1. Дана шахматная доска размером 8\*8 и шахматный конь. Программа должна запросить у пользователя координаты клетки поля и поставить туда коня. Задача программы найти и вывести путь коня, при котором он обойдет все клетки доски, становясь в каждую клетку только один раз.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-

#include <iostream>

#include <ctime>

using namespace std;

const int RD = 8;

int desk[RD][RD];

int step[8][2] = { {1,-2},{2,-1},{2,1},{1,2},{-1,2},{-2,1},{-2,-1},{-1,-2} };

int n;

void print\_desk();

bool set\_horse(int x, int y);

int main()

{

system("chcp 1251");

time\_t t\_start, t\_end;

time(&t\_start);

n = 0;

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

for (int j = 0;j < RD;j++)

desk[i][j] = 0;

int a, b;

cout << "Введите первую цифру координат: " << endl;

cin >> a;

cout << "Введите вторую цифру координат: " << endl;

cin >> b;

set\_horse(a, b);

print\_desk();

time(&t\_end);

cout << "Решение найдено за: " << t\_end - t\_start << "секунд" << endl;

return 0;

}

void print\_desk()

{

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

{

for (int j = 0;j < RD;j++)

cout << desk[i][j] << "\t";

cout << "\n\n\n";

}

}

bool set\_horse(int x, int y)

{

// проверка параметров

if ((x < 0) || (x >= RD) || (y < 0) || (y >= RD))

return false;

if (desk[x][y] != 0)

return false;

// слледующий ход

n++;

desk[x][y] = n;

//стоп игра

if (n == RD \* RD)

return true;

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

if (set\_horse(x + step[i][0], y + step[i][1]))

return true;

n--;

desk[x][y] = 0;

return false;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-

2 Написать игру пятнашки

#include <iostream>

#include <vector>

#include <cstdlib>

#include <iomanip>

#include <conio.h>

using namespace std;

const unsigned short SIZE = 4; // размер игрового поля

vector<int> in\_game\_map(SIZE);

vector<vector<int>> game\_map(SIZE, in\_game\_map); // игровая карта

vector<int> in\_right\_map(SIZE);

vector<vector<int>> right\_map(SIZE, in\_right\_map); // правильная итоговая карта

struct coordinate // хранилище координат нулевого элемента

{

unsigned x;

unsigned y;

} zero; // объект

void create\_right\_map();

void create\_game\_map();

bool check\_map();

void up\_move();

void down\_move();

void right\_move();

void right\_move();

void get\_direction();

void screen();

int main()

{

system("chcp 1251");

srand(static\_cast<int>(time(0)));

create\_right\_map(); // создание игровых карт

do

{

create\_game\_map();

} while (check\_map());

do // игровой цикл

{

screen();

get\_direction();

} while (!check\_map());

cout << " Ты выиграл! Конец игры! ";

//\_getch();

return 0;

}

void create\_right\_map() // создаем правильную карту заполненую по порядку

{

unsigned right\_value = 1;

for (unsigned i = 0; i < SIZE; i++)

{

for (unsigned j = 0; j < SIZE; j++)

right\_map[i][j] = right\_value++;

}

right\_map[SIZE - 1][SIZE - 1] = 0; // нулевой элемент в нижний правый угол

}

void create\_game\_map() // рандомно создаем игровую карту

{

unsigned limit = SIZE \* SIZE;

vector<int> temporary; // временный массив из которого будем брать значения в игровую карту

for (unsigned i = 0; i < limit; i++)

temporary.push\_back(i);

int value;

for (unsigned i = 0; i < SIZE; i++)

{

for (unsigned j = 0; j < SIZE; j++)

{

value = rand() % limit--;

game\_map[i][j] = temporary[value];

if (temporary[value] == 0) // сохраняем координаты нулевого элемента

{

zero.x = j;

zero.y = i;

}

temporary.erase(temporary.begin() + value);

}

}

}

bool check\_map() // сравнение игровой и правильной карты для определения конца игры

{

if (game\_map == right\_map)

return true;

return false;

}

void up\_move() // ход вверх (нулевой элемент вниз)

{

if (zero.y < SIZE - 1)

{

game\_map[zero.y][zero.x] = game\_map[zero.y + 1][zero.x];

zero.y++;

game\_map[zero.y][zero.x] = 0;

}

}

void down\_move() // ход вниз (нулевой элемент вверх)

{

if (zero.y > 0)

{

game\_map[zero.y][zero.x] = game\_map[zero.y - 1][zero.x];

zero.y--;

game\_map[zero.y][zero.x] = 0;

}

}

void right\_move() // ход вправо (нулевой элемент влево)

{

if (zero.x > 0)

{

game\_map[zero.y][zero.x] = game\_map[zero.y][zero.x - 1];

zero.x--;

game\_map[zero.y][zero.x] = 0;

}

}

void left\_move() // ход влево (нулевой элемент вправо)

{

if (zero.x < SIZE - 1)

{

game\_map[zero.y][zero.x] = game\_map[zero.y][zero.x + 1];

zero.x++;

game\_map[zero.y][zero.x] = 0;

