

INSTITUTO POLITECNICO NACIONAL

ESIME CULHUACAN

Fundamentos de Programación

1ICV45

Profesor: Roberto Osornio Soto

Proyecto Final

Alumno: García García Jonathan Eduardo

Código Fuente

/\*

Practica Final

Alumno: García García Jonathan Eduardo

Grupo: 1ICV45

Profesor: Roberto Osornio Soto

Materia: Fundamentos de Programación

\*/

//Includes

#include "stdafx.h"

#include <stdio.h>

#include <math.h>

#include <conio.h>

#include <string.h>

#include <Windows.h>

#include <iostream>

#include <cstdio>

#include <WinUser.h>

#include <string>

#include <cstring>

#include <sstream>

#include <sql.h>

#include <time.h>

#include <vector>

#include <algorithm>

#include <iterator>

#import "C:\Program Files\Common Files\System\ado\msado15.dll" no\_namespace rename("EOF", "EndOfFile")

#define FLECHA 224

#define ARRIBA 72

#define ABAJO 80

#define IZQUIERDA 75

#define DERECHA 77

#define ENTER 13

#define ESC 27

using namespace std;

std::string input;

//Variables Globales

char\* CadenaCon = "Provider=SQLOLEDB.1;Persist Security Info=False;Initial Catalog=GATOSDB;Data Source=GATOSDB2.mssql.somee.com;user id=Jonathan7;pwd=4nl7dmkq9g;";

static char\* contrincante = "Jugador 2";

char apodo[100] = "";

int NewHeight = ((GetSystemMetrics(SM\_CYSCREEN) - 750) / 2);

int Height = GetSystemMetrics(SM\_CYSCREEN);

int Width = GetSystemMetrics(SM\_CXSCREEN);

int NewWidth = (Width - 1080) / 2;

int WindowWidth = 1080;

int WindowHeight = 750;

bool ErrorAlConectarResult = false;

bool AsyncAleatorioActive = false;

bool AsyncAnimation = false;

bool AsyncCodigo = false;

bool AsyncMarco = false;

bool AsyncTurno = false;

bool conectado = false;

bool puedeTirar = true;

bool MITURNO = false;

bool turno = false;

HWND = GetConsoleWindow();

//Prototipos

bool Escribir(char\* codigo, char\* apodo, char disponible = '1', char turno = '1', char buscando = '0', char\* casillas = ",0,0,0,0,0,0,0,0,0");

bool ActualizarTirosVzPlayer(bool newturno, bool aleatorio = false);

bool CrearVentana(char\* title, int width, int height);

bool ConectarAleatorio(char\* code);

bool AleatorioOcupado(char\* code);

bool OcuparAleatorio(char\* code);

bool SeAutoGano(int results[8]);

bool ExisteCodigo(char\* code);

bool Conectar(char\* codigo);

bool SilenceGano(int modo);

bool Ocupado(char\* codigo);

bool TiroDisponible();

bool TiroParaGanar();

bool Gano(int modo);

bool Bloquear();

bool TiraCpu();

void EscribirAleatorio(char\* codigo, char\* apodo, char disponible = '1', char turno = '1', char buscando = '1', char\* casillas = ",0,0,0,0,0,0,0,0,0");

void PrintColorXy(char\* format, const char\* texto, int x = -1, int y = -1, int background = -1, int textColor = -1);

void CargarCasillasMultiplayer(char\* apodo, char\* contrincante, bool aleatorio = false);

void CargarCasillasMultiplayerAleatorio(char\* apodo, char\* contrincante);

void gotoxy(int x, int y, COORD coordenadas);

void RedibujarMarcos(int background = 15);

void SetCursor(HWND h, short x, short y);

void EnviarTiro(bool aleatorio = false);

void SendString(HWND h, char \*text);

void textbackground(int a, int b);

void Ocupar(\_bstr\_t code);

void ErrorAlConectar();

void Introduccion();

void vzAleatorio();

void VzFriend();

void VzCPU();

char\* appendCharToCharArray(char\* array, char a);

char codigoConectar[5] = "XXXX";

char codigo[5] = "XXXX";

int GetFontSize(HANDLE windowHandle, COORD \*size);

int SetFontSize(HANDLE windowHandle, COORD size);

int casillas[] = { 0,0,0,0,0,0,0,0,0 };

int MenuPrincipal();

int pressKey();

int wherex();

int wherey();

BOOL WINAPI MoveWindow(\_In\_ HWND, \_In\_ int NewWidth, \_In\_ int NewHeight, \_In\_ int WindowWidth, \_In\_ int WindowHeight, \_In\_ BOOL bRepaint);

DWORD WINAPI AsyncTableAnimation(void\* data);

DWORD WINAPI AsyncMarcoAnimation(void\* data);

DWORD WINAPI AsyncConectar(void\* data);

DWORD WINAPI AsyncGetTurno(void\* data);

DWORD WINAPI AsyncAleatorio(void\* data);

//Funciones

int main(bool skip = true)

{

if (skip) {

CrearVentana("Hola", WindowWidth, WindowHeight);

MoveWindow(hWnd, NewWidth, NewHeight, WindowWidth, WindowHeight, TRUE);

system("mode con: cols=100 lines=40");

HANDLE h = GetStdHandle(STD\_OUTPUT\_HANDLE);

COORD size;

if (GetFontSize(h, &size))

{

if (size.X != 10)

{

size.X = 10;

}

if (size.Y != 18)

{

size.Y = 18;

}

SetFontSize(h, size);

}

HANDLE thread = CreateThread(NULL, 0, AsyncMarcoAnimation, NULL, 0, NULL);

HANDLE thread2 = CreateThread(NULL, 0, AsyncTableAnimation, NULL, 0, NULL);

HANDLE thread3 = CreateThread(NULL, 0, AsyncConectar, NULL, 0, NULL);

HANDLE thread4 = CreateThread(NULL, 0, AsyncGetTurno, NULL, 0, NULL);

HANDLE thread5 = CreateThread(NULL, 0, AsyncAleatorio, NULL, 0, NULL);

CONSOLE\_CURSOR\_INFO info;

info.dwSize = 100;

info.bVisible = FALSE;

SetConsoleCursorInfo(h, &info);

unsigned int time\_ui = static\_cast<unsigned int>(time(NULL));

srand(time\_ui);

Introduccion();

}

inicio:

RedibujarMarcos();

switch (MenuPrincipal())

{

case 0:

RedibujarMarcos();

VzCPU();

RedibujarMarcos();

goto inicio;

break;

case 1:

RedibujarMarcos();

VzFriend();

goto inicio;

break;

case 2:

RedibujarMarcos();

vzAleatorio();

goto inicio;

break;

default:

RedibujarMarcos();

Sleep(30);

exit(0);

break;

}

\_getch();

return 0;

}

bool CrearVentana(char\* title, int width, int height)

{

WNDCLASS wc;

DWORD dwExStyle;

DWORD dwStyle;

RECT WindowRect;

WindowRect.left = (long)0;

WindowRect.right = (long)width;

WindowRect.top = (long)0;

WindowRect.bottom = (long)height;

wc.style = CS\_HREDRAW | CS\_VREDRAW | CS\_OWNDC;

wc.cbClsExtra = 0;

wc.cbWndExtra = 0;

wc.hIcon = LoadIcon(NULL, IDI\_WINLOGO);

wc.hCursor = LoadCursor(NULL, IDC\_ARROW);

wc.hbrBackground = NULL;

wc.lpszMenuName = NULL;

dwExStyle = WS\_EX\_APPWINDOW | WS\_EX\_WINDOWEDGE;

dwStyle = (WS\_OVERLAPPED | WS\_CAPTION | WS\_SYSMENU | WS\_MINIMIZEBOX | WS\_MAXIMIZEBOX);

AdjustWindowRectEx(&WindowRect, dwStyle, FALSE, dwExStyle);

static PIXELFORMATDESCRIPTOR pfd =

{

sizeof(PIXELFORMATDESCRIPTOR),

1,

PFD\_DRAW\_TO\_WINDOW |

PFD\_SUPPORT\_OPENGL |

PFD\_DOUBLEBUFFER,

PFD\_TYPE\_RGBA,

24,

0, 0, 0, 0, 0, 0,

0,

0,

0,

0, 0, 0, 0,

24,

0,

0,

PFD\_MAIN\_PLANE,

0,

0, 0, 0

};

ShowScrollBar(hWnd, SB\_BOTH, FALSE);

ShowWindow(hWnd, SW\_SHOW);

SetForegroundWindow(hWnd);

SetFocus(hWnd);

::SetWindowLong(hWnd, GWL\_STYLE, GetWindowLong(hWnd, GWL\_STYLE)&~WS\_SIZEBOX);

return true;

}

void gotoxy(int x = -1, int y = -1, COORD coordenadas = { 0,0 })

{

if (x < 0)

{

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coordenadas);

}

else

{

COORD cord = { (short)x, (short)y };

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), cord);

}

}

int GetFontSize(HANDLE windowHandle, COORD \*size)

{

CONSOLE\_FONT\_INFOEX font = { sizeof(CONSOLE\_FONT\_INFOEX) };

if (!GetCurrentConsoleFontEx(windowHandle, 0, &font))

{

return 0;

}

\*size = font.dwFontSize;

return 1;

}

int SetFontSize(HANDLE windowHandle, COORD size)

{

CONSOLE\_FONT\_INFOEX font = { sizeof(CONSOLE\_FONT\_INFOEX) };

if (!GetCurrentConsoleFontEx(windowHandle, 0, &font))

{

return 0;

}

font.dwFontSize = size;

if (!SetCurrentConsoleFontEx(windowHandle, 0, &font))

{

return 0;

}

return 1;

}

void textbackground(int a, int b = -1)

{

if (b <= -1)

{

b = 7;

}

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), a << 4 | b);

}

int RandRange(int Min, int Max)

{

int diff = Max - Min;

return (int)(((double)(diff + 1) / RAND\_MAX) \* rand() + Min);

}

void Introduccion()

{

int randcolor = RandRange(1, 15);

int sleep = 3;

gotoxy(1, 1);

Sleep(sleep);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xC9");

for (int i = 2; i < 98; i++)

{

gotoxy(i, 1);

Sleep(1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xCD");

if (i > 37) {

continue;

}

gotoxy(1, i);

Sleep(sleep);

textbackground(randcolor, 0);

printf("\xBA");

}

sleep = 2;

gotoxy(98, 1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xBB");

gotoxy(1, 38);

textbackground(randcolor, 0);

printf("\xC8");

gotoxy(98, 38);

textbackground(randcolor, 0);

printf("\xBC");

for (int i = 2; i < 98; i++)

{

gotoxy(i, 38);

Sleep(1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xCD");

if (i > 37) {

continue;

}

gotoxy(98, i);

Sleep(sleep);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xBA");

}

for (int i = 2; i < 98; i++)

{

for (int y = 2; y < 38; y++)

{

gotoxy(i, y);

textbackground(07, 0);

printf(" ");

}

}

gotoxy(4, 5);

printf(" \_\_\_\_\_\_ \_ \_\_\_\_\_ \_ \_ ");

gotoxy(4, 6);

printf("| \_\_\_\_| (\_) / \_\_\_\_| | || | ");

gotoxy(4, 7);

printf("| |\_\_ \_\_\_ \_ \_ \_\_ \_\_\_ \_\_\_ | | \_ \_ | || |\_\_ \_ \_ \_\_ \_ \_\_\_ \_\_ \_ \_ \_\_ ");

gotoxy(4, 8);

printf("| \_\_| / \_\_|| || '\_ ` \_ \\ / \_ \\ | | | | | || || '\_ \\ | | | | / \_` | / \_\_|/ \_` || '\_ \\ ");

gotoxy(4, 9);

printf("| |\_\_\_\_ \\\_\_ \\| || | | | | || \_\_/ | |\_\_\_\_| |\_| || || | | || |\_| || (\_| || (\_\_| (\_| || | | |");

gotoxy(4, 10);

printf("|\_\_\_\_\_\_||\_\_\_/|\_||\_| |\_| |\_| \\\_\_\_| \\\_\_\_\_\_|\\\_\_,\_||\_||\_| |\_| \\\_\_,\_| \\\_\_,\_| \\\_\_\_|\\\_\_,\_||\_| |\_|");

gotoxy(2, 37);

printf("\*Use las flechas y teclas del teclado para navegar presione ESC para salir");

for (int x = 4; x <= 94; x++)

{

for (int y = 12; y <= 36; y++)

{

if (y == 13)

{

gotoxy(x, y);

textbackground(0, 15);

printf(" ");

continue;

}

gotoxy(x, y);

textbackground(15, 0);

printf(" ");

}

}

textbackground(0, 15);

gotoxy(30, 13);

printf("JUGAR GATO SOLITARIO Y VZ (Jugador 2)");

gotoxy(5, 16);

textbackground(15, 0);

printf("Alumno: Garc%ca Jonathan Eduardo\n", 161, 161);

gotoxy(5, 17);

printf("Grupo: 1ICV45\n");

gotoxy(5, 18);

printf("Profesor: Roberto Osornio Soto\n");

gotoxy(5, 19);

printf("Materia: Fundamentos de Programaci%cn\n", 162);

gotoxy(5, 21);

printf("En esta versi%cn de el juego de gato clasico podras jugar vz CPU o contra algun amigo...\n", 162);

gotoxy(5, 23);

printf(">>>Pasos para jugar con amigos:\n");

gotoxy(5, 24);

printf("1) Debe seleccionar la opcion para jugar vz amigos y compartir sus codigos de acceso para\n");

gotoxy(5, 25);

printf("que pueda establecerse una conexi%cn.", 162);

gotoxy(5, 26);

printf("2) Ambos deben seleccionar la opcion para jugar con amigos...\n");

gotoxy(5, 28);

printf(">>>Para escoger un oponente al azar simplemte seleccione la opcion marcada con este\n");

gotoxy(5, 29);

printf("nombre y espere su turno... ");

gotoxy(5, 31);

printf(">>>Para jugar contra el Cpu seleccione la opcion marcada con este nombre y comience. \n");

PrintColorXy("%s", " ", 30, 34, 0, 15);

PrintColorXy("%s", ">>>>PRESIONE ENTER PARA CONTINUAR<<<<", 30, 35, 0, 15);

PrintColorXy("%s", " ", 30, 36, 0, 15);

AsyncMarco = true;

\_getch();

AsyncMarco = false;

}

void RedibujarMarcos(int background)

{

int randcolor = RandRange(1, 15);

int sleep = 2;

gotoxy(1, 1);

Sleep(sleep);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xC9");

for (int i = 2; i < 98; i++)

{

gotoxy(i, 1);

Sleep(1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xCD");

if (i > 37) {

continue;

}

gotoxy(1, i);

Sleep(sleep);

textbackground(randcolor, 0);

printf("\xBA");

}

sleep = 2;

gotoxy(98, 1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xBB");

gotoxy(1, 38);

textbackground(randcolor, 0);

printf("\xC8");

gotoxy(98, 38);

textbackground(randcolor, 0);

printf("\xBC");

for (int i = 2; i < 98; i++)

{

gotoxy(i, 38);

Sleep(1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xCD");

if (i > 37) {

continue;

}

gotoxy(98, i);

Sleep(sleep);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xBA");

}

for (int i = 2; i < 98; i++)

{

for (int y = 2; y < 38; y++)

{

gotoxy(i, y);

textbackground(background, 0);

printf(" ");

}

}

}

int MenuPrincipal()

{

PrintColorXy("%s", "\*Use las flechas y teclas del teclado para navegar presione ESC para salir", 2, 37);

AsyncAnimation = true;

int seleccion = 0;

int j = 0;

int tecla = 0;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0);

do

{

AsyncAnimation = false;

Sleep(10);

j = 0;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 3, 30, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

PrintColorXy("%s", (seleccion == j ? "\xBA>>>>>>>>JUGADOR VZ CPU<<<<<<<<\xBA" : "\xBA JUGADOR VZ CPU \xBA"), 3, 31, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 3, 32, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

j++;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 36, 30, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

PrintColorXy("%s", (seleccion == j ? "\xBA>>>>JUGADOR 1 VZ JUGADOR 2<<<<\xBA" : "\xBA JUGADOR 1 VZ JUGADOR 2 \xBA"), 36, 31, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 36, 32, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

j++;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 69, 30, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

