3, 8, 7)
addedge(8, 9, 5)
addedge(8, 10, 6)
addedge(9, 11, 1)
addedge(9, 12, 10)
addedge(9, 13, 2)

source = 0
target = 9
best_first_search(source, target, v)
```