Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение  
высшего образования   
«Алтайский государственный технический университет им. И. И. Ползунова»

Факультет информационных технологий

Кафедра прикладной математики

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Отчет

по лабораторной работе № 3

по дисциплине

«Тестирование и отладка программного обеспечения»

Студент гр. ПИ-11

Корниенко. В. Р

Преподаватель,

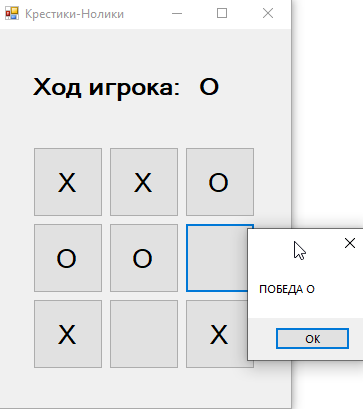
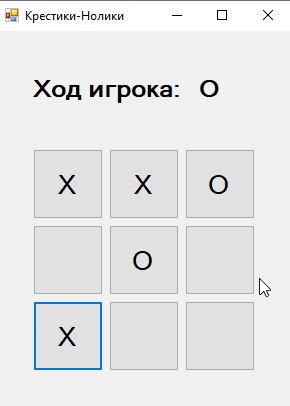
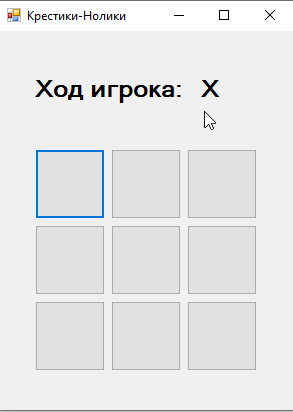
Ерёмин Я. Б.

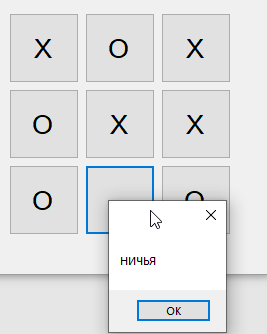
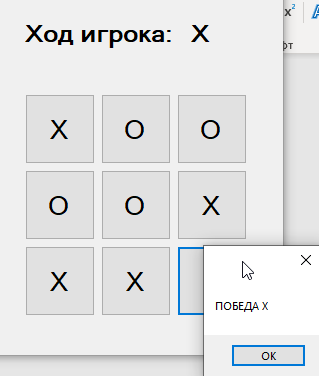
Барнаул 2025

**Тема:** Игра «Крестики-нолики»

**Ссылка на репозиторий:** <https://github.com/KornienkoVl/Test.Lab3>

**Тесты программы:**





**Вывод:** количество строк кода для юнит тестов превышает объем программы (281 против 269). При увеличении временных затрат на написание тестов уменьшились временные затраты на написание и дебагинг методов программы.

**Код тестов программы:**

namespace TicTac.Test

{

[TestClass]

public sealed class TestMove

{

[TestMethod]

public void TestMethodMakeMove()

{

Game game = new Game();

Assert.IsTrue(game.MakeMove(1,1));

//проверка изменения матрицы после хода

Assert.AreEqual(game.map[1,1],1);

}

[TestMethod]

public void TestMethodMakeMoveXY()

{

Game game = new Game();

Assert.IsFalse(game.MakeMove(-1, 1));

Assert.IsFalse(game.MakeMove(-1, -1));

Assert.IsFalse(game.MakeMove(1, -1));

Assert.IsFalse(game.MakeMove(3, 1));

Assert.IsFalse(game.MakeMove(2, 3));

Assert.IsFalse(game.MakeMove(3, 3));

Assert.IsFalse(game.MakeMove(-5, 6));

Assert.IsFalse(game.MakeMove(6, -5));

Assert.IsTrue(game.MakeMove(0, 0));

Assert.IsTrue(game.MakeMove(1, 1));

Assert.IsTrue(game.MakeMove(2, 2));

Assert.IsTrue(game.MakeMove(0, 2));

Assert.IsTrue(game.MakeMove(2, 0));

}

[TestMethod]

public void TestMethodMakeMoveEmpty()

{

Game game = new Game();

Assert.IsTrue(game.MakeMove(1, 1));

Assert.IsFalse(game.MakeMove(1, 1));

}

[TestMethod]

public void TestMethodMakeMoveChangePlayer()

{

Game game = new Game();

game.MakeMove(0, 0);

game.MakeMove(0, 1);

game.MakeMove(0, 2);

Assert.AreEqual(game.map[0, 0], 1);

Assert.AreEqual(game.map[0, 1], -1);

Assert.AreEqual(game.map[0, 2], 1);

}

}

[TestClass]

public sealed class TestChangePlayer

{

[TestMethod]

public void TestMethodChangePlayer()

{

Game game = new Game();

Assert.AreEqual(game.player, 1);

game.ChangePlayer();

Assert.AreEqual(game.player, -1);

game.ChangePlayer();