PrintColorXy("%s", (seleccion == j ? "\xBA>>>>OPONENTE ALEATORIO<<<<\xBA" : "\xBA OPONENTE ALEATORIO \xBA"), 69, 31, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 69, 32, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0));

AsyncAnimation = true;

tecla = \_getch();

AsyncAnimation = false;

if (tecla == FLECHA) {

tecla = \_getch();

if (tecla == ARRIBA || tecla == DERECHA) {

seleccion++;

}

else

{

if (tecla == ABAJO || tecla == IZQUIERDA) {

seleccion--;

}

}

}

else

{

if (tecla == ESC)

{

return 3;

}

}

if (seleccion < 0)

{

seleccion = 2;

}

if (seleccion > 2)

{

seleccion = 0;

}

} while (tecla != ENTER);

return seleccion;

}

void PrintColorXy(char\* format, const char\* texto, int x, int y, int background, int textColor)

{

if (background > -1)

{

textbackground(background, textColor);

}

if (x > -1 && y > -1)

{

COORD c = { (short)x,(short)y };

gotoxy(-1, -1, c);

}

if (wherex() == x&&wherey() == y)

{

printf(format, texto);

}

}

DWORD WINAPI AsyncConectar(void\* data) {

while (0 == 0)

{

if (AsyncCodigo)

{

string code(codigo);

ErrorAlConectarResult = false;

if (code.compare("XXXX") != 0 && Ocupado(codigo) && !ErrorAlConectarResult)

{

turno = true;

conectado = true;

pressKey();

return 0;

}

if (ErrorAlConectarResult) {

return 0;

}

}

else { Sleep(2000); }

}

return 0;

}

DWORD WINAPI AsyncTableAnimation(void\* data) {

while (0 == 0)

{

if (AsyncAnimation)

{

int results[] = { 0,0,0,0,0,0,0,0,0 };

results[RandRange(0, 8)] = 1;

results[RandRange(0, 8)] = 2;

results[RandRange(0, 8)] = 1;

results[RandRange(0, 8)] = 2;

results[RandRange(0, 8)] = 1;

results[RandRange(0, 8)] = 2;

results[RandRange(0, 8)] = 1;

results[RandRange(0, 8)] = 2;

results[RandRange(0, 8)] = 1;

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

{

if (results[i] == 0)

{

if (i > 0 && results[i - 1] == 1)

{

results[i] = 2;

}

else

{

if (i > 0 && results[i - 1] == 2)

{

results[i] = 1;

}

else

{

results[i] = RandRange(1, 2);

}

}

}

}

int temp = 7;

int tempx = 0;

int sleep = 1000;

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

{

if (!AsyncAnimation)

{

break;

}

if (i >= 6)

{

temp = 13;

}

else

{

if (i <= 2)

{

temp = 1;

}

else

{

if (i >= 3 && i <= 5)

{

temp = 7;

tempx = 1;

}

}

}

if (!AsyncAnimation)

{

break;

}

switch (i)

{

case 0:

tempx = 0;

break;

case 1:

tempx = 11;

break;

case 2:

tempx = 22;

break;

case 3:

tempx = 0;

break;

case 4:

tempx = 11;

break;

case 5:

tempx = 22;

break;

case 6:

tempx = 0;

break;

case 7:

tempx = 11;

break;

case 8:

tempx = 22;

break;

}

if (!AsyncAnimation)

{

break;

}

switch (results[i])

{

case 0:

if (!AsyncAnimation)

{

break;

}

PrintColorXy("%s", " \n", (31 + tempx), 6 + temp, 15, 0);

PrintColorXy("%s", " \n", (31 + tempx), 7 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 8 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 10 + temp);

Sleep(sleep);

break;

case 1:

if (!AsyncAnimation)

{

break;

}

PrintColorXy("%s", " \\ \\ / /", (31 + tempx), 6 + temp, 9, 15);

PrintColorXy("%s", " \\ V / ", (31 + tempx), 7 + temp);

PrintColorXy("%s", " > < ", (31 + tempx), 8 + temp);

PrintColorXy("%s", " / . \\ ", (31 + tempx), 9 + temp);

PrintColorXy("%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp);

Sleep(sleep);

break;

case 2:

if (!AsyncAnimation)

{

break;

}

PrintColorXy("%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, 4, 15);

PrintColorXy("%s", "| | | |", (31 + tempx), 7 + temp);

PrintColorXy("%s", "| | | |", (31 + tempx), 8 + temp);

PrintColorXy("%s", "| `--' |", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp);

Sleep(sleep);

break;

}

if (!AsyncAnimation)

{

break;

}

int ar[] = { 0,0,0,0,0,0,0,0,0 };

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

{

ar[j] = results[j];

}

if (SeAutoGano(ar))

{

Sleep(2000);

if (!AsyncAnimation)

{

break;

}

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

{

if (!AsyncAnimation)

{

break;

}

if (i >= 6)

{

temp = 13;

}

else

{

if (i <= 2)

{

temp = 1;

}

else

{

if (i >= 3 && i <= 5)

{

temp = 7;

tempx = 1;

}

}

}

if (!AsyncAnimation)

{

break;

}

switch (i)

{

case 0:

tempx = 0;

break;

case 1:

tempx = 11;

break;

case 2:

tempx = 22;

break;

case 3:

tempx = 0;

break;

case 4:

tempx = 11;

break;

case 5:

tempx = 22;

break;

case 6:

tempx = 0;

break;

case 7:

tempx = 11;

break;

case 8:

tempx = 22;

break;

}

if (!AsyncAnimation)

{

break;

}

PrintColorXy("%s", " \n", (31 + tempx), 6 + temp, 15, 0);

PrintColorXy("%s", " \n", (31 + tempx), 7 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 8 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 10 + temp);

}

i = 9;

}

}

}

else { Sleep(2000); }

}

return 0;

}

DWORD WINAPI AsyncMarcoAnimation(void\* data) {

while (AsyncMarco)

{

int randcolor = RandRange(1, 15);

gotoxy(1, 1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xC9");

for (int i = 2; i < 98; i++)

{

gotoxy(i, 1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xCD");

if (i > 37) {

continue;

}

gotoxy(1, i);

textbackground(randcolor, 0);

printf("\xBA");

}

gotoxy(98, 1);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xBB");

gotoxy(1, 38);

textbackground(randcolor, 0);

printf("\xC8");

gotoxy(98, 38);

textbackground(randcolor, 0);

printf("\xBC");

for (int i = 2; i < 98; i++)

{

gotoxy(i, 38);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xCD");

if (i > 37) {

continue;

}

gotoxy(98, i);

randcolor = RandRange(1, 15);

textbackground(randcolor, 0);

printf("\xBA");

}

Sleep(500);

}

return 0;

}

DWORD WINAPI AsyncGetTurno(void\* data) {

while (0 == 0)

{

if (AsyncTurno)

{

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t strCnn(CadenaCon);

\_bstr\_t valField1;

\_RecordsetPtr pRstAuthors = NULL;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

code = result;

\_bstr\_t strSQL("SELECT [TURNO] FROM [JUEGOS] WHERE [CODIGO]=");

strSQL += code;

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

pRstAuthors->MoveFirst();

if (!pRstAuthors->EndOfFile)

{

while (!pRstAuthors->EndOfFile)

{

valField1 = pRstAuthors->Fields->GetItem("TURNO")->Value;

std::string disponible(valField1);

std::string::size\_type sz;

bool newturno = (bool)std::stoi(disponible, &sz);

if (!puedeTirar || turno != newturno)

{

if (!ActualizarTirosVzPlayer(newturno))

{

pressKey();

AsyncTurno = false;

}

}

pRstAuthors->MoveNext();

}

}

}

catch (\_com\_error & ce)

{

}

CoUninitialize();

}

else { Sleep(2000); }

}

return 0;

}

int wherex()

{

CONSOLE\_SCREEN\_BUFFER\_INFO csbi;

if (!GetConsoleScreenBufferInfo(

GetStdHandle(STD\_OUTPUT\_HANDLE),

&csbi

))

return -1;

return csbi.dwCursorPosition.X;

}

int wherey()

{

CONSOLE\_SCREEN\_BUFFER\_INFO csbi;

if (!GetConsoleScreenBufferInfo(

GetStdHandle(STD\_OUTPUT\_HANDLE),

&csbi

))

return -1;

return csbi.dwCursorPosition.Y;

}

bool SeAutoGano(int results[8])

{

for (int i = 0; i <= 6; i++)

{

if (results[i] != 0 && results[i + 1] != 0 && results[i + 2] != 0 && (i == 0 || i == 3 || i == 6))

{

if (1 == results[i] && 1 == results[i + 1] && 1 == results[i + 2])

{

return true;

}

if (2 == results[i] && 2 == results[i + 1] && 2 == results[i + 2])

{

return true;

}

}

if (i == 0)

{

if (results[i] != 0 && results[i + 4] != 0 && results[i + 8] != 0)

{

if (1 == results[i] && 1 == results[i + 4] && 1 == results[i + 8])

{

return true;

}

if (2 == results[i] && 2 == results[i + 4] && 2 == results[i + 8])

{

return true;

}

}

if (results[i] != 0 && results[i + 3] != 0 && results[i + 6] != 0)

{

if (1 == results[i] && 1 == results[i + 3] && 1 == results[i + 6])

{

return true;

}

if (2 == results[i] && 2 == results[i + 3] && 2 == results[i + 6])

{

return true;

}

}

}

if (i == 1)

{

if (results[i] != 0 && results[i + 3] != 0 && results[i + 6] != 0)

{

if (1 == results[i] && 1 == results[i + 3] && 1 == results[i + 6])

{

return true;

}

if (2 == results[i] && 2 == results[i + 3] && 2 == results[i + 6])

{

return true;

}

}

}

if (i == 2)

{

if (results[i] != 0 && results[i + 2] != 0 && results[i + 4] != 0)

{

if (1 == results[i] && 1 == results[i + 2] && 1 == results[i + 4])

{

return true;

}

if (2 == results[i] && 2 == results[i + 2] && 2 == results[i + 4])

{

return true;

}

}

if (results[i] != 0 && results[i + 3] != 0 && results[i + 6] != 0)

{

if (1 == results[i] && 1 == results[i + 3] && 1 == results[i + 6])

{

return true;

}

if (2 == results[i] && 2 == results[i + 3] && 2 == results[i + 6])

{

return true;

}

}

}

}

return false;

}

void VzCPU()

{

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

{

casillas[i] = 0;

}

bool turno = true;

int tecla = 0;

int seleccion = 0;

int tempx = 0;

int temp = 0;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0);

PrintColorXy("%s", " \n", 31, 7, 2, 15);

PrintColorXy("%s", " \n", 31, 8);

PrintColorXy("%s", " \n", 31, 9);

PrintColorXy("%s", " \n", 31, 10);

PrintColorXy("%s", " \n", 31, 11);

do

{

tecla = 0;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno ? 9 : 15), (turno ? 15 : 0));

PrintColorXy("%s", (turno ? "\xBA>>>>>>>>>>>>TU TURNO<<<<<<<<<<<<\xBA" : "\xBA>>>>>>>>>>>>MI TURNO<<<<<<<<<<<<\xBA"), 30, 31, (turno ? 9 : 15), (turno ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno ? 9 : 15), (turno ? 15 : 0));

tecla = \_getch();

if (tecla == FLECHA) {

tecla = \_getch();

switch (tecla)

{

case ARRIBA:

if (seleccion == 0 || seleccion == 1 || seleccion == 2)

{

seleccion += 6;

}

else

{

seleccion -= 3;

}

break;

case DERECHA:

if (seleccion == 5 || seleccion == 2 || seleccion == 8)

{

seleccion -= 2;

}

else

{

seleccion++;

}

break;

case ABAJO:

if (seleccion == 6 || seleccion == 7 || seleccion == 8)

{

seleccion -= 6;

}

else

{

seleccion += 3;

}

break;

case IZQUIERDA:

if (seleccion == 6 || seleccion == 3 || seleccion == 0)

{

seleccion += 2;

}

else

{

seleccion--;

}

break;

}

}

else

{

if (tecla == ENTER)

{

if (casillas[seleccion] == 0 && turno)

{

casillas[seleccion] = 1;

turno = false;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno ? 9 : 15), (turno ? 15 : 0));

PrintColorXy("%s", (turno ? "\xBA>>>>>>>>>>>>TU TURNO<<<<<<<<<<<<\xBA" : "\xBA>>>>>>>>>>>>MI TURNO<<<<<<<<<<<<\xBA"), 30, 31, (turno ? 9 : 15), (turno ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno ? 9 : 15), (turno ? 15 : 0));

Sleep(2000);

if (!TiraCpu())

{

MessageBoxA(NULL, "El tablero esta lleno", "Atención!", MB\_OK);

return;

}

else

{

turno = true;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno ? 9 : 15), (turno ? 15 : 0));

PrintColorXy("%s", (turno ? "\xBA>>>>>>>>>>>>TU TURNO<<<<<<<<<<<<\xBA" : "\xBA>>>>>>>>>>>>MI TURNO<<<<<<<<<<<<\xBA"), 30, 31, (turno ? 9 : 15), (turno ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno ? 9 : 15), (turno ? 15 : 0));

}

}

else

{

MessageBoxA(NULL, "Casilla Ocupada", "Atención!", MB\_OK);

}

}

else

{

if (tecla == ESC)

{

return;

}

}

}

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

{

switch (i)

{

case 0:

tempx = 0;

break;

case 1:

tempx = 11;

break;

case 2:

tempx = 22;

break;

case 3:

tempx = 0;

break;

case 4:

tempx = 11;

break;

case 5:

tempx = 22;

break;

case 6:

tempx = 0;

break;

case 7:

tempx = 11;

break;

case 8:

tempx = 22;

break;

}

if (i >= 6)

{

temp = 13;

}

else

{

if (i <= 2)

{

temp = 1;

}

else

{

if (i >= 3 && i <= 5)

{

temp = 7;

}

}

}

switch (casillas[i])

{

case 0:

PrintColorXy("%s", " \n", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", " \n", (31 + tempx), 7 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 8 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 10 + temp);

break;

case 1:

PrintColorXy("%s", " \\ \\ / /", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", " \\ V / ", (31 + tempx), 7 + temp);

PrintColorXy("%s", " > < ", (31 + tempx), 8 + temp);

PrintColorXy("%s", " / . \\ ", (31 + tempx), 9 + temp);

PrintColorXy("%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp);

break;

case 2:

PrintColorXy("%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", "| | | |", (31 + tempx), 7 + temp);

PrintColorXy("%s", "| | | |", (31 + tempx), 8 + temp);

PrintColorXy("%s", "| `--' |", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp);

break;

}

}

} while (Gano(0));

}

bool TiraCpu()

{

if (casillas[7] == 0)

{

casillas[7] = 2;

return true;

}

if (Bloquear())

{

return true;

}

if (TiroParaGanar())

{

return true;

}

return TiroDisponible();

}

bool Bloquear()

{

for (int i = 0; i <= 5; i++)

{

if (i == 0 || i == 3 || i == 6)

{

if (casillas[i] != 0 && casillas[i + 1] != 0 && casillas[i + 2] == 0)

{

if (1 == casillas[i] && 1 == casillas[i + 1] && casillas[i + 2] == 0)

{

casillas[i + 2] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 1] == 0 && casillas[i + 2] != 0)

{

if (1 == casillas[i] && casillas[i + 1] == 0 && 1 == casillas[i + 2])

{

casillas[i + 1] = 2;

return true;

}

}

if (casillas[i] == 0 && casillas[i + 1] != 0 && casillas[i + 2] != 0)

{

if (casillas[i] == 0 && 1 == casillas[i + 1] && 1 == casillas[i + 2])

{

casillas[i] = 2;

return true;

}

}

}

if (i == 0)

{

if (casillas[i] != 0 && casillas[i + 4] != 0 && casillas[i + 8] == 0)

{

if (1 == casillas[i] && 1 == casillas[i + 4] && casillas[i + 8] == 0)

{

casillas[i + 8] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 4] == 0 && casillas[i + 8] != 0)