}

}

void get\_direction() // определяем нажатую игроком стрелку

{

int move = static\_cast<int> (\_getch()); // UP = 72, DOWN = 80, RIGHT = 77, LEFT = 75

switch (move)

{

case 72:

{

up\_move(); break;

}

case 80:

{

down\_move(); break;

}

case 77:

{

right\_move(); break;

}

case 75:

{

left\_move(); break;

}

default:

{

get\_direction();

}

}

}

void screen() // выводим массив на экран

{

system("cls");

for (unsigned i = 0; i < SIZE; i++)

{

for (unsigned j = 0; j < SIZE; j++)

{

if (game\_map[i][j] != 0)

cout << setw(2) << setfill('0') << game\_map[i][j] << ' ';

else

cout << "## "; // нулевой элемент

}

cout << '\n';

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-

3 Написать игру крестики нолики

#include <iostream>

#include <cmath>

#include <cstdlib>

using namespace std;

int main()

{

system("chcp 1251");

const int N = 3;

int x, y, k, dgt;

char A[N][N], c, z;

string s, a, b, name1, name2;

a = "Нолики";

b = "Крестики";

z = '0';

c = 'X';

cout << "Игра крестики-нолики" << endl;

cout << "Введите имя первого игрока: ";

cin>>name1;

cout << "Введите имя второго игрока: ";

cin>> name2;

cout << "За кого ходит первый игрок?(введите цифру):" << endl;

cout << "0: 1" << endl;

cout << "X: 2" << endl;

cout << "Цифра: ";

cin >> dgt;

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

{

for (int j = 0; j < N; j++)

{

A[i][j] = '?';

cout << A[i][j] << " ";

}

cout << endl;

}

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

{

if ((!(i % 2)) && (dgt == 1)) s = a;

if ((!(i % 2)) && (dgt == 2)) s = b;

if ((i % 2) && (dgt == 1)) s = b;

if ((i % 2) && (dgt == 2)) s = a;

mylabel:

cout << "Сделайте ваш ход (" << s << ")" << endl;

cout << "Строка: ";

cin >> x;

cout << "Столбец: ";

cin >> y;

x--;

y--;

if (A[x][y] != '?')

{

system("cls");

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

{

for (int j = 0; j < N; j++)

{

cout << A[i][j] << " ";

}

cout << endl;

}

cout << "Данная позиция занята!" << endl;

goto mylabel;

}

if ((!(i % 2)) && (dgt == 1)) A[x][y] = z;

if ((!(i % 2)) && (dgt == 2)) A[x][y] = c;

if ((i % 2) && (dgt == 1)) A[x][y] = c;

if ((i % 2) && (dgt == 2)) A[x][y] = z;

system("cls");

k = 0;

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

{

for (int j = 0; j < N; j++)

{

cout << A[i][j] << " ";

}

cout << endl;

}

if (((A[0][0] == '0') && (A[0][1] == '0') && (A[0][2] == '0')) ||

((A[1][0] == '0') && (A[1][1] == '0') && (A[1][2] == '0')) ||

((A[2][0] == '0') && (A[2][1] == '0') && (A[2][2] == '0')) ||

((A[0][0] == '0') && (A[1][0] == '0') && (A[2][0] == '0')) ||

((A[0][1] == '0') && (A[1][1] == '0') && (A[2][1] == '0')) ||

((A[0][2] == '0') && (A[1][2] == '0') && (A[2][2] == '0')) ||

((A[0][0] == '0') && (A[1][1] == '0') && (A[2][2] == '0')) ||

((A[2][0] == '0') && (A[1][1] == '0') && (A[0][2] == '0')) ||

((A[0][0] == 'X') && (A[0][1] == 'X') && (A[0][2] == 'X')) ||

((A[1][0] == 'X') && (A[1][1] == 'X') && (A[1][2] == 'X')) ||

((A[2][0] == 'X') && (A[2][1] == 'X') && (A[2][2] == 'X')) ||

((A[0][0] == 'X') && (A[1][0] == 'X') && (A[2][0] == 'X')) ||

((A[0][1] == 'X') && (A[1][1] == 'X') && (A[2][1] == 'X')) ||

((A[0][2] == 'X') && (A[1][2] == 'X') && (A[2][2] == 'X')) ||

((A[0][0] == 'X') && (A[1][1] == 'X') && (A[2][2] == 'X')) ||

((A[2][0] == 'X') && (A[1][1] == 'X') && (A[0][2] == 'X')))

{

k++;

cout << "Игра закончена! ";

if ((A[x][y] == z) && (dgt == 1)) cout << name1 << " победа!" << endl;

if ((A[x][y] == c) && (dgt == 2)) cout << name1 << " победа!" << endl;

if ((A[x][y] == c) && (dgt == 1)) cout << name2 << " победа!" << endl;

if ((A[x][y] == z) && (dgt == 2)) cout << name2 << " победа!" << endl;

break;

}

}

if (!(k)) cout << "Игра закончилась вничью!" << endl;

system("pause");

return 0;

}