

#include <iostream>

#include <vector>

#include <algorithm>

#include <random>

#include <ctime>

#include <cstdlib>

using namespace std;

const int N = 8; // Размер шахматной доски

typedef vector<pair<int, int>> pole;

pole hodi = { {1,-2},{-1,-2},{2,-1},{2,1},{1,2},{-1,2},{-2,-1},{-2,1} };

int VSE\_HODI = 0;

pole rand\_vec() {

srand(time(nullptr));

pole a, hodi\_ = hodi;

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

a.push\_back(hodi\_[rand()%hodi\_.size()]);

auto it = std::remove(hodi\_.begin(), hodi\_.end(), a.back());

hodi\_.erase(it, hodi\_.end());

}

for (const auto& a\_ : a) {

//cout << a\_.first << " " << a\_.second << endl;

}

//cin.get();

return a;

}

int vec\_sum(vector<vector<int>> &a) {

int k = 0;

for (const auto& a\_ : a) {

for (const auto& b\_ : a\_) {

k += b\_;

}

}

return k;

}

void print\_doska(vector<vector<int>> a, int pos\_y = -1, int pos\_x = -1) {

int y = 0;

for (const auto& a\_ : a) {

int x = 0;

for (const auto& b\_ : a\_) {

if (y == pos\_y && x == pos\_x) {

cout << "\x1b[42m" << b\_ << "\x1b[40m ";

}

else cout << b\_ << " ";

x++;

}

cout << endl;

y++;

}

cout << vec\_sum(a)<<endl;

}

void clear\_doska(vector<vector<int>> &a, int pos\_y=0, int pos\_x=0) {

int y = 0;

for (auto& a\_ : a) {

int x = 0;

for (auto& b : a\_) {

if (x == pos\_x && y == pos\_y) {

b = 1;

}

else b = 0;

x++;

}

y++;

}

}

void print\_path(vector<vector<pair<int,int>>> in) {

for (const auto& a : in) {

for (const auto& b : a) {

printf("%d\t%d\n", b.first,b.second);

}

cout << endl;

}

cout << endl;

}

int path\_exist(vector<vector<pair<int, int>>> &in, vector<pair<int,int>> &in\_) {

int k = 0;

for (auto& a\_ : in) {

if (a\_ == in\_) {

k++;

}

}

if (k == 0)

return 1;

return 0;

}

void doska\_resh(vector<vector<int>> v, int nachY, int nachX, int konY, int konX) {

int cant\_go = 0; int TOTAL\_ERRORS = 0;

vector<pole> all\_path; pole path;

for (int y = nachY; y < v.size(); y++) {

for (int x = nachX; x < v[y].size(); x++) {

while (TOTAL\_ERRORS < 100000) {

for (const auto& a : rand\_vec()) {

if (y + a.first >= 0 && x + a.second >= 0 && y + a.first < 8 && x + a.second < 8 && v[y + a.first][x + a.second] != 1) {

cant\_go = 0;

y += a.first; x += a.second;

v[y][x] = 1; VSE\_HODI++;

path.push\_back(a);

if (y == konY && x == konX) {

if (all\_path.size() == 0) {

all\_path.push\_back(path);

path.clear();

}

else {

if (path\_exist(all\_path, path) == 1) {

all\_path.push\_back(path);

print\_doska(v, y, x);

printf("Размер path: %d\n", path.size());

path.clear();

TOTAL\_ERRORS = 0;

}

}

}

}

else {

cant\_go++;

if (cant\_go > 17) {

path.clear();

clear\_doska(v, nachY, nachX);

cant\_go = 0;

TOTAL\_ERRORS++;

break;

}

}

}

}

}

}

cout << endl << all\_path.size() << endl;

}

int main() {

//pole hodi = { {-2,1},{-2,-1},{-1,2},{1,2},{2,1},{2,-1},{-1,-2},{1,-2} };

setlocale(0, "");

printf("Введите две пары чисел: {начY,начX} {конY,конX}: \n");

int nachY, nachX, konY, konX;

nachY = 3; nachX = 4; konY = 7; konX = 0;

//cin >> nachY >> nachX >> konY >> konX; //не удалять

vector<vector<int>> DOSKA(8, {0,0,0,0,0,0,0,0});

pair<int, int> kin\_pos = {nachY,nachX};

DOSKA[kin\_pos.first][kin\_pos.second] = 1;

print\_doska(DOSKA);

cout <<endl;

doska\_resh(DOSKA, nachY, nachX, konY, konX);

}