{

if (1 == casillas[i] && casillas[i + 4] == 0 && 1 == casillas[i + 8])

{

casillas[i + 4] = 2;

return true;

}

}

if (casillas[i] == 0 && casillas[i + 4] != 0 && casillas[i + 8] != 0)

{

if (casillas[i] == 0 && 1 == casillas[i + 4] && 1 == casillas[i + 8])

{

casillas[i] = 2;

return true;

}

}

if (casillas[i + 6] == 0 && casillas[i + 7] != 0 && casillas[i + 8] != 0)

{

if (casillas[i + 6] == 0 && 1 == casillas[i + 7] && 1 == casillas[i + 8])

{

casillas[i + 6] = 2;

return true;

}

}

if (casillas[i + 6] != 0 && casillas[i + 7] == 0 && casillas[i + 8] != 0)

{

if (1 == casillas[i + 6] && casillas[i + 7] == 0 && 1 == casillas[i + 8])

{

casillas[i + 7] = 2;

return true;

}

}

if (casillas[i + 6] != 0 && casillas[i + 7] != 0 && casillas[i + 8] == 0)

{

if (1 == casillas[i + 6] && 1 == casillas[i + 4] && casillas[i + 8] == 0)

{

casillas[i + 8] = 2;

return true;

}

}

}

if (i == 1)

{

if (casillas[i] != 0 && casillas[i + 3] != 0 && casillas[i + 6] == 0)

{

if (1 == casillas[i] && 1 == casillas[i + 3] && casillas[i + 6] == 0)

{

casillas[i + 6] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 3] == 0 && casillas[i + 6] != 0)

{

casillas[i + 3] = 2;

return true;

}

if (casillas[i] == 0 && casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (casillas[i] == 0 && 1 == casillas[i + 3] && 1 == casillas[i + 6])

{

casillas[i] = 2;

return true;

}

}

}

if (i == 2)

{

if (casillas[i] != 0 && casillas[i + 2] != 0 && casillas[i + 4] == 0)

{

if (1 == casillas[i] && 1 == casillas[i + 2] && casillas[i + 4] == 0)

{

casillas[i + 4] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 2] == 0 && casillas[i + 4] != 0)

{

if (1 == casillas[i] && casillas[i + 2] == 0 && 1 == casillas[i + 4])

{

casillas[i + 2] = 2;

return true;

}

}

if (casillas[i] == 0 && casillas[i + 2] != 0 && casillas[i + 4] != 0)

{

if (casillas[i] == 0 && 1 == casillas[i + 2] && 1 == casillas[i + 4])

{

casillas[i] = 2;

return true;

}

}

}

if (i == 0 || i == 1 || i == 2)

{

if (casillas[i] != 0 && casillas[i + 3] != 0 && casillas[i + 6] == 0)

{

if (2 == casillas[i] && 2 == casillas[i + 3] && casillas[i + 6] == 0)

{

casillas[i + 6] = 2;

return true;

}

if (1 == casillas[i] && 1 == casillas[i + 3] && casillas[i + 6] == 0)

{

casillas[i + 6] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 3] == 0 && casillas[i + 6] != 0)

{

if (1 == casillas[i] && casillas[i + 3] == 0 && 1 == casillas[i + 6])

{

casillas[i + 3] = 2;

return true;

}

}

if (casillas[i] == 0 && casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (casillas[i] == 0 && 1 == casillas[i + 3] && 1 == casillas[i + 6])

{

casillas[i] = 2;

return true;

}

}

}

if (i == 5 || i == 2)

{

if (casillas[i + 1] != 0 && casillas[i + 2] != 0 && casillas[i + 3] == 0)

{

if (1 == casillas[i + 1] && 1 == casillas[i + 2])

{

casillas[i + 3] = 2;

return true;

}

}

if (casillas[i + 1] != 0 && casillas[i + 2] == 0 && casillas[i + 3] != 0)

{

if (1 == casillas[i + 1] && 1 == casillas[i + 3])

{

casillas[i + 2] = 2;

return true;

}

}

if (casillas[i + 1] == 0 && casillas[i + 2] != 0 && casillas[i + 3] != 0)

{

if (1 == casillas[i + 2] && 1 == casillas[i + 3])

{

casillas[i + 1] = 2;

return true;

}

}

}

}

if (casillas[0] == 0)

{

casillas[0] = 2;

return true;

}

if (casillas[2] == 0)

{

casillas[2] = 2;

return true;

}

if (casillas[6] == 0)

{

casillas[6] = 2;

return true;

}

if (casillas[8] == 0)

{

casillas[8] = 2;

return true;

}

return false;

}

bool TiroParaGanar()

{

for (int i = 0; i <= 6; i++)

{

if (casillas[i] != 0 && casillas[i + 1] != 0)

{

if (2 == casillas[i] && 2 == casillas[i + 1] && casillas[i + 2] == 0)

{

casillas[i + 2] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 2] != 0)

{

if (2 == casillas[i] && casillas[i + 1] == 0 && 2 == casillas[i + 2])

{

casillas[i + 1] = 2;

return true;

}

}

if (casillas[i + 1] != 0 && casillas[i + 2] != 0)

{

if (casillas[i] == 0 && 2 == casillas[i + 1] && 2 == casillas[i + 2])

{

casillas[i] = 2;

return true;

}

}

if (i == 0)

{

if (casillas[i] != 0 && casillas[i + 4] != 0)

{

if (2 == casillas[i] && 2 == casillas[i + 4] && casillas[i + 8] == 0)

{

casillas[i + 8] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 8] != 0)

{

if (2 == casillas[i] && casillas[i + 4] == 0 && 2 == casillas[i + 8])

{

casillas[i + 4] = 2;

return true;

}

}

if (casillas[i + 4] != 0 && casillas[i + 8] != 0)

{

if (casillas[i] == 0 && 2 == casillas[i + 4] && 2 == casillas[i + 8])

{

casillas[i] = 2;

return true;

}

}

}

if (i == 1)

{

if (casillas[i] != 0 && casillas[i + 3] != 0)

{

if (2 == casillas[i] && 2 == casillas[i + 3] && casillas[i + 6] == 0)

{

casillas[i + 6] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 6] != 0)

{

if (2 == casillas[i] && casillas[i + 3] == 0 && 2 == casillas[i + 6])

{

casillas[i + 3] = 2;

return true;

}

}

if (casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (casillas[i] == 0 && 2 == casillas[i + 3] && 2 == casillas[i + 6])

{

casillas[i] = 2;

return true;

}

}

}

if (i == 2)

{

if (casillas[i] != 0 && casillas[i + 2] != 0)

{

if (2 == casillas[i] && 2 == casillas[i + 2] && casillas[i + 4] == 0)

{

casillas[i + 4] = 2;

return true;

}

}

if (casillas[i] != 0 && casillas[i + 4] != 0)

{

if (2 == casillas[i] && casillas[i + 2] == 0 && 2 == casillas[i + 4])

{

casillas[i + 2] = 2;

return true;

}

}

if (casillas[i + 2] != 0 && casillas[i + 4] != 0)

{

if (casillas[i] == 0 && 2 == casillas[i + 2] && 2 == casillas[i + 4])

{

casillas[i] = 2;

return true;

}

}

}

//if (casillas[i] !=0 && casillas[i + 3] !=0 && casillas[i + 6] !=0)

//{

// if (2==casillas[i] && 2==casillas[i + 3] && 1==casillas[i + 6])

// {

// casillas[i + 6]=2;

// return true;

// }

// if (2==casillas[i] && 1==casillas[i + 3] && 2==casillas[i + 6])

// {

// casillas[i + 3]=2;

// return true;

// }

// if (1==casillas[i] && 2==casillas[i + 3] && 2==casillas[i + 6])

// {

// casillas[i]=2;

// return true;

// }

//}

}

return false;

}

bool TiroDisponible()

{

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

{

if (casillas[i] == 0)

{

casillas[i] = 2;

return true;

}

}

return false;

}

bool Gano(int modo)

{

for (int i = 0; i <= 6; i++)

{

if (casillas[i] != 0 && casillas[i + 1] != 0 && casillas[i + 2] != 0 && (i == 0 || i == 3 || i == 6))

{

if (1 == casillas[i] && 1 == casillas[i + 1] && 1 == casillas[i + 2])

{

if (modo == 0)

{

MessageBoxA(NULL, "Venciste al Cpu :)", "Felicidades", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (X)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 1] && 2 == casillas[i + 2])

{

if (modo == 0)

{

MessageBoxA(NULL, "Gana Cpu :(", "Lastima...", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (O)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

}

if (i == 0)

{

if (casillas[i] != 0 && casillas[i + 4] != 0 && casillas[i + 8] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 4] && 1 == casillas[i + 8])

{

if (modo == 0)

{

MessageBoxA(NULL, "Venciste al Cpu :)", "Felicidades", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (X)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 4] && 2 == casillas[i + 8])

{

if (modo == 0)

{

MessageBoxA(NULL, "Gana Cpu :(", "Lastima...", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (O)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

}

if (casillas[i] != 0 && casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 3] && 1 == casillas[i + 6])

{

if (modo == 0)

{

MessageBoxA(NULL, "Venciste al Cpu :)", "Felicidades", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (X)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 3] && 2 == casillas[i + 6])

{

if (modo == 0)

{

MessageBoxA(NULL, "Gana Cpu :(", "Lastima...", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (O)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

}

}

if (i == 1)

{

if (casillas[i] != 0 && casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 3] && 1 == casillas[i + 6])

{

if (modo == 0)

{

MessageBoxA(NULL, "Venciste al Cpu :)", "Felicidades", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (X)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 3] && 2 == casillas[i + 6])

{

if (modo == 0)

{

MessageBoxA(NULL, "Venciste al Cpu :)", "Felicidades", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (O)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

}

}

if (i == 2)

{

if (casillas[i] != 0 && casillas[i + 2] != 0 && casillas[i + 4] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 2] && 1 == casillas[i + 4])

{

if (modo == 0)

{

MessageBoxA(NULL, "Venciste al Cpu :)", "Felicidades", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (X)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 2] && 2 == casillas[i + 4])

{

if (modo == 0)

{

MessageBoxA(NULL, "Gana Cpu :(", "Lastima...", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (O)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

}

if (casillas[i] != 0 && casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 3] && 1 == casillas[i + 6])

{

if (modo == 0)

{

MessageBoxA(NULL, "Venciste al Cpu :)", "Felicidades", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (X)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 3] && 2 == casillas[i + 6])

{

if (modo == 0)

{

MessageBoxA(NULL, "Gana Cpu :(", "Lastima...", MB\_OK);

conectado = true; return false;

}

if (modo == 1)

{

MessageBoxA(NULL, "Gana: (O)", "Felicidades", MB\_OK);

conectado = true; return false;

}

}

}

}

}

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

{

if (casillas[i] == 0)

{

conectado = true; return true;

}

}

MessageBoxA(NULL, "El tablero esta lleno...", "Empate!", MB\_OK);

conectado = true; return true;

}

bool SilenceGano(int modo)

{

for (int i = 0; i <= 6; i++)

{

if (casillas[i] != 0 && casillas[i + 1] != 0 && casillas[i + 2] != 0 && (i == 0 || i == 3 || i == 6))

{

if (1 == casillas[i] && 1 == casillas[i + 1] && 1 == casillas[i + 2])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 1] && 2 == casillas[i + 2])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

}

if (i == 0)

{

if (casillas[i] != 0 && casillas[i + 4] != 0 && casillas[i + 8] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 4] && 1 == casillas[i + 8])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 4] && 2 == casillas[i + 8])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

}

if (casillas[i] != 0 && casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 3] && 1 == casillas[i + 6])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 3] && 2 == casillas[i + 6])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

}

}

if (i == 1)

{

if (casillas[i] != 0 && casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 3] && 1 == casillas[i + 6])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 3] && 2 == casillas[i + 6])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

}

}

if (i == 2)

{

if (casillas[i] != 0 && casillas[i + 2] != 0 && casillas[i + 4] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 2] && 1 == casillas[i + 4])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 2] && 2 == casillas[i + 4])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

}

if (casillas[i] != 0 && casillas[i + 3] != 0 && casillas[i + 6] != 0)

{

if (1 == casillas[i] && 1 == casillas[i + 3] && 1 == casillas[i + 6])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

if (2 == casillas[i] && 2 == casillas[i + 3] && 2 == casillas[i + 6])

{

if (modo == 0)

{

return false;

}

if (modo == 1)

{

return false;

}

}

}

}

}

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

{

if (casillas[i] == 0)

{

return true;

}

}

MessageBoxA(NULL, "El tablero esta lleno...", "Empate!", MB\_OK);

return true;

}

void VzFriend()

{

turno = true;

for (int i = 0; i <= 20; i++)

{

apodo[i] = ' ';

}

contrincante = "Jugador 2";

int tecla = 0;

int i = 0;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 25, 15, 0);

PrintColorXy("%s", "\xBA", 30, 26);

PrintColorXy("%s", " ", 31, 26, 0, 15);

PrintColorXy("%s", "\xBA", 64, 26, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 27, 15, 0);

PrintColorXy("%s", "Escribe un apodo original ;)", 30, 23, 15, 0);

while (tecla != ENTER)

{

tecla = toupper((char)\_getch());

if (tecla == ESC)

{

return;

}

if (tecla != ENTER&&tecla != ESC&&tecla != FLECHA)

{

apodo[i] = (char)tecla;

i++;

PrintColorXy("%s", apodo, 31, 26, 0, 15);

}

}

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 25, 15, 0);

PrintColorXy("%s", "\xBA", 30, 26);

PrintColorXy("%s", " ", 31, 26, 0, 15);

PrintColorXy("%s", "\xBA", 64, 26, 15, 0);

PrintColorXy("%s", "\xBA", 30, 27);

PrintColorXy("%s", ">>Conectando con el servidor...<<", 31, 27, 0, 15);

PrintColorXy("%s", "\xBA", 64, 27, 15, 0);

PrintColorXy("%s", "\xBA", 30, 28);

PrintColorXy("%s", " ", 31, 28, 0, 15);

PrintColorXy("%s", "\xBA", 64, 28, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 29, 15, 0);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0);

AsyncAnimation = true;

char\* TempCode = "";

char code[100] = "";

TempCode = appendCharToCharArray(TempCode, (char)RandRange(65, 90));

sprintf(code, "%d", RandRange(0, 9));

TempCode = appendCharToCharArray(TempCode, code[0]);

TempCode = appendCharToCharArray(TempCode, (char)RandRange(65, 90));

sprintf(code, "%d", RandRange(0, 9));

TempCode = appendCharToCharArray(TempCode, code[0]);

ErrorAlConectarResult = false;

while (ExisteCodigo(TempCode) && !ErrorAlConectarResult)

{

TempCode = "";

TempCode = appendCharToCharArray(TempCode, (char)RandRange(65, 90));

sprintf(code, "%d", RandRange(0, 9));

TempCode = appendCharToCharArray(TempCode, code[0]);

TempCode = appendCharToCharArray(TempCode, (char)RandRange(65, 90));

sprintf(code, "%d", RandRange(0, 9));

TempCode = appendCharToCharArray(TempCode, code[0]);

}

if (ErrorAlConectarResult)

{

return;

}

PrintColorXy("%s", ">>Obteniendo Codigo de Jugador.<<", 31, 27, 0, 15);

codigo[0] = TempCode[0];

codigo[1] = TempCode[1];

codigo[2] = TempCode[2];

codigo[3] = TempCode[3];

ErrorAlConectarResult = false;

Escribir(codigo, apodo);

if (ErrorAlConectarResult) {

return;

}

AsyncAnimation = false;

Sleep(10);

RedibujarMarcos();

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 12, 15, 0);

PrintColorXy("%s", "\xBA", 30, 13);

PrintColorXy("%s", " ", 31, 13, 0, 15);

PrintColorXy("%s", "\xBA", 64, 13, 15, 0);

PrintColorXy("%s", "\xBA", 30, 14);

PrintColorXy(">>Tu codigo de jugador es:%s <<", codigo, 31, 14, 0, 15);

PrintColorXy("%s", "\xBA", 64, 14, 15, 0);

PrintColorXy("%s", "\xBA", 30, 15);

