八数码

描述：

在九宫格里放在1到8共8个数字还有一个是空格，与空格相邻的数字可以移动到空格的位置，问给定的状态最少需要几步能到达目标状态（用0表示空格）：

1 2 3

4 5 6

7 8 0

输入：

输入一个给定的状态。

输出：

输出到达目标状态的最小步数。不能到达时输出-1。

输入样例：

1 2 3

4 0 6

7 5 8

输出样例：

2

#include<iostream>

#include<queue>

#include<map>

using namespace std;

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

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

queue<int> q1;

map<int,int> step;

void init(int m);

int readdate();

int bfs();

int canmove(int u,int dire);

int move(int u,int dire);

int main()

{

int s,num;

s=readdate();

init(s);

num=bfs();

cout<<num<<endl;

return 0;

}

int readdate()

{

int a[3][3];

int i,j,sum;

sum = 0;

for(i = 0;i < 3;i++)

{

for(j = 0;j < 3;j++)

{

cin >> a[i][j];

sum = sum \* 10 + a[i][j];

}

}

return (sum);

}

void init(int m)

{

q1.push(m);

step[m] = 0;

}

int bfs()

{

int u,v,i;

while(!q1.empty())

{

u = q1.front();

q1.pop();

for(i = 0;i < 4;i++)

{

if(canmove(u,i))

{

v = move(u,i);

if(v == 123456780)

{

return(step[u] + 1);

}

if(step.count(v) == 0)

{

q1.push(v);

step[v] = step[u] + 1;

}

}

}

}

return(-1);

}

int canmove(int u,int dire)

{

int Map[3][3];

int i,j;

int row,col;

int r,c;

int v;

v = u;

for(i = 2;i >= 0;i--)

{

for(j = 2;j >= 0;j--)

{

Map[i][j] = v%10;

v = v/10;

if(Map[i][j] == 0)

{

row = i;

col = j;

}

}

}

r = row + dr[dire];

c = col + dc[dire];

if(r >= 0 && r < 3 && c >= 0 && c < 3)

{

return 1;

}

else

{

return 0;

}

}

int move(int u,int dire)

{

int none[3][3];

int i,j;

int row,col;

int r,c;

int sum;

for(i = 2;i >= 0;i--)

{

for(j = 2;j >= 0;j--)

{

none[i][j] = u%10;

u = u/10;

if(none[i][j] == 0)

{

row = i;

col = j;

}

}

}

r = row + dr[dire];

c = col + dc[dire];

none[row][col] = none[r][c];

none[r][c] = 0;

sum = 0;

for(i = 0;i < 3;i++)

{

for(j = 0;j < 3;j++)

{

sum = sum\*10 + none[i][j];

}

}

return (sum);

}