ФГБОУ ВО «КЕМЕРОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

КАФЕДРА ЮНЕСКО ПО ИНФОРМАЦИОННЫМ ВЫЧИСЛИТЕЛЬНЫМ ТЕХНОЛОГИЯМ

ЛАБОРАТОРНАЯ РАБОТА № 2

по дисциплине «Языки программирования»

Семестр 2

Тема: Динамические многомерные массивы. Дополнительное задание

Выполнил:

студент группы ФИТ-194

Максименко Р.В.

Кемерово, 2020

Напишите программу, реализующую игру "крестики-нолики". Матрица для игры в крестики-нолики имеет вид двумерного массива символов 3 на 3. Пользователь всегда играет крестиками, а компьютер - ноликами. Когда ходит пользователь, «X» помещается в указанную позицию матрицы. Когда наступает очередь ходить компьютеру, он сканирует матрицу и помещает «О» в пустое место матрицы. Если компьютер не может найти пустой ячейки, он выводит результат игры и завершает работу программы.

#include "pch.h"

#include <iostream>

#include <fstream>

#include <conio.h>

#include <stdlib.h>

#define SPACE ' '

char matrix[3][3] = { /\* matrix for tic-tac-toe \*/

{SPACE, SPACE, SPACE},

{SPACE, SPACE, SPACE},

{SPACE, SPACE, SPACE}

};

void get\_computer\_move(void), get\_player\_move(void);

void disp\_matrix(void);

char check(void);

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

char done;

done = SPACE;

do {

disp\_matrix(); //game board output

get\_player\_move(); //player walks

done = check(); //victory check

if (done != SPACE) break; //winner

get\_computer\_move(); //computer is walking

done = check(); //victory check

} while (done == SPACE);

if (done == 'X') printf("You won!\n");

else printf("Computer won!!!!\n");

disp\_matrix(); //display of the resulting position

return 0;

}

//player turn input

void get\_player\_move(void)

{

int x, y;

cout << "Enter the coordinates for your X" << endl;

cout << "Line=";

cin >> x;

cout << "Column=";

cin >> y;

x--; y--;

if (x < 0 || y < 0 || x>2 || y>2 || matrix[x][y] != SPACE)

{

printf("Wrong move, try again.\n");

get\_player\_move();

}

else matrix[x][y] = 'X';

}

//computer running

void get\_computer\_move(void)

{

register int t;

char \*p;

p = (char \*)matrix;

for (t = 0; \*p != SPACE && t < 9; ++t) p++;

if (t == 9)

{

exit(0); //game over

}

else \*p = 'O';

}

//game board display

void disp\_matrix(void)

{

int t;

for (t = 0; t < 3; t++)

{

printf(" %c | %c | %c", matrix[t][0], matrix[t][1], matrix[t][2]);

if (t != 2) printf("\n---|---|---\n");

}

printf("\n");

}

//victory check

char check(void)

{

int t;

char \*p;

for (t = 0; t < 3; t++) { //string check

p = &matrix[t][0];

if (\*p == \*(p + 1) && \* (p + 1) == \*(p + 2)) return \*p;

}

for (t = 0; t < 3; t++) { //column check

p = &matrix[0][t];

if (\*p == \*(p + 3) && \*(p + 3) == \*(p + 6)) return \*p;

}

//diagonal check

if (matrix[0][0] == matrix[1][1] && matrix[1][1] == matrix[2][2])

return matrix[0][0];

if (matrix[0][2] == matrix[1][1] && matrix[1][1] == matrix[2][0])

return matrix[0][2];

return SPACE;

}