PrintColorXy("%s", " ", 31, 15, 0, 15);

PrintColorXy("%s", "\xBA", 64, 15, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 16, 15, 0);

PrintColorXy("%s", "Comparte tu codigo con un amigo o pide que te lo compartan y escribelo aqui.", 11, 27, 15, 0);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 20, 15, 0);

PrintColorXy("%s", "\xBA", 30, 21);

PrintColorXy("%s", " ", 31, 21, 0, 15);

PrintColorXy("%s", "\xBA", 64, 21, 15, 0);

PrintColorXy("%s", "\xBA", 30, 22);

PrintColorXy("-------------->%s<--------------", codigoConectar, 31, 22, 0, 15);

PrintColorXy("%s", "\xBA", 64, 22, 15, 0);

PrintColorXy("%s", "\xBA", 30, 23);

PrintColorXy("%s", " ", 31, 23, 0, 15);

PrintColorXy("%s", "\xBA", 64, 23, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0);

AsyncCodigo = true;

i = 0;

while (!conectado)

{

tecla = \_getch();

if (tecla != ESC)

{

codigoConectar[i] = toupper((char)tecla);

PrintColorXy("-------------->%s<-------------", codigoConectar, 31, 22, 0, 15);

if (i < 3)

{

i++;

}

else {

i = 0;

string codigoConectarStd(codigoConectar);

string codigoStd(codigo);

if (codigoConectarStd == codigoStd)

{

MessageBoxA(NULL, "No puedes usar tu mismo codigo, para conectarte con un amigo deben intercambiar sus codigos...", "Atención!", MB\_OK);

PrintColorXy("-------------->%s<-------------", "XXXX", 31, 22, 12, 15);

codigoConectar[0] = 'X';

codigoConectar[1] = 'X';

codigoConectar[2] = 'X';

codigoConectar[3] = 'X';

}

else

{

ErrorAlConectarResult = false;

if (Conectar(codigoConectar))

{

MITURNO = false;

AsyncCodigo = false;

Sleep(20);

RedibujarMarcos();

CargarCasillasMultiplayer(apodo, contrincante);

}

else

{

if (ErrorAlConectarResult) {

return;

}

PrintColorXy("-------------->%s<-------------", codigoConectar, 31, 22, 0, 15);

codigoConectar[0] = 'X';

codigoConectar[1] = 'X';

codigoConectar[2] = 'X';

codigoConectar[3] = 'X';

MessageBoxA(NULL, "Codigo Incorrecto!", "Atención!", MB\_OK);

}

}

}

}

else

{

AsyncCodigo = false;

return;

}

}

AsyncCodigo = false;

Sleep(20);

RedibujarMarcos();

CargarCasillasMultiplayer(apodo, contrincante);

}

bool Escribir(char\* codigo, char\* apodo, char disponible, char turno, char buscando, char\* casillas)

{

\_bstr\_t strCodigo;

\_bstr\_t strAge;

\_bstr\_t strDOB;

\_bstr\_t strSalary;

\_bstr\_t strBuscando;

\_bstr\_t strCasillas;

\_ConnectionPtr pConn = NULL;

\_bstr\_t strCon(CadenaCon);

HRESULT hr = S\_OK;

CoInitialize(NULL);

try

{

hr = pConn.CreateInstance((\_\_uuidof(Connection)));

if (FAILED(hr))

{

CoUninitialize();

return false;

}

hr = pConn->Open(strCon, "", "", 0);

if (FAILED(hr))

{

CoUninitialize();

return false;

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

strCodigo = result;

char result2[100] = "";

strcpy(result2, ",\'");

strcat(result2, apodo);

strcat(result2, "\',");

strAge = result2;

char\* result3 = "";

result3 = appendCharToCharArray(result3, turno);

strcat(result3, ",");

strDOB = result3;

char\* result4 = "";

result4 = appendCharToCharArray(result4, disponible);

strcat(result4, ",");

strSalary = result4;

char\* result5 = "";

result5 = appendCharToCharArray(result5, buscando);

strcat(result5, ",");

strBuscando = result5;

char result6[100] = "";

strcpy(result6, "\'");

strcat(result6, casillas);

strcat(result6, "\')");

strCasillas = result6;

\_bstr\_t strSQL("INSERT INTO JUEGOS(CODIGO,APODO,TURNO,DISPONIBLE,BUSCANDO,CASILLAS) Values(");

strSQL += strCodigo + strAge + strDOB + strSalary + strBuscando + strCasillas;

pConn->Execute(strSQL, NULL, adExecuteNoRecords);

pConn->Close();

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return false;

}

CoUninitialize();

return true;

}

char\* appendCharToCharArray(char\* array, char a)

{

size\_t len = strlen(array);

char\* ret = new char[len + 2];

strcpy(ret, array);

ret[len] = a;

ret[len + 1] = '\0';

return ret;

}

bool ExisteCodigo(char\* codigo) {

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t strCnn(CadenaCon);

\_RecordsetPtr pRstAuthors = NULL;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr)) //Comprueba que se pueda crear la instancia

{

return false;

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

code = result;

\_bstr\_t strSQL("SELECT [CODIGO] FROM [JUEGOS] WHERE [CODIGO]=");

strSQL += code;

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

\_bstr\_t valField1;

pRstAuthors->MoveFirst();

if (!pRstAuthors->EndOfFile)

{

while (!pRstAuthors->EndOfFile)

{

valField1 = pRstAuthors->Fields->GetItem("CODIGO")->Value;

return true;

pRstAuthors->MoveNext();

}

}

else

{

return false;

}

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return false;

}

CoUninitialize();

return false;

}

bool Conectar(char\* codigo)

{

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t casillaR;

\_bstr\_t contrincanteR;

\_bstr\_t strCnn(CadenaCon);

\_RecordsetPtr pRstAuthors = NULL;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr))

{

return false;

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

code = result;

\_bstr\_t strSQL("SELECT [CASILLAS],[APODO] FROM [JUEGOS] WHERE DISPONIBLE=1 AND [CODIGO]=");

strSQL += code;

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

\_bstr\_t valField1;

pRstAuthors->MoveFirst();

if (!pRstAuthors->EndOfFile)

{

while (!pRstAuthors->EndOfFile)

{

casillaR = pRstAuthors->Fields->GetItem("CASILLAS")->Value;

contrincanteR = pRstAuthors->Fields->GetItem("APODO")->Value;

static std::string str(contrincanteR);

static const char \*cstr = str.c\_str();

contrincante = (char\*)cstr;

ErrorAlConectarResult = false;

Ocupar(code);

if (ErrorAlConectar) {

return false;

}

codigo = codigoConectar;

return true;

pRstAuthors->MoveNext();

}

}

else

{

return false;

}

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return false;

}

CoUninitialize();

return false;

}

void Ocupar(\_bstr\_t code)

{

\_ConnectionPtr pConn = NULL;

\_bstr\_t strCon(CadenaCon);

\_bstr\_t CodigoActual(codigo);

HRESULT hr = S\_OK;

CoInitialize(NULL);

try

{

hr = pConn.CreateInstance((\_\_uuidof(Connection)));

if (FAILED(hr))

{

CoUninitialize();

return;

}

hr = pConn->Open(strCon, "", "", 0);

if (FAILED(hr))

{

CoUninitialize();

return;

}

\_bstr\_t strSQL("UPDATE JUEGOS SET [DISPONIBLE]=0,[BUSCANDO]=0,FECHA=NULL WHERE [CODIGO]=");

strSQL += (code + ";DELETE FROM JUEGOS WHERE [CODIGO]='" + codigo + "'");

pConn->Execute(strSQL, NULL, adExecuteNoRecords);

pConn->Close();

char \*myCharArray = NULL;

myCharArray = \_com\_util::ConvertBSTRToString(code);

::SysFreeString(code);

codigo[0] = myCharArray[1];

codigo[1] = myCharArray[2];

codigo[2] = myCharArray[3];

codigo[3] = myCharArray[4];

return;

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return;

}

CoUninitialize();

return;

}

bool Ocupado(char\* codigo)

{

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t strCnn(CadenaCon);

\_RecordsetPtr pRstAuthors = NULL;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr))

{

return false;

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

code = result;

\_bstr\_t strSQL("SELECT [DISPONIBLE], FROM [JUEGOS] WHERE [CODIGO]=");

strSQL += code;

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

\_bstr\_t valField1;

pRstAuthors->MoveFirst();

if (!pRstAuthors->EndOfFile)

{

while (!pRstAuthors->EndOfFile)

{

valField1 = pRstAuthors->Fields->GetItem("DISPONIBLE")->Value;

std::string disponible(valField1);

if (disponible.compare("0") == 0)

{

MITURNO = true;

return true;

}

else

{

return false;

}

pRstAuthors->MoveNext();

}

}

else

{

return false;

}

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return false;

}

CoUninitialize();

return false;

}

int pressKey()

{

//SetForegroundWindow(hWnd);

SetCursor(hWnd, 0, 0);

SendString(hWnd, "a");

return 0;

}

void CargarCasillasMultiplayer(char\* apodo, char\* contrincante, bool aleatorio)

{

int tecla = 0;

int seleccion = 0;

int tempx = 0;

int temp = 0;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0);

PrintColorXy("%s", " \n", 31, 7, 2, 15);

PrintColorXy("%s", " \n", 31, 8);

PrintColorXy("%s", " \n", 31, 9);

PrintColorXy("%s", " \n", 31, 10);

PrintColorXy("%s", " \n", 31, 11);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy("%s", "\xBA", 31, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy(" TURNO DE:%s", (turno == MITURNO ? apodo : contrincante), 30, 31, 15, 0);

PrintColorXy("%s", "\xBA", 63, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

do

{

tecla = 0;

AsyncTurno = true;

tecla = \_getch();

if (tecla == FLECHA) {

tecla = \_getch();

switch (tecla)

{

case ARRIBA:

if (seleccion == 0 || seleccion == 1 || seleccion == 2)

{

seleccion += 6;

}

else

{

seleccion -= 3;

}

break;

case DERECHA:

if (seleccion == 5 || seleccion == 2 || seleccion == 8)

{

seleccion -= 2;

}

else

{

seleccion++;

}

break;

case ABAJO:

if (seleccion == 6 || seleccion == 7 || seleccion == 8)

{

seleccion -= 6;

}

else

{

seleccion += 3;

}

break;

case IZQUIERDA:

if (seleccion == 6 || seleccion == 3 || seleccion == 0)

{

seleccion += 2;

}

else

{

seleccion--;

}

break;

}

}

else

{

if (tecla == ENTER)

{

if (turno != MITURNO || !puedeTirar)

{

MessageBoxA(NULL, "No es tu turno...", "Atención!", MB\_OK);

}

else

{

if (casillas[seleccion] == 0)

{

casillas[seleccion] = (MITURNO ? 1 : 2);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, 15, 0);

PrintColorXy("%s", "\xBA", 31, 31, 15, 0);

PrintColorXy(" TURNO DE:%s", contrincante, 30, 31, 15, 0);

PrintColorXy("%s", "\xBA", 63, 31, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, 15, 0);

ErrorAlConectarResult = false;

EnviarTiro();

if (ErrorAlConectarResult) {

return;

}

puedeTirar = false;

}

else

{

MessageBoxA(NULL, "Casilla Ocupada", "Atención!", MB\_OK);

}

}

}

else

{

if (tecla == ESC)

{

return;

}

}

}

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

{

switch (i)

{

case 0:

tempx = 0;

break;

case 1:

tempx = 11;

break;

case 2:

tempx = 22;

break;

case 3:

tempx = 0;

break;

case 4:

tempx = 11;

break;

case 5:

tempx = 22;

break;

case 6:

tempx = 0;

break;

case 7:

tempx = 11;

break;

case 8:

tempx = 22;

break;

}

if (i >= 6)

{

temp = 13;

}

else

{

if (i <= 2)

{

temp = 1;

}

else

{

if (i >= 3 && i <= 5)

{

temp = 7;

}

}

}

switch (casillas[i])

{

case 0:

PrintColorXy("%s", " \n", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", " \n", (31 + tempx), 7 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 8 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 10 + temp);

break;

case 1:

PrintColorXy("%s", " \\ \\ / /", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", " \\ V / ", (31 + tempx), 7 + temp);

PrintColorXy("%s", " > < ", (31 + tempx), 8 + temp);

PrintColorXy("%s", " / . \\ ", (31 + tempx), 9 + temp);

PrintColorXy("%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp);

break;

case 2:

PrintColorXy("%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", "| | | |", (31 + tempx), 7 + temp);

PrintColorXy("%s", "| | | |", (31 + tempx), 8 + temp);

PrintColorXy("%s", "| `--' |", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp);

break;

}

}

} while (Gano(1));

return;

}

void EnviarTiro(bool aleatorio)

{

std::string res = "'";

for each(int t in casillas)

{

res += ",";

res += std::to\_string(t);

}

res += "' WHERE [CODIGO]='";

if (!aleatorio)

{

res += codigo;

}

else

{

res += codigoConectar;

}

res += "'";

\_bstr\_t strCasillas;

\_ConnectionPtr pConn = NULL;

\_bstr\_t strCon(CadenaCon);

HRESULT hr = S\_OK;

CoInitialize(NULL);

try

{

hr = pConn.CreateInstance((\_\_uuidof(Connection)));

if (FAILED(hr))

{

CoUninitialize();

return;

}

hr = pConn->Open(strCon, "", "", 0);

if (FAILED(hr))

{

CoUninitialize();

return;

}

int wslen = ::MultiByteToWideChar(CP\_ACP, 0,

res.data(), res.length(),

NULL, 0);

BSTR wsdata = ::SysAllocStringLen(NULL, wslen);

::MultiByteToWideChar(CP\_ACP, 0,

res.data(), res.length(),

wsdata, wslen);

strCasillas = wsdata;

\_bstr\_t strSQL("UPDATE JUEGOS SET [TURNO]=~[TURNO],[CASILLAS]=");

strSQL += strCasillas;

pConn->Execute(strSQL, NULL, adExecuteNoRecords);

pConn->Close();

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return;

}

CoUninitialize();

return;

}

void SendString(HWND h, char \*text)

{

PostMessage(h, WM\_CHAR, text[0], 0);

}

void SetCursor(HWND h, short x, short y)

{

PostMessage(h, WM\_LBUTTONDOWN, MK\_LBUTTON, MAKELPARAM(x, y));

PostMessage(h, WM\_LBUTTONUP, MK\_LBUTTON, MAKELPARAM(x, y));

}

bool ActualizarTirosVzPlayer(bool newturno, bool aleatorio)

{

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t strCnn(CadenaCon);

\_RecordsetPtr pRstAuthors = NULL;

std::string::size\_type sz;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr))

{

}

char result[100] = "";

strcpy(result, "\'");

if (!aleatorio)

{

strcat(result, codigo);

}

else

{

strcat(result, codigoConectar);

}

strcat(result, "\'");

code = result;

\_bstr\_t strSQL("SELECT [CASILLAS] FROM [JUEGOS] WHERE [CODIGO]=");

strSQL += code;

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

\_bstr\_t valField1;

pRstAuthors->MoveFirst();

if (!pRstAuthors->EndOfFile)

{

while (!pRstAuthors->EndOfFile)

{

valField1 = pRstAuthors->Fields->GetItem("CASILLAS")->Value;

std::string disponible(valField1);

int i = 0;

for each(char c in disponible)

{

if (c == '0' || c == '1' || c == '2')

{

casillas[i] = (int)c - 48;

i++;

}

}

pRstAuthors->MoveNext();

}

}

}

catch (\_com\_error & ce)

{

}

CoUninitialize();

int tempx = 0;

int temp = 0;

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

{

switch (i)

{

case 0:

tempx = 0;

break;

case 1:

tempx = 11;

break;

case 2:

tempx = 22;

break;

case 3:

tempx = 0;

break;

case 4:

tempx = 11;

break;

case 5:

tempx = 22;

break;

case 6:

tempx = 0;

break;

case 7:

tempx = 11;

break;

case 8:

tempx = 22;

break;

}

if (i >= 6)

{

temp = 13;

}

else

{

if (i <= 2)

{

temp = 1;

}

else

{

if (i >= 3 && i <= 5)

{

temp = 7;

