WEEK7 八数码问题

郁思敏 10153901225

1. **main.cpp**

#include <iostream>

#include <cstring>

#include <set>

#include <vector>

#include "klotski.h"

using namespace std;

int main()

{

//start:264137058

//end: 815736402

klotski test;

if(test.Judge())

test.BFS();

else

cout << "it cannot be changed into end status." << endl;

return 0;

}

1. **klotski.h**

#ifndef KLOTSKI\_H

#define KLOTSKI\_H

#include <iostream>

#include <set>

#include <vector>

#include <cstring>

using namespace std;

typedef struct node

{

string str;

struct node \*pre;

int steps;

}Node;

class klotski

{

public:

klotski(); //initialize start status.

virtual ~klotski();

void PrintString(string str); //print the store[index]->string.

bool Judge(); //judge whether this change can be realized.

void BFS();

bool Existed(string str); //judge whether it is already in exist.

int FindZero(int index); //return the index of zero in string.

private:

vector<Node \*> store; //store status.

set<string> exist; //judge whether has been used.

string end\_; //end string.

int cur; //current index;

};

#endif // KLOTSKI\_H

1. **klotski.cpp**

#include "klotski.h"

klotski::klotski() //initialize start status.

{

Node \*head = new Node;

if(head != NULL)

{

cout << "please enter your start status:" << endl;

cin >> head->str;

head->pre = NULL;

head->steps = 0;

exist.insert(head->str);

store.push\_back(head);

}

cout << "please enter your end status:" << endl;

cin >> end\_;

cur = 0;

}

void klotski::PrintString(string str) //print the store[index]->string.

{

int i;

for(i=0; i<9; i++)

{

cout << str[i] << " ";

if((i+1)%3 == 0)

cout << endl;

}

}

bool klotski::Existed(string str) //judge whether it is already in exist.

{

unsigned size\_ = exist.size();

exist.insert(str);

if(size\_+1 == exist.size())

return false;

return true;

}

int InvertPair(string str)

{

int i, j, val=0;

for(i=0; i<8; i++)

{

for(j=i; j<9; j++)

{

if(str[i]>str[j])

val++;

}

}

return val;

}

bool klotski::Judge() //judge whether this change can be realized.

{

int invert1 = InvertPair(store[0]->str);

int invert2 = InvertPair(end\_);

if(invert1%2 == invert2%2)

return false;

return true;

}

int klotski::FindZero(int index) //return the index of zero in string.

{

int val = -1;

string temp = store[index]->str;

for(int i=0; i<9; i++)

{

if(temp[i] == '0')

{

val = i;

break;

}

}

return val;

}

void klotski::BFS()

{

bool flag = true;

while(flag)

{

if(end\_ == store[cur]->str)

{

Node \*\_end = store[cur];

vector<Node \*> print;

while(\_end != NULL)

{

print.push\_back(\_end);

\_end = \_end->pre;

}

cout << "following is the path:" << endl;

for(int i=(int)print.size()-1; i>=0; i--)

{

cout << "step " << print[i]->steps << ":" << endl;

PrintString(print[i]->str);

cout << endl;

}

cout << "overall steps:" << store[cur]->steps << endl;

flag = false;

}

else

{

int zero\_index = FindZero(cur);

int dir[4] = {zero\_index-3, zero\_index+3, zero\_index-1, zero\_index+1}; //up, down, left, right

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

{

if(dir[i]<9 && dir[i]>=0)

{

if(zero\_index%3==0 && (dir[i]+1)%3==0)

{

continue;

}

else if(zero\_index%3==2 && (dir[i]-1)%3==2)

{

continue;

}

else

{

Node \*node = new Node;

if(node != NULL)

{

node->str = store[cur]->str;

node->str[zero\_index] = node->str[dir[i]];

node->str[dir[i]] = '0';

node->pre = store[cur];

node->steps = store[cur]->steps+1;

if(!Existed(node->str))

store.push\_back(node);

}

}

}

}

cur++;

}

}

}

klotski::~klotski()

{

}