**ПРИЛОЖЕНИЕ А**

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

# КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

**«**КРЕСТИКИ-НОЛИКИ**»**

**ТЕКСТ ПРОГРАММЫ**

КР.АС56.190261-01 12 00

## Листов 9

|  |  |
| --- | --- |
| Руководитель | Т.Н. Гирель |
| Выполнил | М.В. Карпенко |
| Консультант |  |
| по ЕСПД | Т.Н. Гирель |
|  |  |

2020

СОДЕРЖАНИЕ

main.cpp – основной модуль в котором содержится меню для работы с программой.

#include <iostream>

#include<string>

#include<iomanip>

#include<Windows.h>

using namespace std;

void play\_area(char data[3][3])//выводит игровую зону

{

cout << " " << 1 << " | " << 2 << " | " << 3 << endl;

cout << "-------------" << endl;

cout << " " << 4 << " | " << 5 << " | " << 6 << endl;

cout << "-------------" << endl;

cout << " " << 7 << " | " << 8 << " | " << 9 << endl;

cout << "\nChoose position";

cout << endl << endl;

for (int k = 0; k < 3; k++)

{

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

{

if (data[k][i] == '1')

{

cout << " ";

}

else

{

cout << ' ' << data[k][i] << ' ';

}

if (i == 2)continue;

cout << " | ";

}

cout << endl;

if (k == 2) continue;

cout << "---------------" << endl;

}

}

int get\_key()

{

int key = 10;

bool flag = false;

while (key > 9 || key < 1)

{

if (flag)

{

return -1;

}

cin >> key;

flag = true;

}

return key;

}

bool matrix\_writing(char matrix[3][3], int index, char temp)

{

int t = 0;

while (index > 3)

{

t++;

index -= 3;

}

if ((matrix[t][index] == 'X' || matrix[t][index] == '0') && (temp == '0' || temp == 'X'))

return false;

else

{

matrix[t][index] = temp;

return true;

}

}

int y\_or\_n(char data[3][3], int index)

{

bool temp;

temp = matrix\_writing(data, index, 'X');

if (!temp)

return 0;

system("Cls");

play\_area(data);

while (true)

{

char answ = ' ';

cout << "Enter yes or no(y/n): ";

cin >> answ;

if (answ == 'n')

{

matrix\_writing(data, index, '1');

system("Cls");

play\_area(data);

return -1;

}

else

{

return 1;

}

}

}

bool person\_action(char data[3][3])

{

while (true)

{

int person\_decision = get\_key();

if (person\_decision == -1)

return false;

person\_decision--;

switch (y\_or\_n(data, person\_decision))

{

case 0:

return false;

case 1:

return true;

}

}

}

bool check\_gorizontal(char data[3][3], char flag1, char flag2)

{

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

{

int result = 0;

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

{

if (data[i][j] == flag1)

{

result++;

}

}

if (result == 2)

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

{

if (data[i][j] == '1')

{

data[i][j] = flag2;

return true;

}

}

}

return false;

}

bool check\_vertikal(char data[3][3], char flag1, char flag2)

{

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

{

int result = 0;

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

{

if (data[j][i] == flag1)

{

result++;

}

}

if (result == 2)

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

{

if (data[j][i] == '1')

{

data[j][i] = flag2;

return true;

}

}

}

return false;

}

bool check\_diagonals(char data[3][3], char flag1, char flag2)

{

int result = 0;

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

{

if (data[i][i] == flag1)

{

result++;

}

}

if (result == 2)

{

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

{

if (data[i][i] == '1')

{

data[i][i] = flag2;

return true;

}

}

}

result = 0;

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

{

if (data[i][2 - i] == flag1)

{

result++;

}

}

if (result == 2)

{

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

{

if (data[i][2 - i] == '1')

{

data[i][2 - i] = flag2;

return true;

}

}

}

return false;

}

bool other\_fields(char data[3][3])

{

if (data[1][1] == '1')

{

data[1][1] = '0';

return true;

}

if (data[0][0] == '1')

{

data[0][0] = '0';

return true;

}

if (data[0][2] == '1')

{

data[0][2] = '0';

return true;

}

if (data[2][0] == '1')

{

data[2][0] = '0';

return true;

}

if (data[2][2] == '1')

{

data[2][2] = '0';

return true;

}

if (data[0][1] == '1')

{

data[0][1] = '0';

return true;

}

if (data[1][0] == '1')

{

data[1][0] = '0';

return true;

}

if (data[1][2] == '1')

{

data[1][2] = '0';

return true;

}

if (data[2][1] == '1')

{

data[2][1] = '0';

return true;

}

}

bool person\_win(char data[3][3])

{

int result = 0;

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

{

result = 0;

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

{

if (data[i][j] == 'X')

{

result++;

}

}

if (result == 3)

return true;

}

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

{

result = 0;

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

{

if (data[j][i] == 'X')

{

result++;

}

}

if (result == 3)

return true;

}

result = 0;

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

{

if (data[i][i] == 'X')

{

result++;

}

}

if (result == 3)

{

return true;

}

result = 0;

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

{

if (data[i][2 - i] == 'X')

{

result++;

}

}

if (result == 3)

{

return true;

}

return false;

}

bool komputer\_action(char data[3][3])

{

if (check\_diagonals(data, '0', '0'))

{

return true;

}

if (check\_gorizontal(data, '0', '0'))

{

return true;

}

if (check\_vertikal(data, '0', '0'))

{

return true;

}

if (check\_diagonals(data, 'X', '0'))

{

return false;

}

if (check\_gorizontal(data, 'X', '0'))

{

return false;

}

if (check\_vertikal(data, 'X', '0'))

{

return false;

}

if (other\_fields(data))

{

return false;

}

return false;

}

void game()

{

char matrix[3][3] =

{

{'1','1','1'},

{'1','1','1'},

{'1','1','1'}

};

int iteration = 0;

system("Cls");

play\_area(matrix);

bool flag;

bool t = false;

while (true)

{

if (t)

{

cout << "Komputer Win" << endl;

break;

}

bool answer = person\_action(matrix);

flag = true;

if (!answer)

{

flag = false;

system("Cls");

play\_area(matrix);

cout << "You should try again" << endl;//выбрать число от 1 дот 9

}

else

{

iteration++;

}

if (iteration >= 5 && person\_win(matrix))

{

cout << "You Win" << endl;

break;

}

if (iteration == 9)

{

cout << "No Winers" << endl;

break;

}

if (flag)

{

t = komputer\_action(matrix);

iteration++;

system("Cls");

play\_area(matrix);

}

}

system("pause");

system("Cls");

}

void menu()

{

while (true)

{

cout << "\tMENU" << endl;

cout << "Choose:" << endl;

cout << "1 - game" << endl;

cout << "2 - exit" << endl;

char menu;

cin >> menu;

if (menu == '1')

game();

else

break;

}

}

int main()

{

menu();

system("pause");

return 0;

}