}

}

}

switch (casillas[i])

{

case 0:

PrintColorXy("%s", " \n", (31 + tempx), 6 + temp, 15, 0);

PrintColorXy("%s", " \n", (31 + tempx), 7 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 8 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 10 + temp);

break;

case 1:

PrintColorXy("%s", " \\ \\ / /", (31 + tempx), 6 + temp, 15, 0);

PrintColorXy("%s", " \\ V / ", (31 + tempx), 7 + temp);

PrintColorXy("%s", " > < ", (31 + tempx), 8 + temp);

PrintColorXy("%s", " / . \\ ", (31 + tempx), 9 + temp);

PrintColorXy("%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp);

break;

case 2:

PrintColorXy("%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, 15, 0);

PrintColorXy("%s", "| | | |", (31 + tempx), 7 + temp);

PrintColorXy("%s", "| | | |", (31 + tempx), 8 + temp);

PrintColorXy("%s", "| `--' |", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp);

break;

}

if (turno != newturno)

{

turno = newturno;

puedeTirar = true;

}

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy("%s", "\xBA", 30, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy("TURNO DE:%s", (turno == MITURNO ? apodo : contrincante), 31, 31, 15, 0);

PrintColorXy("%s", "\xBA", 63, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

}

return SilenceGano(1);

}

void vzAleatorio()

{

turno = true;

for (int i = 0; i <= 20; i++)

{

apodo[i] = ' ';

}

contrincante = "Jugador 2";

int tecla = 0;

int i = 0;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 25, 15, 0);

PrintColorXy("%s", "\xBA", 30, 26);

PrintColorXy("%s", " ", 31, 26, 0, 15);

PrintColorXy("%s", "\xBA", 64, 26, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 27, 15, 0);

PrintColorXy("%s", "Escribe un apodo original ;)", 30, 23, 15, 0);

while (tecla != ENTER) {

tecla = toupper((char)\_getch());

if (tecla == ESC)

{

return;

}

if (tecla != ENTER&&tecla != FLECHA&& tecla != ESC)

{

apodo[i] = (char)tecla;

i++;

PrintColorXy("%s", apodo, 31, 26, 0, 15);

}

}

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 25, 15, 0);

PrintColorXy("%s", "\xBA", 30, 26);

PrintColorXy("%s", " ", 31, 26, 0, 15);

PrintColorXy("%s", "\xBA", 64, 26, 15, 0);

PrintColorXy("%s", "\xBA", 30, 27);

PrintColorXy("%s", ">>Conectando con el servidor...<<", 31, 27, 0, 15);

PrintColorXy("%s", "\xBA", 64, 27, 15, 0);

PrintColorXy("%s", "\xBA", 30, 28);

PrintColorXy("%s", " ", 31, 28, 0, 15);

PrintColorXy("%s", "\xBA", 64, 28, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 29, 15, 0);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0);

AsyncAnimation = true;

char\* TempCode = "";

char code[100] = "";

TempCode = appendCharToCharArray(TempCode, (char)RandRange(65, 90));

sprintf(code, "%d", RandRange(0, 9));

TempCode = appendCharToCharArray(TempCode, code[0]);

TempCode = appendCharToCharArray(TempCode, (char)RandRange(65, 90));

sprintf(code, "%d", RandRange(0, 9));

TempCode = appendCharToCharArray(TempCode, code[0]);

while (ExisteCodigo(TempCode))

{

TempCode = "";

TempCode = appendCharToCharArray(TempCode, (char)RandRange(65, 90));

sprintf(code, "%d", RandRange(0, 9));

TempCode = appendCharToCharArray(TempCode, code[0]);

TempCode = appendCharToCharArray(TempCode, (char)RandRange(65, 90));

sprintf(code, "%d", RandRange(0, 9));

TempCode = appendCharToCharArray(TempCode, code[0]);

}

PrintColorXy("%s", ">>>>>Registrando Jugador<<<<<<", 31, 27, 0, 15);

codigo[0] = TempCode[0];

codigo[1] = TempCode[1];

codigo[2] = TempCode[2];

codigo[3] = TempCode[3];

ErrorAlConectarResult = false;

EscribirAleatorio(codigo, apodo);

if (ErrorAlConectarResult) {

return;

}

AsyncAnimation = false;

Sleep(10);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0);

AsyncAnimation = true;

AsyncAleatorioActive = true;

ErrorAlConectarResult = false;

while (!ConectarAleatorio(codigo) && !AleatorioOcupado(codigo) && !ErrorAlConectarResult)

{

tecla = \_getch();

if (tecla == ESC)

{

AsyncAleatorioActive = false;

return;

}

}

if (ErrorAlConectarResult) {

return;

}

AsyncAnimation = false;

AsyncAleatorioActive = false;

Sleep(20);

RedibujarMarcos();

CargarCasillasMultiplayerAleatorio(apodo, contrincante);

}

void EscribirAleatorio(char\* codigo, char\* apodo, char disponible, char turno, char buscando, char\* casillas)

{

\_bstr\_t strCodigo;

\_bstr\_t strAge;

\_bstr\_t strDOB;

\_bstr\_t strSalary;

\_bstr\_t strBuscando;

\_bstr\_t strCasillas;

\_ConnectionPtr pConn = NULL;

\_bstr\_t strCon(CadenaCon);

HRESULT hr = S\_OK;

CoInitialize(NULL);

try

{

hr = pConn.CreateInstance((\_\_uuidof(Connection)));

if (FAILED(hr))

{

CoUninitialize();

return;

}

hr = pConn->Open(strCon, "", "", 0);

if (FAILED(hr))

{

CoUninitialize();

return;

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

strCodigo = result;

char result2[100] = "";

strcpy(result2, ",\'");

strcat(result2, apodo);

strcat(result2, "\',");

strAge = result2;

char\* result3 = "";

result3 = appendCharToCharArray(result3, turno);

strcat(result3, ",");

strDOB = result3;

char\* result4 = "";

result4 = appendCharToCharArray(result4, disponible);

strcat(result4, ",");

strSalary = result4;

char\* result5 = "";

result5 = appendCharToCharArray(result5, buscando);

strcat(result5, ",");

strBuscando = result5;

char result6[100] = "";

strcpy(result6, "\'");

strcat(result6, casillas);

strcat(result6, "\')");

strCasillas = result6;

\_bstr\_t strSQL("INSERT INTO JUEGOS(CODIGO,APODO,TURNO,DISPONIBLE,BUSCANDO,CASILLAS) Values(");

strSQL += strCodigo + strAge + strDOB + strSalary + strBuscando + strCasillas;

pConn->Execute(strSQL, NULL, adExecuteNoRecords);

pConn->Close();

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return;

}

CoUninitialize();

return;

}

bool ConectarAleatorio(char\* code)

{

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t contrincanteR;

\_bstr\_t strCnn(CadenaCon);

\_RecordsetPtr pRstAuthors = NULL;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr))

{

return false;

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

code = result;

\_bstr\_t strSQL("SELECT [APODO],[CODIGO] FROM [JUEGOS] WHERE [DISPONIBLE]=1 AND [BUSCANDO]=1 AND DATEDIFF(minute, [FECHA], SYSDATETIME())<=5 AND [CODIGO]!=");

strSQL += code;

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

\_bstr\_t valField1;

pRstAuthors->MoveFirst();

if (!pRstAuthors->EndOfFile)

{

while (!pRstAuthors->EndOfFile)

{

contrincanteR = pRstAuthors->Fields->GetItem("APODO")->Value;

code = pRstAuthors->Fields->GetItem("CODIGO")->Value;

static std::string str(contrincanteR);

static const char \*cstr = str.c\_str();

contrincante = (char\*)cstr;

static std::string str2(code);

static const char \*cstr2 = str2.c\_str();

char\* tempcode = (char\*)cstr2;

codigoConectar[0] = tempcode[0];

codigoConectar[1] = tempcode[1];

codigoConectar[2] = tempcode[2];

codigoConectar[3] = tempcode[3];

ErrorAlConectarResult = false;

return OcuparAleatorio(codigoConectar);

pRstAuthors->MoveNext();

}

}

else

{

pressKey();

return false;

}

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return false;

}

CoUninitialize();

return false;

}

bool OcuparAleatorio(char\* code)

{

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t codeConectar;

\_bstr\_t contrincanteR;

\_bstr\_t strCnn(CadenaCon);

\_RecordsetPtr pRstAuthors = NULL;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr))

{

return false;

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

code = result;

char result1[100] = "";

strcpy(result1, "\'");

strcat(result1, codigoConectar);

strcat(result1, "\'");

codeConectar = result1;

\_bstr\_t strSQL("UPDATE JUEGOS SET DISPONIBLE=0,BUSCANDO=0,FECHA=NULL WHERE [CODIGO]=");

strSQL += (codeConectar + ";DELETE FROM JUEGOS WHERE [CODIGO]=" + code + ";");

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

turno = true;

MITURNO = false;

return true;

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

return false;

}

CoUninitialize();

return false;

}

bool AleatorioOcupado(char\* code)

{

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t disponible;

\_bstr\_t strCnn(CadenaCon);

\_RecordsetPtr pRstAuthors = NULL;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr))

{

pressKey();

return false;

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigo);

strcat(result, "\'");

code = result;

\_bstr\_t strSQL("SELECT [DISPONIBLE] FROM [JUEGOS] WHERE [BUSCANDO]=0 AND [DISPONIBLE]=0 AND [FECHA] IS NULL AND [CODIGO]=");

strSQL += code;

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

\_bstr\_t valField1;

pRstAuthors->MoveFirst();

if (!pRstAuthors->EndOfFile)

{

while (!pRstAuthors->EndOfFile)

{

turno = true;

MITURNO = true;

return true;

pRstAuthors->MoveNext();

}

}

else

{

pressKey();

return false;

}

}

catch (\_com\_error & ce)

{

ErrorAlConectar();

pressKey();

return false;

}

CoUninitialize();

pressKey();

return false;

}

DWORD WINAPI AsyncAleatorio(void\* data) {

while (0 == 0)

{

if (AsyncAleatorioActive)

{

HRESULT hr = S\_OK;

try

{

CoInitialize(NULL);

\_bstr\_t code;

\_bstr\_t strCnn(CadenaCon);

\_bstr\_t valField1;

\_RecordsetPtr pRstAuthors = NULL;

hr = pRstAuthors.CreateInstance(\_\_uuidof(Recordset));

std::string str(codigoConectar);

if (str.compare("XXXX") == 0)

{

codigoConectar[0] = codigo[0];

codigoConectar[1] = codigo[1];

codigoConectar[2] = codigo[2];

codigoConectar[3] = codigo[3];

codigoConectar[4] = codigo[4];

}

char result[100] = "";

strcpy(result, "\'");

strcat(result, codigoConectar);

strcat(result, "\'");

code = result;

\_bstr\_t strSQL("SELECT [TURNO] FROM [JUEGOS] WHERE [CODIGO]=");

strSQL += code;

pRstAuthors->Open(strSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText);

pRstAuthors->MoveFirst();

if (!pRstAuthors->EndOfFile)

{

while (!pRstAuthors->EndOfFile)

{

valField1 = pRstAuthors->Fields->GetItem("TURNO")->Value;

std::string disponible(valField1);

std::string::size\_type sz;

bool newturno = (bool)std::stoi(disponible, &sz);

if (turno != newturno)

{

if (!ActualizarTirosVzPlayer(newturno, true))

{

AsyncAleatorioActive = false;

}

}

pRstAuthors->MoveNext();

}

}

}

catch (\_com\_error & ce)

{

ErrorAlConectarResult = true;

ErrorAlConectar();

}

CoUninitialize();

}

else { Sleep(2000); }

}

return 0;

}

void CargarCasillasMultiplayerAleatorio(char\* apodo, char\* contrincante)

{

int tecla = 0;

int seleccion = 0;

int tempx = 0;

int temp = 0;

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0);

PrintColorXy("%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0);

PrintColorXy("%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0);

PrintColorXy("%s", " \n", 31, 7, 2, 15);

PrintColorXy("%s", " \n", 31, 8);

PrintColorXy("%s", " \n", 31, 9);

PrintColorXy("%s", " \n", 31, 10);

PrintColorXy("%s", " \n", 31, 11);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy("%s", "\xBA", 30, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy("TURNO DE:%s", (turno == MITURNO ? apodo : contrincante), 32, 31, 15, 0);

PrintColorXy("%s", "\xBA", 63, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0));

do

{

tecla = 0;

ErrorAlConectarResult = false;

AsyncAleatorioActive = true;

if (ErrorAlConectarResult)

{

return;

}

tecla = \_getch();

if (tecla == FLECHA) {

tecla = \_getch();

switch (tecla)

{

case ARRIBA:

if (seleccion == 0 || seleccion == 1 || seleccion == 2)

{

seleccion += 6;

}

else

{

seleccion -= 3;

}

break;

case DERECHA:

if (seleccion == 5 || seleccion == 2 || seleccion == 8)

{

seleccion -= 2;

}

else

{

seleccion++;

}

break;

case ABAJO:

if (seleccion == 6 || seleccion == 7 || seleccion == 8)

{

seleccion -= 6;

}

else

{

seleccion += 3;

}

break;

case IZQUIERDA:

if (seleccion == 6 || seleccion == 3 || seleccion == 0)

{

seleccion += 2;

}

else

{

seleccion--;

}

break;

}

}

else

{

if (tecla == ENTER)

{

if (turno != MITURNO || !puedeTirar)

{

MessageBoxA(NULL, "No es tu turno...", "Atención!", MB\_OK);

}

else

{

if (casillas[seleccion] == 0)

{

casillas[seleccion] = (MITURNO ? 1 : 2);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, 15, 0);

PrintColorXy("%s", "\xBA", 30, 31, 15, 0);

PrintColorXy("TURNO DE:%s", contrincante, 31, 31, 15, 0);

PrintColorXy("%s", "\xBA", 63, 31, 15, 0);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, 15, 0);

ErrorAlConectarResult = false;

EnviarTiro(true);

if (ErrorAlConectarResult) {

return;

}

puedeTirar = false;

}

else

{

MessageBoxA(NULL, "Casilla Ocupada", "Atención!", MB\_OK);

}

}

}

else

{

if (tecla == ESC)

{

return;

}

}

}

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

{

switch (i)

{

case 0:

tempx = 0;

break;

case 1:

tempx = 11;

break;

case 2:

tempx = 22;

break;

case 3:

tempx = 0;

break;

case 4:

tempx = 11;

break;

case 5:

tempx = 22;

break;

case 6:

tempx = 0;

break;

case 7:

tempx = 11;

break;

case 8:

tempx = 22;

break;

}

if (i >= 6)

{

temp = 13;

}

else

{

if (i <= 2)

{

temp = 1;

}

else

{

if (i >= 3 && i <= 5)

{

temp = 7;

}

}

}

switch (casillas[i])

{

case 0:

PrintColorXy("%s", " \n", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", " \n", (31 + tempx), 7 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 8 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \n", (31 + tempx), 10 + temp);

break;

case 1:

PrintColorXy("%s", " \\ \\ / /", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", " \\ V / ", (31 + tempx), 7 + temp);

PrintColorXy("%s", " > < ", (31 + tempx), 8 + temp);

PrintColorXy("%s", " / . \\ ", (31 + tempx), 9 + temp);

PrintColorXy("%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp);

break;

case 2:

PrintColorXy("%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0));

PrintColorXy("%s", "| | | |", (31 + tempx), 7 + temp);

PrintColorXy("%s", "| | | |", (31 + tempx), 8 + temp);

PrintColorXy("%s", "| `--' |", (31 + tempx), 9 + temp);

PrintColorXy("%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp);

break;

}

}

} while (Gano(1));

return;

}

void ErrorAlConectar()

{

AsyncAleatorioActive = false;

AsyncAnimation = false;

AsyncCodigo = false;

AsyncMarco = false;

AsyncTurno = false;

ErrorAlConectarResult = true;

RedibujarMarcos(4);

PrintColorXy("%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 15, 10, 15, 0);

PrintColorXy("%s", "\xBA", 15, 11, 15, 4);

PrintColorXy("%s", "Error al conectar con el servidor comprueba tu conexi\xA2n a internet... ", 16, 11, 15, 4);