Assert.AreEqual(game.player, 1);

}

}

[TestClass]

public sealed class TestCheckWin

{

[TestMethod]

public void TestMethodCheckWinHorizon()

{

Game game = new Game();

Assert.AreEqual(game.checkWin(), 0);

game.map[0, 0] = 1;

game.map[0, 1] = 1;

game.map[0, 2] = 1;

Assert.AreEqual(game.checkWin(), 1);

game.map[0, 0] = 0;

game.map[0, 1] = 0;

game.map[0, 2] = 0;

game.map[2, 0] = -1;

game.map[2, 1] = -1;

game.map[2, 2] = -1;

Assert.AreEqual(game.checkWin(), -1);

}

[TestMethod]

public void TestMethodCheckWinVertical()

{

Game game = new Game();

Assert.AreEqual(game.checkWin(), 0);

game.map[0, 0] = 1;

game.map[1, 0] = 1;

game.map[2, 0] = 1;

Assert.AreEqual(game.checkWin(), 1);

game.map[0, 0] = 0;

game.map[1, 0] = 0;

game.map[2, 0] = 0;

game.map[0, 2] = -1;

game.map[1, 2] = -1;

game.map[2, 2] = -1;

Assert.AreEqual(game.checkWin(), -1);

}

[TestMethod]

public void TestMethodCheckWinDiagonal()

{

Game game = new Game();

Assert.AreEqual(game.checkWin(), 0);

game.map[0, 0] = 1;

game.map[1, 1] = 1;

game.map[2, 2] = 1;

Assert.AreEqual(game.checkWin(), 1);

game.map[0, 0] = 0;

game.map[1, 1] = 0;

game.map[2, 2] = 0;

game.map[0, 2] = -1;

game.map[1, 1] = -1;

game.map[2, 0] = -1;

Assert.AreEqual(game.checkWin(), -1);

}

}

[TestClass]

public sealed class TestCheckDraw

{

[TestMethod]

public void TestMethodCheckDraw()

{

Game game = new Game();

Assert.IsFalse(game.checkDraw());

game.MakeMove(0, 0);

game.MakeMove(0, 1);

game.MakeMove(0, 2);

Assert.IsFalse(game.checkDraw());

game.MakeMove(1, 0);

game.MakeMove(1, 1);

game.MakeMove(1, 2);

Assert.IsFalse(game.checkDraw());

game.MakeMove(2, 0);

game.MakeMove(2, 1);

Assert.IsFalse(game.checkDraw());

game.MakeMove(2, 2);

Assert.IsTrue(game.checkDraw());

}

}

[TestClass]

public sealed class TestClear

{

[TestMethod]

public void TestMethodClear()

{

Game game = new Game();

game.MakeMove(0, 0);

game.MakeMove(2, 1);

game.MakeMove(1, 2);

game.Clear();

Assert.AreEqual(game.map[0, 0], 0);

Assert.AreEqual(game.map[2, 1], 0);

Assert.AreEqual(game.map[1, 2], 0);

Assert.AreEqual(game.player, 1);

}

}

[TestClass]

public sealed class TestPlay

{

[TestMethod]

public void TestMethodPlayWin()

{

Game game = new Game();

Assert.AreEqual(game.Play(0, 0), 0);

Assert.AreEqual(game.Play(1, 0), 0);

Assert.AreEqual(game.Play(0, 1), 0);

Assert.AreEqual(game.Play(0, 2), 0);

Assert.AreEqual(game.Play(2, 1), 0);

Assert.AreEqual(game.Play(1, 1), 0);

Assert.AreEqual(game.Play(2, 2), 0);

Assert.AreEqual(game.Play(1, 2), 3);

Assert.AreEqual(game.map[0, 0], 0);

Assert.AreEqual(game.map[1, 0], 0);

Assert.AreEqual(game.map[0, 1], 0);

Assert.AreEqual(game.map[0, 2], 0);

Assert.AreEqual(game.map[2, 1], 0);

Assert.AreEqual(game.map[1, 1], 0);

Assert.AreEqual(game.map[2, 2], 0);

Assert.AreEqual(game.map[1, 2], 0);

Assert.AreEqual(game.player, 1);

}

[TestMethod]

public void TestMethodPlayDraw()

{

Game game = new Game();

Assert.AreEqual(game.Play(0, 0), 0);

Assert.AreEqual(game.Play(1, 1), 0);

Assert.AreEqual(game.Play(0, 1), 0);

Assert.AreEqual(game.Play(0, 2), 0);

Assert.AreEqual(game.Play(2, 0), 0);

Assert.AreEqual(game.Play(1, 0), 0);

Assert.AreEqual(game.Play(2, 1), 0);

Assert.AreEqual(game.Play(2, 2), 0);

Assert.AreEqual(game.Play(1, 2), 2);

Assert.AreEqual(game.map[0, 0], 0);

Assert.AreEqual(game.map[1, 0], 0);

Assert.AreEqual(game.map[0, 1], 0);