PrintColorXy("%s", "\xBA", 86, 11, 15, 4);

PrintColorXy("%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 15, 12, 15, 0);

PrintColorXy("%s", " ¶¶", 24, 13, 4, 14);

PrintColorXy("%s", " ¶¶¶¶", 24, 14, 4, 14);

PrintColorXy("%s", " ¶¶¶¶¶¶", 24, 15, 4, 14);

PrintColorXy("%s", " ¶¶¶;;¶¶¶", 24, 16, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;¶¶¶", 24, 17, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;¶¶¶", 24, 18, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;¶¶;;;¶¶¶", 24, 19, 4, 14);

PrintColorXy("%s", " ¶¶¶;;¶¶¶¶¶¶;;¶¶¶", 24, 20, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;¶¶¶¶¶¶;;;¶¶¶", 24, 21, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;¶¶¶¶¶¶;;;;¶¶¶", 24, 22, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;¶¶¶¶¶¶;;;;;¶¶", 24, 23, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;¶¶¶¶¶;;;;;;¶¶", 24, 24, 4, 14);

PrintColorXy("%s", " ¶¶;;;;;;;;¶¶¶¶;;;;;;;;¶¶", 24, 25, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;;;¶¶¶¶;;;;;;;;;¶¶¶", 24, 26, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;;;;¶¶¶¶;;;;;;;;;;¶¶¶", 24, 27, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;;;;;;¶¶;;;;;;;;;;;;¶¶¶", 24, 28, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;;;;;;;;;;;;;;;;;;;;;;¶¶¶", 24, 29, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;;;;;;;¶¶¶¶¶;;;;;;;;;;;;¶¶¶", 24, 30, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;;;;;;;¶¶¶¶¶¶;;;;;;;;;;;;;¶¶¶", 24, 31, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;;;;;;;;;¶¶¶¶¶;;;;;;;;;;;;;;¶¶¶", 24, 32, 4, 14);

PrintColorXy("%s", " ¶¶¶;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;¶¶¶", 24, 33, 4, 14);

PrintColorXy("%s", "¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶", 24, 34, 4, 14);

\_getch();

RedibujarMarcos();

conectado = false;

puedeTirar = true;

MITURNO = false;

turno = false;

main(false);

}

PSeudocódigo

//Practica Final

//Alumno: García García Jonthan Eduardo

//Grupo: 1ICV45

//Profesor: Roberto Osornio Soto

//Materia: Fundamentos de Programación

Algoritmo PracticaFinal

Si (skip)

CrearVentana("Hola", WindowWidth, WindowHeight);

COORD variableReservadaze;

Si (GetFontvariableReservadaze(h &variableReservadaze))

Si (variableReservadaze.X != 10)

variableReservadaze.X = 10

FinSi

Si (variableReservadaze.Y != 18)

variableReservadaze.Y = 18

FinSi

SetFontvariableReservadazeh, variableReservadaze

FinSi

CONSOLE\_CURSOR\_INFO info

info.dwvariableReservadaze = 100

info.bVivariableReservadable = FALSE

SetConsoleCursorInfoh, &info

Introduccion

FinSi

RedibujarMarcos()

Segun MenuPrincipal Hacer

0:

RedibujarMarcos()

VzCPU()

RedibujarMarcos()

1:

RedibujarMarcos()

VzFriend()

2:

RedibujarMarcos()

VzAleatorio()

De Otro Modo:

RedibujarMarcos()

Esperar 0 segundos

exit

FinSegun

Esperar Tecla

FinAlgoritmo

SubProceso CREARVENTANA(\* TITLE, WIDTH, HEIGHT)

DWORD dwExStyle

DWORD dwStyle

wc.style = CS\_HREDRAW | CS\_VREDRAW | CS\_OWNDC

wc.cbClsExtra = 0

wc.cbWndExtra = 0

wc.hIcon = LoadIconNULL, IDI\_WINLOGO

wc.hCursor = LoadCursorNULL, IDC\_ARROW

wc.hbrBackground = NULL

wc.lpszMenuName = NULL

dwExStyle = WS\_EX\_APPWINDOW | WS\_EX\_WINDOWEDGE

dwStyle = WS\_OVERLAPPED | WS\_CAPTION | WS\_SYSMENU | WS\_MINIMIZEBOX | WS\_MAXIMIZEBOX

AdjustWindowRectEx&WindowRect, dwStyle, FALSE, dwExStyle

static PIXELFORMATDESCRIPTOR pfd =

variableReservadazeofPIXELFORMATDESCRIPTOR,

1,

PFD\_DRAW\_TO\_WINDOW |

PFD\_SUPPORT\_OPENGL |

PFD\_DOUBLEBUFFER,

PFD\_TYPE\_RGBA,

24,

0, 0, 0, 0, 0, 0,

0,

0,

0,

0, 0, 0, 0,

24,

0,

0,

PFD\_MAIN\_PLANE,

0,

0, 0, 0

FinSubProceso

0,0

Si (x < 0)

SetConsoleCursorPovariableReservadationGetStdHandle(STD\_OUTPUT\_HANDLE), coordenadas

FinSi

SetConsoleCursorPovariableReservadationGetStdHandle(STD\_OUTPUT\_HANDLE), cord

FinSubproceso

SubProceso GETFONTVARIABLERESERVADAZE( WINDOW, COORD \*VARIABLERESERVADAZE)

CONSOLE\_FONT\_INFOEX font =

variableReservadazeofCONSOLE\_FONT\_INFOEX

FinSubProceso

Si (!GetCurrentConsoleFontEx(windowHandle 0 &font))

FinSi

\*variableReservadaze = font.dwFontvariableReservadaze

SubProceso SETFONTVARIABLERESERVADAZE( WINDOW, COORD VARIABLERESERVADAZE)

CONSOLE\_FONT\_INFOEX font =

variableReservadazeofCONSOLE\_FONT\_INFOEX

FinSubProceso

Si (!GetCurrentConsoleFontEx(windowHandle 0 &font))

FinSi

font.dwFontvariableReservadaze = variableReservadaze

Si (!SetCurrentConsoleFontEx(windowHandle 0 &font))

FinSi

Si (b <= -1)

b = 7

FinSi

SetConsoleTextAttributeGetStdHandle(STD\_OUTPUT\_HANDLE), a << 4 | b

SubProceso RANDRANGE( MIN, MAX)

FinSubProceso

SubProceso RODUCCION()

Esperar 0 segundos

randcolor = RandRange1, 15

Escribir '\xC9'

Para int i = 2 Hasta 98 Con Paso 1

1Esperar 0 segundos

randcolor = RandRange1, 15

Escribir '\xCD'

Si (i > 37)

FinSi

Esperar 0 segundos

Escribir '\xBA'

FinPara

sleep = 2

randcolor = RandRange1, 15

Escribir '\xBB'

Escribir '\xC8'

Escribir '\xBC'

Para int i = 2 Hasta 98 Con Paso 1

1Esperar 0 segundos

randcolor = RandRange1, 15

Escribir '\xCD'

Si (i > 37)

FinSi

Esperar 0 segundos

randcolor = RandRange1, 15

Escribir '\xBA'

FinPara

Para int i = 2 Hasta 98 Con Paso 1

Para int y = 2 Hasta 38 Con Paso 1

Escribir ' '

FinPara

FinPara

Escribir ' \_\_\_\_\_\_ \_ \_\_\_\_\_ \_ \_ '

Escribir '| \_\_\_\_| \_ / \_\_\_\_| | || | '

Escribir '| |\_\_ \_\_\_ \_ \_ \_\_ \_\_\_ \_\_\_ | | \_ \_ | || |\_\_ \_ \_ \_\_ \_ \_\_\_ \_\_ \_ \_ \_\_ '

Escribir '| \_\_| / \_\_|| || \_ ` \_ \\ / \_ \\ | | | | | || || \_ \\ | | | | / \_` | / \_\_|/ \_` || \_ \\ '

Escribir '| |\_\_\_\_ \\\_\_ \\| || | | | | || \_\_/ | |\_\_\_\_| |\_| || || | | || |\_| || \_| || \_\_| \_| || | | |'

Escribir '|\_\_\_\_\_\_||\_\_\_/|\_||\_| |\_| |\_| \\\_\_\_| \\\_\_\_\_\_|\\\_\_,\_||\_||\_| |\_| \\\_\_,\_| \\\_\_,\_| \\\_\_\_|\\\_\_,\_||\_| |\_|'

Escribir '\*Use las flechas y teclas del teclado para navegar presione ESC para salir'

Para int x = 4 Hasta 9 Con Paso 1

Para int y = 12 Hasta 3 Con Paso 1

Si (y == 13)

Escribir ' '

FinSi

Escribir ' '

FinPara

FinPara

Escribir 'JUGAR GATO SOLITARIO Y VZ Jugador 2'

Escribir 'Alumno: Garca Garca Jonathan Eduardo', 161, 161

Escribir 'Grupo: 1ICV45'

Escribir 'Profesor: Roberto Osornio Soto'

Escribir 'Materia: Fundamentos de Programacin', 162

Escribir 'En esta versin de el juego de gato clasico podras jugar vz CPU o contra algun amigo...', 162

Escribir '>>>Pasos para jugar con amigos:'

Escribir '1 Debe seleccionar la opcion para jugar vz amigos y compartir sus codigos de acceso para'

Escribir 'que pueda establecerse una conexin.', 162

Escribir '2 Ambos deben seleccionar la opcion para jugar con amigos...'

Escribir '>>>Para escoger un oponente al azar simplemte seleccione la opcion marcada con este'

Escribir 'nombre y espere su turno... '

Escribir '>>>Para jugar contra el Cpu seleccione la opcion marcada con este nombre y comience. ' Hasta PrintColorXy"%s", " ", 30, 34, 0, 15 Con Paso 1

PrintColorXy"%s", " ", 30, 36, 0, 15

AsyncMarco = true

Esperar Tecla

AsyncMarco = false

FinSubProceso

SubProceso REDIBUJARMARCOS( BACKGROUND)

Esperar 0 segundos

randcolor = RandRange1, 15

Escribir '\xC9'

Para int i = 2 Hasta 98 Con Paso 1

1Esperar 0 segundos

randcolor = RandRange1, 15

Escribir '\xCD'

Si (i > 37)

FinSi

Esperar 0 segundos

Escribir '\xBA'

FinPara

sleep = 2

randcolor = RandRange1, 15

Escribir '\xBB'

Escribir '\xC8'

Escribir '\xBC'

Para int i = 2 Hasta 98 Con Paso 1

1Esperar 0 segundos

randcolor = RandRange1, 15

Escribir '\xCD'

Si (i > 37)

FinSi

Esperar 0 segundos

randcolor = RandRange1, 15

Escribir '\xBA'

FinPara

Para int i = 2 Hasta 98 Con Paso 1

Para int y = 2 Hasta 38 Con Paso 1

Escribir ' '

FinPara

FinPara

FinSubProceso

SubProceso MENUPRINCIPAL()

PrintColorXy"%s", "\*Use las flechas y teclas del teclado para navegar prevariableReservadaone ESC para salir", 2, 37

AsyncAnimation = true

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0

Hacer

AsyncAnimation = false

10Esperar 0 segundos

j = 0

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 3, 30, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

PrintColorXy"%s", (seleccion == j ? "\xBA>>>>>>>>JUGADOR VZ CPU<<<<<<<<\xBA" : "\xBA JUGADOR VZ CPU \xBA"), 3, 31, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 3, 32, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

j=j+1

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 36, 30, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

PrintColorXy"%s", (seleccion == j ? "\xBA>>>>JUGADOR 1 VZ JUGADOR 2<<<<\xBA" : "\xBA JUGADOR 1 VZ JUGADOR 2 \xBA"), 36, 31, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 36, 32, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

j=j+1

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 69, 30, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

PrintColorXy"%s", (seleccion == j ? "\xBA>>>>OPONENTE ALEATORIO<<<<\xBA" : "\xBA OPONENTE ALEATORIO \xBA"), 69, 31, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 69, 32, (seleccion == j ? 9 : 15), (seleccion == j ? 15 : 0)

AsyncAnimation = true

Esperar Tecla

AsyncAnimation = false

Si (tecla == FLECHA)

Esperar Tecla

Si (tecla == ARRIBA || tecla == DERECHA)

seleccion=seleccion+1

SiNo

Si (tecla == ABAJO || tecla == IZQUIERDA)

seleccion=seleccion-1

FinSi

FinSi

SiNo

Si (tecla == ESC)

FinSi

FinSi

Si (seleccion < 0)

seleccion = 2

FinSi

Si (seleccion > 2)

seleccion = 0

FinSi

Hasta Que (tecla != ENTER)

FinSubProceso

SubProceso PRCOLORXY(\* FORMAT, \* TEXTO, X, Y, BACKGROUND, TEXTCOLOR)

Si (background > -1)

FinSi

Si (x > -1 y y > -1)

COORD c =

FinSi

FinSubProceso

Si (wherex() == x y wherey() == y)

Escribir format, texto

FinSi

}

SubProceso ASYNCCONECTAR(\* DATA)

Mientras 0 == 0

Si (AsyncCodigo)

string codecodigo

ErrorAlConectarResult = false

Si (code.compare(XXXX) != 0 y Ocupado(codigo) y !ErrorAlConectarResult)

turno = true

conectado = true

pressKey

FinSi

Si (ErrorAlConectarResult)

FinSi

SiNo

2000Esperar 2 segundos

FinSi

FinMientras

FinSubProceso

SubProceso ASYNCTABLEANIMATION(\* DATA)

Mientras 0 == 0

Si (AsyncAnimation)

0,0,0,0,0,0,0,0,0

FinSi

results[RandRange0, 8] = 1

results[RandRange0, 8] = 2

results[RandRange0, 8] = 1

results[RandRange0, 8] = 2

results[RandRange0, 8] = 1

results[RandRange0, 8] = 2

results[RandRange0, 8] = 1

results[RandRange0, 8] = 2

results[RandRange0, 8] = 1

Para int i = 0 Hasta Con Paso 1

Si (results[i] == 0)

Si (i > 0 y results[i - 1] == 1)

results[i] = 2

SiNo

Si (i > 0 y results[i - 1] == 2)

results[i] = 1

SiNo

results[i] = RandRange1, 2

FinSi

FinSi

FinSi

FinPara

Para int i = 0 Hasta Con Paso 1

Si (!AsyncAnimation)

FinSi

Si (i >= 6)

temp = 13

SiNo

Si (i <= 2)

temp = 1

SiNo

Si (i >= 3 y i <= 5)

temp = 7

tempx = 1

FinSi

FinSi

FinSi

Si (!AsyncAnimation)

FinSi

Segun i Hacer

0:

tempx = 0

1:

tempx = 11

2:

tempx = 22

3:

tempx = 0

4:

tempx = 11

5:

tempx = 22

6:

tempx = 0

7:

tempx = 11

8:

tempx = 22

FinSegun

Si (!AsyncAnimation)

FinSi

Segun results[i] Hacer

0:

Si (!AsyncAnimation)

FinSi

PrintColorXy"%s", " \n", (31 + tempx), 6 + temp, 15, 0

PrintColorXy"%s", " \n", (31 + tempx), 7 + temp

PrintColorXy"%s", " \n", (31 + tempx), 8 + temp

PrintColorXy"%s", " \n", (31 + tempx), 9 + temp

PrintColorXy"%s", " \n", (31 + tempx), 10 + temp

Esperar 0 segundos

1:

Si (!AsyncAnimation)

FinSi

PrintColorXy"%s", " \\ \\ / /", (31 + tempx), 6 + temp, 9, 15

PrintColorXy"%s", " \\ V / ", (31 + tempx), 7 + temp

PrintColorXy"%s", " > < ", (31 + tempx), 8 + temp

PrintColorXy"%s", " / . \\ ", (31 + tempx), 9 + temp

PrintColorXy"%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp

Esperar 0 segundos

2:

Si (!AsyncAnimation)

FinSi

PrintColorXy"%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, 4, 15

PrintColorXy"%s", "| | | |", (31 + tempx), 7 + temp

PrintColorXy"%s", "| | | |", (31 + tempx), 8 + temp

PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp=PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp-1

PrintColorXy"%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp

Esperar 0 segundos

FinSegun

Si (!AsyncAnimation)

FinSi

0,0,0,0,0,0,0,0,0

FinPara

Para int j = 0 Hasta Con Paso 1

ar[j] = results[j]

FinPara

Si (SeAutoGano(ar))

2000Esperar 2 segundos

Si (!AsyncAnimation)

FinSi

Para int i = 0 Hasta Con Paso 1

Si (!AsyncAnimation)

FinSi

Si (i >= 6)

temp = 13

SiNo

Si (i <= 2)

temp = 1

SiNo

Si (i >= 3 y i <= 5)

temp = 7

tempx = 1

FinSi

FinSi

FinSi

Si (!AsyncAnimation)

FinSi

Segun i Hacer

0:

tempx = 0

1:

tempx = 11

2:

tempx = 22

3:

tempx = 0

4:

tempx = 11

5:

tempx = 22

6:

tempx = 0

7:

tempx = 11

8:

tempx = 22

FinSegun

Si (!AsyncAnimation)

FinSi

PrintColorXy"%s", " \n", (31 + tempx), 6 + temp, 15, 0

PrintColorXy"%s", " \n", (31 + tempx), 7 + temp

PrintColorXy"%s", " \n", (31 + tempx), 8 + temp

PrintColorXy"%s", " \n", (31 + tempx), 9 + temp

PrintColorXy"%s", " \n", (31 + tempx), 10 + temp

FinPara

i = 9

FinSi

FinMientras

SiNo

Esperar 2 segundos

FinSi

FinSubProceso

SubProceso ASYNCMARCOANIMATION(\* DATA)

Mientras AsyncMarco

randcolor = RandRange1, 15

Escribir '\xC9'

Para int i = 2 Hasta 98 Con Paso 1

randcolor = RandRange1, 15

Escribir '\xCD'

Si (i > 37)

FinSi

Escribir '\xBA'

FinPara

randcolor = RandRange1, 15

Escribir '\xBB'

Escribir '\xC8'

Escribir '\xBC'

Para int i = 2 Hasta 98 Con Paso 1

randcolor = RandRange1, 15

Escribir '\xCD'

Si (i > 37)

FinSi

randcolor = RandRange1, 15

Escribir '\xBA'

FinPara

500Esperar 0 segundos

FinMientras

FinSubProceso

SubProceso ASYNCGETTURNO(\* DATA)

Mientras 0 == 0

Si (AsyncTurno)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t strCnnCadenaCon

\_bstr\_t valField1

\_RecordsetPtr pRstAuthors = NULL

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

code = result

\_bstr\_t strSQL"SELECT [TURNO] FROM [JUEGOS] WHERE [CODIGO]="

strSQL += code

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

pRstAuthors->MoveFirst

Si (!pRstAuthors->EndOfFile)

Mientras !pRstAuthors->EndOfFile

valField1 = pRstAuthors->Fields->GetItem"TURNO"->Value

std::string disponiblevalField1

std::string::variableReservadaze\_type sz

Si (!puedeTirar || turno != newturno)

Si (!ActualizarTirosVzPlayer(newturno))

pressKey

AsyncTurno = false

FinSi

FinSi

pRstAuthors->MoveNext

FinMientras

FinSi

FinSi

catch \_com\_error & ce

FinMientras

CoUninitialize

SiNo

Esperar 2 segundos

FinSi

FinSubProceso

SubProceso WHEREX()

CONSOLE\_SCREEN\_BUFFER\_INFO csbi

Si (!GetConsoleScreenBufferInfo(

GetStdHandleSTD\_OUTPUT\_HANDLE,

&csbi

)

FinSi

SubProceso WHEREY()

CONSOLE\_SCREEN\_BUFFER\_INFO csbi

Si (!GetConsoleScreenBufferInfo(

GetStdHandleSTD\_OUTPUT\_HANDLE,

&csbi

)

FinSi

SubProceso SEAUTOGANO( RESULTS[8])

Para int i = 0 Hasta Con Paso 1

Si (results[i] != 0 y results[i + 1] != 0 y results[i + 2] != 0 y (i == 0 || i == 3 || i == 6))

Si (1 == results[i] y 1 == results[i + 1] y 1 == results[i + 2])

FinSi

Si (2 == results[i] y 2 == results[i + 1] y 2 == results[i + 2])

FinSi

FinSi

Si (i == 0)

Si (results[i] != 0 y results[i + 4] != 0 y results[i + 8] != 0)

Si (1 == results[i] y 1 == results[i + 4] y 1 == results[i + 8])

FinSi

Si (2 == results[i] y 2 == results[i + 4] y 2 == results[i + 8])

FinSi

FinSi

Si (results[i] != 0 y results[i + 3] != 0 y results[i + 6] != 0)

Si (1 == results[i] y 1 == results[i + 3] y 1 == results[i + 6])

FinSi

Si (2 == results[i] y 2 == results[i + 3] y 2 == results[i + 6])

FinSi

FinSi

FinSi

Si (i == 1)

Si (results[i] != 0 y results[i + 3] != 0 y results[i + 6] != 0)

Si (1 == results[i] y 1 == results[i + 3] y 1 == results[i + 6])

FinSi

Si (2 == results[i] y 2 == results[i + 3] y 2 == results[i + 6])

FinSi

FinSi

FinSi

Si (i == 2)

Si (results[i] != 0 y results[i + 2] != 0 y results[i + 4] != 0)

Si (1 == results[i] y 1 == results[i + 2] y 1 == results[i + 4])

FinSi

Si (2 == results[i] y 2 == results[i + 2] y 2 == results[i + 4])

FinSi

FinSi

Si (results[i] != 0 y results[i + 3] != 0 y results[i + 6] != 0)

Si (1 == results[i] y 1 == results[i + 3] y 1 == results[i + 6])

FinSi

Si (2 == results[i] y 2 == results[i + 3] y 2 == results[i + 6])

FinSi

FinSi

FinSi

FinPara

FinSubProceso

SubProceso VZCPU()

Para int i = 0 Hasta Con Paso 1

cavariableReservadallas[i] = 0

FinPara

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0

PrintColorXy"%s", " \n", 31, 7, 2, 15

PrintColorXy"%s", " \n", 31, 8

PrintColorXy"%s", " \n", 31, 9

PrintColorXy"%s", " \n", 31, 10

PrintColorXy"%s", " \n", 31, 11

Hacer

tecla = 0

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno ? 9 : 15), (turno ? 15 : 0)

PrintColorXy"%s", (turno ? "\xBA>>>>>>>>>>>>TU TURNO<<<<<<<<<<<<\xBA" : "\xBA>>>>>>>>>>>>MI TURNO<<<<<<<<<<<<\xBA"), 30, 31, (turno ? 9 : 15), (turno ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno ? 9 : 15), (turno ? 15 : 0)

Esperar Tecla

Si (tecla == FLECHA)

Esperar Tecla

Segun tecla Hacer

ARRIBA:

Si (seleccion == 0 || seleccion == 1 || seleccion == 2)

seleccion += 6

SiNo

seleccion -= 3

FinSi

DERECHA:

Si (seleccion == 5 || seleccion == 2 || seleccion == 8)

seleccion -= 2

SiNo

seleccion=seleccion+1

FinSi

ABAJO:

Si (seleccion == 6 || seleccion == 7 || seleccion == 8)

seleccion -= 6

SiNo

seleccion += 3

FinSi

IZQUIERDA:

Si (seleccion == 6 || seleccion == 3 || seleccion == 0)

seleccion += 2

SiNo

seleccion=seleccion-1

FinSi

FinSegun

SiNo

Si (tecla == ENTER)

Si (cavariableReservadallas[seleccion] == 0 y turno)

cavariableReservadallas[seleccion] = 1

turno = false

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno ? 9 : 15), (turno ? 15 : 0)

PrintColorXy"%s", (turno ? "\xBA>>>>>>>>>>>>TU TURNO<<<<<<<<<<<<\xBA" : "\xBA>>>>>>>>>>>>MI TURNO<<<<<<<<<<<<\xBA"), 30, 31, (turno ? 9 : 15), (turno ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno ? 9 : 15), (turno ? 15 : 0)

2000Esperar 2 segundos

Si (!TiraCpu())

MessageBoxANULL, "El tablero esta lleno", "Atención!", MB\_OK

SiNo

turno = true

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno ? 9 : 15), (turno ? 15 : 0)

PrintColorXy"%s", (turno ? "\xBA>>>>>>>>>>>>TU TURNO<<<<<<<<<<<<\xBA" : "\xBA>>>>>>>>>>>>MI TURNO<<<<<<<<<<<<\xBA"), 30, 31, (turno ? 9 : 15), (turno ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno ? 9 : 15), (turno ? 15 : 0)

FinSi

SiNo

MessageBoxANULL, "CavariableReservadalla Ocupada", "Atención!", MB\_OK

FinSi

SiNo

Si (tecla == ESC)

FinSi

FinSi

FinSi

Para int i = 0 Hasta Con Paso 1

Segun i Hacer

0:

tempx = 0

1:

tempx = 11

2:

tempx = 22

3:

tempx = 0

4:

tempx = 11

5:

tempx = 22

6:

tempx = 0

7:

tempx = 11

8:

tempx = 22

FinSegun

Si (i >= 6)

temp = 13

SiNo

Si (i <= 2)

temp = 1

SiNo

Si (i >= 3 y i <= 5)

temp = 7

FinSi

FinSi

FinSi

Segun cavariableReservadallas[i] Hacer

0:

PrintColorXy"%s", " \n", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", " \n", (31 + tempx), 7 + temp

PrintColorXy"%s", " \n", (31 + tempx), 8 + temp

PrintColorXy"%s", " \n", (31 + tempx), 9 + temp

PrintColorXy"%s", " \n", (31 + tempx), 10 + temp

1:

PrintColorXy"%s", " \\ \\ / /", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", " \\ V / ", (31 + tempx), 7 + temp

PrintColorXy"%s", " > < ", (31 + tempx), 8 + temp

PrintColorXy"%s", " / . \\ ", (31 + tempx), 9 + temp

PrintColorXy"%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp

2:

PrintColorXy"%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", "| | | |", (31 + tempx), 7 + temp

PrintColorXy"%s", "| | | |", (31 + tempx), 8 + temp

PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp=PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp-1

PrintColorXy"%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp

FinSegun

FinPara

Hasta Que (Gano(0))

FinSubProceso

SubProceso TIRACPU()

Si (cavariableReservadallas[7] == 0)

cavariableReservadallas[7] = 2

FinSi

Si (Bloquear())

FinSi

Si (TiroParaGanar())

FinSi

FinSubProceso

SubProceso BLOQUEAR()

Para int i = 0 Hasta Con Paso 1

Si (i == 0 || i == 3 || i == 6)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 1] != 0 y cavariableReservadallas[i + 2] == 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 1] y cavariableReservadallas[i + 2] == 0)

cavariableReservadallas[i + 2] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 1] == 0 y cavariableReservadallas[i + 2] != 0)

Si (1 == cavariableReservadallas[i] y cavariableReservadallas[i + 1] == 0 y 1 == cavariableReservadallas[i + 2])

cavariableReservadallas[i + 1] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] == 0 y cavariableReservadallas[i + 1] != 0 y cavariableReservadallas[i + 2] != 0)

Si (cavariableReservadallas[i] == 0 y 1 == cavariableReservadallas[i + 1] y 1 == cavariableReservadallas[i + 2])

cavariableReservadallas[i] = 2

FinSi

FinSi

FinSi

Si (i == 0)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 4] != 0 y cavariableReservadallas[i + 8] == 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 4] y cavariableReservadallas[i + 8] == 0)

cavariableReservadallas[i + 8] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 4] == 0 y cavariableReservadallas[i + 8] != 0)

Si (1 == cavariableReservadallas[i] y cavariableReservadallas[i + 4] == 0 y 1 == cavariableReservadallas[i + 8])

cavariableReservadallas[i + 4] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] == 0 y cavariableReservadallas[i + 4] != 0 y cavariableReservadallas[i + 8] != 0)

Si (cavariableReservadallas[i] == 0 y 1 == cavariableReservadallas[i + 4] y 1 == cavariableReservadallas[i + 8])

cavariableReservadallas[i] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 6] == 0 y cavariableReservadallas[i + 7] != 0 y cavariableReservadallas[i + 8] != 0)

Si (cavariableReservadallas[i + 6] == 0 y 1 == cavariableReservadallas[i + 7] y 1 == cavariableReservadallas[i + 8])

cavariableReservadallas[i + 6] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 6] != 0 y cavariableReservadallas[i + 7] == 0 y cavariableReservadallas[i + 8] != 0)

Si (1 == cavariableReservadallas[i + 6] y cavariableReservadallas[i + 7] == 0 y 1 == cavariableReservadallas[i + 8])

cavariableReservadallas[i + 7] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 6] != 0 y cavariableReservadallas[i + 7] != 0 y cavariableReservadallas[i + 8] == 0)

Si (1 == cavariableReservadallas[i + 6] y 1 == cavariableReservadallas[i + 4] y cavariableReservadallas[i + 8] == 0)

cavariableReservadallas[i + 8] = 2

FinSi

FinSi

FinSi

Si (i == 1)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] == 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 3] y cavariableReservadallas[i + 6] == 0)

cavariableReservadallas[i + 6] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] == 0 y cavariableReservadallas[i + 6] != 0)

cavariableReservadallas[i + 3] = 2

FinSi

Si (cavariableReservadallas[i] == 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (cavariableReservadallas[i] == 0 y 1 == cavariableReservadallas[i + 3] y 1 == cavariableReservadallas[i + 6])

cavariableReservadallas[i] = 2

FinSi

FinSi

FinSi

Si (i == 2)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 2] != 0 y cavariableReservadallas[i + 4] == 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 2] y cavariableReservadallas[i + 4] == 0)

cavariableReservadallas[i + 4] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 2] == 0 y cavariableReservadallas[i + 4] != 0)

Si (1 == cavariableReservadallas[i] y cavariableReservadallas[i + 2] == 0 y 1 == cavariableReservadallas[i + 4])

cavariableReservadallas[i + 2] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] == 0 y cavariableReservadallas[i + 2] != 0 y cavariableReservadallas[i + 4] != 0)

Si (cavariableReservadallas[i] == 0 y 1 == cavariableReservadallas[i + 2] y 1 == cavariableReservadallas[i + 4])

cavariableReservadallas[i] = 2

FinSi

FinSi

FinSi

Si (i == 0 || i == 1 || i == 2)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] == 0)

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 3] y cavariableReservadallas[i + 6] == 0)

cavariableReservadallas[i + 6] = 2

FinSi

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 3] y cavariableReservadallas[i + 6] == 0)

cavariableReservadallas[i + 6] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] == 0 y cavariableReservadallas[i + 6] != 0)

Si (1 == cavariableReservadallas[i] y cavariableReservadallas[i + 3] == 0 y 1 == cavariableReservadallas[i + 6])

cavariableReservadallas[i + 3] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] == 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (cavariableReservadallas[i] == 0 y 1 == cavariableReservadallas[i + 3] y 1 == cavariableReservadallas[i + 6])

cavariableReservadallas[i] = 2

FinSi

FinSi

FinSi

Si (i == 5 || i == 2)

Si (cavariableReservadallas[i + 1] != 0 y cavariableReservadallas[i + 2] != 0 y cavariableReservadallas[i + 3] == 0)

Si (1 == cavariableReservadallas[i + 1] y 1 == cavariableReservadallas[i + 2])

cavariableReservadallas[i + 3] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 1] != 0 y cavariableReservadallas[i + 2] == 0 y cavariableReservadallas[i + 3] != 0)

Si (1 == cavariableReservadallas[i + 1] y 1 == cavariableReservadallas[i + 3])

cavariableReservadallas[i + 2] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 1] == 0 y cavariableReservadallas[i + 2] != 0 y cavariableReservadallas[i + 3] != 0)