Assert.AreEqual(game.map[0, 2], 0);

Assert.AreEqual(game.map[2, 1], 0);

Assert.AreEqual(game.map[1, 1], 0);

Assert.AreEqual(game.map[2, 2], 0);

Assert.AreEqual(game.map[2, 0], 0);

Assert.AreEqual(game.map[1, 2], 0);

Assert.AreEqual(game.player, 1);

}

[TestMethod]

public void TestMethodPlayNoMove()

{

Game game = new Game();

Assert.AreEqual(game.Play(0, 0), 0);

Assert.AreEqual(game.Play(0, 0), 4);

}

}

}

**Код программы:**

namespace TicTac

{

public partial class Form1: Form

{

Game game = new Game();

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

lblPlayer.Text = "X";

}

private void btn1\_Click(object sender, EventArgs e)

{

int var = game.Play(0, 0);

if (var == 0)

{

if (game.player == 1) { btn1.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn1.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void btn2\_Click(object sender, EventArgs e)

{

int var = game.Play(0, 1);

if (var == 0)

{

if (game.player == 1) { btn2.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn2.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void btn3\_Click(object sender, EventArgs e)

{

int var = game.Play(0, 2);

if (var == 0)

{

if (game.player == 1) { btn3.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn3.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void btn4\_Click(object sender, EventArgs e)

{

int var = game.Play(1, 0);

if (var == 0)

{

if (game.player == 1) { btn4.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn4.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void btn5\_Click(object sender, EventArgs e)

{

int var = game.Play(1, 1);

if (var == 0)

{

if (game.player == 1) { btn5.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn5.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void btn6\_Click(object sender, EventArgs e)

{

int var = game.Play(1, 2);

if (var == 0)

{

if (game.player == 1) { btn6.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn6.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void btn7\_Click(object sender, EventArgs e)

{

int var = game.Play(2, 0);

if (var == 0)

{

if (game.player == 1) { btn7.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn7.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void btn8\_Click(object sender, EventArgs e)

{

int var = game.Play(2, 1);

if (var == 0)

{

if (game.player == 1) { btn8.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn8.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void btn9\_Click(object sender, EventArgs e)

{

int var = game.Play(2, 2);

if (var == 0)

{

if (game.player == 1) { btn9.Text = "O"; lblPlayer.Text = "X"; }

if (game.player == -1) { btn9.Text = "X"; lblPlayer.Text = "O"; }

}

if (var == 1) { MessageBox.Show("ПОБЕДА Х"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 2) { MessageBox.Show("НИЧЬЯ"); clearBtns(); lblPlayer.Text = "X"; }

if (var == 3) { MessageBox.Show("ПОБЕДА О"); clearBtns(); lblPlayer.Text = "X"; }

}

private void clearBtns()

{

btn1.Text = "";

btn2.Text = "";

btn3.Text = "";

btn4.Text = "";

btn5.Text = "";

btn6.Text = "";

btn7.Text = "";

btn8.Text = "";

btn9.Text = "";

}

}

public class Game

{

public int[,] map = { { 0, 0, 0 }, { 0, 0, 0 }, { 0, 0, 0 } };

public int player = 1;

public bool MakeMove(int cellX, int cellY)

{

if (cellX < 0 || cellX > 2 || cellY < 0 || cellY > 2)

return false;

if (map[cellX, cellY] != 0)

return false;

map[cellX, cellY] = player;

ChangePlayer();

return true;

}

public void ChangePlayer()

{

player = 0 - player;

}

// 1 - победа Х

// 0 - нет победителя

//-1 - победа О

public int checkWin()

{

int checkSum = 0;

//горизонталь

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

{

checkSum = map[i, 0] + map[i, 1] + map[i, 2];

if (checkSum == 3) return 1;

if (checkSum == -3) return -1;

}

//вертикаль

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

{

checkSum = map[0, i] + map[1, i] + map[2, i];

if (checkSum == 3) return 1;

if (checkSum == -3) return -1;

}

//диагонали

checkSum = map[0, 0] + map[1, 1] + map[2, 2];

if (checkSum == 3) return 1;

if (checkSum == -3) return -1;

checkSum = map[0, 2] + map[1, 1] + map[2, 0];

if (checkSum == 3) return 1;

if (checkSum == -3) return -1;

return 0;

}

public bool checkDraw()

{

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

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

if (map[i, j] == 0) return false;

return true;

}

public void Clear()

{

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

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

map[i, j] = 0;

player = 1;

}

// 0 - игра продолжается

// 1 - победил Х

// 2 - ничья

// 3 - победил O

// 4 - нельзя сделать ход

public int Play(int cellX, int cellY)

{

if (MakeMove(cellX, cellY))

{

int win = checkWin();

if (win == 1)

{

Clear();

return 1;

}

if (win == -1)

{

Clear();

return 3;

}

if (checkDraw())

{

Clear();

return 2;

}

return 0;

}

return 4;

}

};

}