Si (1 == cavariableReservadallas[i + 2] y 1 == cavariableReservadallas[i + 3])

cavariableReservadallas[i + 1] = 2

FinSi

FinSi

FinSi

FinPara

Si (cavariableReservadallas[0] == 0)

cavariableReservadallas[0] = 2

FinSi

Si (cavariableReservadallas[2] == 0)

cavariableReservadallas[2] = 2

FinSi

Si (cavariableReservadallas[6] == 0)

cavariableReservadallas[6] = 2

FinSi

Si (cavariableReservadallas[8] == 0)

cavariableReservadallas[8] = 2

FinSi

FinSubProceso

SubProceso TIROPARAGANAR()

Para int i = 0 Hasta Con Paso 1

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 1] != 0)

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 1] y cavariableReservadallas[i + 2] == 0)

cavariableReservadallas[i + 2] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 2] != 0)

Si (2 == cavariableReservadallas[i] y cavariableReservadallas[i + 1] == 0 y 2 == cavariableReservadallas[i + 2])

cavariableReservadallas[i + 1] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 1] != 0 y cavariableReservadallas[i + 2] != 0)

Si (cavariableReservadallas[i] == 0 y 2 == cavariableReservadallas[i + 1] y 2 == cavariableReservadallas[i + 2])

cavariableReservadallas[i] = 2

FinSi

FinSi

Si (i == 0)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 4] != 0)

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 4] y cavariableReservadallas[i + 8] == 0)

cavariableReservadallas[i + 8] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 8] != 0)

Si (2 == cavariableReservadallas[i] y cavariableReservadallas[i + 4] == 0 y 2 == cavariableReservadallas[i + 8])

cavariableReservadallas[i + 4] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 4] != 0 y cavariableReservadallas[i + 8] != 0)

Si (cavariableReservadallas[i] == 0 y 2 == cavariableReservadallas[i + 4] y 2 == cavariableReservadallas[i + 8])

cavariableReservadallas[i] = 2

FinSi

FinSi

FinSi

Si (i == 1)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0)

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 3] y cavariableReservadallas[i + 6] == 0)

cavariableReservadallas[i + 6] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 6] != 0)

Si (2 == cavariableReservadallas[i] y cavariableReservadallas[i + 3] == 0 y 2 == cavariableReservadallas[i + 6])

cavariableReservadallas[i + 3] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (cavariableReservadallas[i] == 0 y 2 == cavariableReservadallas[i + 3] y 2 == cavariableReservadallas[i + 6])

cavariableReservadallas[i] = 2

FinSi

FinSi

FinSi

Si (i == 2)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 2] != 0)

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 2] y cavariableReservadallas[i + 4] == 0)

cavariableReservadallas[i + 4] = 2

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 4] != 0)

Si (2 == cavariableReservadallas[i] y cavariableReservadallas[i + 2] == 0 y 2 == cavariableReservadallas[i + 4])

cavariableReservadallas[i + 2] = 2

FinSi

FinSi

Si (cavariableReservadallas[i + 2] != 0 y cavariableReservadallas[i + 4] != 0)

Si (cavariableReservadallas[i] == 0 y 2 == cavariableReservadallas[i + 2] y 2 == cavariableReservadallas[i + 4])

cavariableReservadallas[i] = 2

FinSi

FinSi

FinSi

FinSubProceso

SubProceso TIRODISPONIBLE()

Para int i = 0 Hasta Con Paso 1

Si (cavariableReservadallas[i] == 0)

cavariableReservadallas[i] = 2

FinSi

FinPara

FinSubProceso

SubProceso GANO( MODO)

Para int i = 0 Hasta Con Paso 1

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 1] != 0 y cavariableReservadallas[i + 2] != 0 y (i == 0 || i == 3 || i == 6))

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 1] y 1 == cavariableReservadallas[i + 2])

Si (modo == 0)

MessageBoxANULL, "Venciste al Cpu :)", "Felicidades", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (X)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 1] y 2 == cavariableReservadallas[i + 2])

Si (modo == 0)

MessageBoxANULL, "Gana Cpu :(", "Lastima...", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (O)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

FinSi

Si (i == 0)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 4] != 0 y cavariableReservadallas[i + 8] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 4] y 1 == cavariableReservadallas[i + 8])

Si (modo == 0)

MessageBoxANULL, "Venciste al Cpu :)", "Felicidades", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (X)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 4] y 2 == cavariableReservadallas[i + 8])

Si (modo == 0)

MessageBoxANULL, "Gana Cpu :(", "Lastima...", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (O)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 3] y 1 == cavariableReservadallas[i + 6])

Si (modo == 0)

MessageBoxANULL, "Venciste al Cpu :)", "Felicidades", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (X)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 3] y 2 == cavariableReservadallas[i + 6])

Si (modo == 0)

MessageBoxANULL, "Gana Cpu :(", "Lastima...", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (O)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

FinSi

FinSi

Si (i == 1)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 3] y 1 == cavariableReservadallas[i + 6])

Si (modo == 0)

MessageBoxANULL, "Venciste al Cpu :)", "Felicidades", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (X)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 3] y 2 == cavariableReservadallas[i + 6])

Si (modo == 0)

MessageBoxANULL, "Venciste al Cpu :)", "Felicidades", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (O)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

FinSi

FinSi

Si (i == 2)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 2] != 0 y cavariableReservadallas[i + 4] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 2] y 1 == cavariableReservadallas[i + 4])

Si (modo == 0)

MessageBoxANULL, "Venciste al Cpu :)", "Felicidades", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (X)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 2] y 2 == cavariableReservadallas[i + 4])

Si (modo == 0)

MessageBoxANULL, "Gana Cpu :(", "Lastima...", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (O)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 3] y 1 == cavariableReservadallas[i + 6])

Si (modo == 0)

MessageBoxANULL, "Venciste al Cpu :)", "Felicidades", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (X)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 3] y 2 == cavariableReservadallas[i + 6])

Si (modo == 0)

MessageBoxANULL, "Gana Cpu :(", "Lastima...", MB\_OK

conectado = true

FinSi

Si (modo == 1)

MessageBoxANULL, "Gana: (O)", "Felicidades", MB\_OK

conectado = true

FinSi

FinSi

FinSi

FinSi

FinPara

Para int i = 0 Hasta Con Paso 1

Si (cavariableReservadallas[i] == 0)

conectado = true

FinSi

FinPara

MessageBoxANULL, "El tablero esta lleno...", "Empate!", MB\_OK

conectado = true

FinSubProceso

SubProceso VARIABLERESERVADALENCEGANO( MODO)

Para int i = 0 Hasta Con Paso 1

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 1] != 0 y cavariableReservadallas[i + 2] != 0 y (i == 0 || i == 3 || i == 6))

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 1] y 1 == cavariableReservadallas[i + 2])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 1] y 2 == cavariableReservadallas[i + 2])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

FinSi

Si (i == 0)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 4] != 0 y cavariableReservadallas[i + 8] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 4] y 1 == cavariableReservadallas[i + 8])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 4] y 2 == cavariableReservadallas[i + 8])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 3] y 1 == cavariableReservadallas[i + 6])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 3] y 2 == cavariableReservadallas[i + 6])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

FinSi

FinSi

Si (i == 1)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 3] y 1 == cavariableReservadallas[i + 6])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 3] y 2 == cavariableReservadallas[i + 6])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

FinSi

FinSi

Si (i == 2)

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 2] != 0 y cavariableReservadallas[i + 4] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 2] y 1 == cavariableReservadallas[i + 4])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 2] y 2 == cavariableReservadallas[i + 4])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

FinSi

Si (cavariableReservadallas[i] != 0 y cavariableReservadallas[i + 3] != 0 y cavariableReservadallas[i + 6] != 0)

Si (1 == cavariableReservadallas[i] y 1 == cavariableReservadallas[i + 3] y 1 == cavariableReservadallas[i + 6])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

Si (2 == cavariableReservadallas[i] y 2 == cavariableReservadallas[i + 3] y 2 == cavariableReservadallas[i + 6])

Si (modo == 0)

FinSi

Si (modo == 1)

FinSi

FinSi

FinSi

FinSi

FinPara

Para int i = 0 Hasta Con Paso 1

Si (cavariableReservadallas[i] == 0)

FinSi

FinPara

MessageBoxANULL, "El tablero esta lleno...", "Empate!", MB\_OK

FinSubProceso

SubProceso VZFRIEND()

turno = true

Para int i = 0 Hasta 2 Con Paso 1

apodo[i] = ' '

FinPara

contrincante = "Jugador 2"

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 25, 15, 0

PrintColorXy"%s", "\xBA", 30, 26

PrintColorXy"%s", " ", 31, 26, 0, 15

PrintColorXy"%s", "\xBA", 64, 26, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 27, 15, 0

PrintColorXy"%s", "Escribe un apodo original

)", 30, 23, 15, 0

Mientras tecla != ENTER

Si (tecla == ESC)

FinSi

Si (tecla != ENTER y tecla != ESC y tecla != FLECHA)

i=i+1

PrintColorXy"%s", apodo, 31, 26, 0, 15

FinSi

FinMientras

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 25, 15, 0

PrintColorXy"%s", "\xBA", 30, 26

PrintColorXy"%s", " ", 31, 26, 0, 15

PrintColorXy"%s", "\xBA", 64, 26, 15, 0

PrintColorXy"%s", "\xBA", 30, 27

PrintColorXy"%s", ">>Conectando con el servidor...<<", 31, 27, 0, 15

PrintColorXy"%s", "\xBA", 64, 27, 15, 0

PrintColorXy"%s", "\xBA", 30, 28

PrintColorXy"%s", " ", 31, 28, 0, 15

PrintColorXy"%s", "\xBA", 64, 28, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 29, 15, 0

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0

AsyncAnimation = true

sprintfcode, "%d", RandRange(0, 9)

TempCode = appendCharToCharArrayTempCode, code[0]

sprintfcode, "%d", RandRange(0, 9)

TempCode = appendCharToCharArrayTempCode, code[0]

ErrorAlConectarResult = false

Mientras ExisteCodigoTempCode && !ErrorAlConectarResult

TempCode = ""

sprintfcode, "%d", RandRange(0, 9)

TempCode = appendCharToCharArrayTempCode, code[0]

sprintfcode, "%d", RandRange(0, 9)

TempCode = appendCharToCharArrayTempCode, code[0]

FinMientras

Si (ErrorAlConectarResult)

FinSi

PrintColorXy"%s", ">>Obteniendo Codigo de Jugador.<<", 31, 27, 0, 15

codigo[0] = TempCode[0]

codigo[1] = TempCode[1]

codigo[2] = TempCode[2]

codigo[3] = TempCode[3]

ErrorAlConectarResult = false

Escribircodigo, apodo

Si (ErrorAlConectarResult)

FinSi

AsyncAnimation = false

10Esperar 0 segundos

RedibujarMarcos

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 12, 15, 0

PrintColorXy"%s", "\xBA", 30, 13

PrintColorXy"%s", " ", 31, 13, 0, 15

PrintColorXy"%s", "\xBA", 64, 13, 15, 0

PrintColorXy"%s", "\xBA", 30, 14

PrintColorXy">>Tu codigo de jugador es:%s <<", codigo, 31, 14, 0, 15

PrintColorXy"%s", "\xBA", 64, 14, 15, 0

PrintColorXy"%s", "\xBA", 30, 15

PrintColorXy"%s", " ", 31, 15, 0, 15

PrintColorXy"%s", "\xBA", 64, 15, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 16, 15, 0

PrintColorXy"%s", "Comparte tu codigo con un amigo o pide que te lo compartan y escribelo aqui.", 11, 27, 15, 0

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 20, 15, 0

PrintColorXy"%s", "\xBA", 30, 21

PrintColorXy"%s", " ", 31, 21, 0, 15

PrintColorXy"%s", "\xBA", 64, 21, 15, 0

PrintColorXy"%s", "\xBA", 30, 22

PrintColorXy">%s<", codigoConectar, 31, 22, 0, 15=PrintColorXy">%s<", codigoConectar, 31, 22, 0, 15-1

PrintColorXy"%s", "\xBA", 64, 22, 15, 0

PrintColorXy"%s", "\xBA", 30, 23

PrintColorXy"%s", " ", 31, 23, 0, 15

PrintColorXy"%s", "\xBA", 64, 23, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0

AsyncCodigo = true

i = 0

Mientras !conectado

Esperar Tecla

Si (tecla != ESC)

PrintColorXy">%s<-", codigoConectar, 31, 22, 0, 15=PrintColorXy">%s<-", codigoConectar, 31, 22, 0, 15-1

Si (i < 3)

i=i+1

SiNo

i = 0

string codigoConectarStdcodigoConectar

string codigoStdcodigo

Si (codigoConectarStd == codigoStd)

MessageBoxANULL, "No puedes usar tu mismo codigo, para conectarte con un amigo deben intercambiar sus codigos...", "Atención!", MB\_OK

PrintColorXy">%s<-", "XXXX", 31, 22, 12, 15=PrintColorXy">%s<-", "XXXX", 31, 22, 12, 15-1

codigoConectar[0] = 'X'

codigoConectar[1] = 'X'

codigoConectar[2] = 'X'

codigoConectar[3] = 'X'

SiNo

ErrorAlConectarResult = false

Si (Conectar(codigoConectar))

MITURNO = false

AsyncCodigo = false

20Esperar 0 segundos

RedibujarMarcos

CargarCavariableReservadallasMultiplayerapodo, contrincante

SiNo

Si (ErrorAlConectarResult)

FinSi

PrintColorXy">%s<-", codigoConectar, 31, 22, 0, 15=PrintColorXy">%s<-", codigoConectar, 31, 22, 0, 15-1

codigoConectar[0] = 'X'

codigoConectar[1] = 'X'

codigoConectar[2] = 'X'

codigoConectar[3] = 'X'

MessageBoxANULL, "Codigo Incorrecto!", "Atención!", MB\_OK

FinSi

FinSi

FinSi

SiNo

AsyncCodigo = false

FinSi

FinMientras

AsyncCodigo = false

20Esperar 0 segundos

RedibujarMarcos

CargarCavariableReservadallasMultiplayerapodo, contrincante

FinSubProceso

SubProceso ESCRIBIR(\* CODIGO, \* APODO, DISPONIBLE, TURNO, BUSCANDO, \* CAVARIABLERESERVADALLAS)

\_bstr\_t strCodigo

\_bstr\_t strAge

\_bstr\_t strDOB

\_bstr\_t strSalary

\_bstr\_t strBuscando

\_bstr\_t strCavariableReservadallas

\_ConnectionPtr pConn = NULL

\_bstr\_t strConCadenaCon

HRESULT hr = S\_OK

CoInitializeNULL

try

hr = pConn.CreateInstance(\_\_uuidof(Connection))

Si (FAILED(hr))

CoUninitialize

FinSi

hr = pConn->OpenstrCon, "", "", 0

Si (FAILED(hr))

CoUninitialize

FinSi

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

strCodigo = result

strcpyresult2, ",\'"

strcatresult2, apodo

strcatresult2, "\',"

strAge = result2

result3 = appendCharToCharArrayresult3, turno

strcatresult3, ","

strDOB = result3

result4 = appendCharToCharArrayresult4, disponible

strcatresult4, ","

strSalary = result4

result5 = appendCharToCharArrayresult5, buscando

strcatresult5, ","

strBuscando = result5

strcpyresult6, "\'"

strcatresult6, cavariableReservadallas

strcatresult6, "\')"

strCavariableReservadallas = result6

\_bstr\_t strSQL"INSERT INTO JUEGOS(CODIGO,APODO,TURNO,DISPONIBLE,BUSCANDO,CAvariableReservadaLLAS) Values("

strSQL += strCodigo + strAge + strDOB + strSalary + strBuscando + strCavariableReservadallas

pConn->ExecutestrSQL, NULL, adExecuteNoRecords

pConn->Close

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

FinSubProceso

SubProceso \* APPENDTOARRAY(\* ARRAY, A)

variableReservadaze\_t len = strlenarray

strcpyret, array

ret[len] = a

ret[len + 1] = '\0'

FinSubProceso

SubProceso EXISTECODIGO(\* CODIGO)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t strCnnCadenaCon

\_RecordsetPtr pRstAuthors = NULL

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

Si (FAILED(hr)) //Comprueba que se pueda crear la instancia

FinSi

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

code = result

\_bstr\_t strSQL"SELECT [CODIGO] FROM [JUEGOS] WHERE [CODIGO]="

strSQL += code

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

\_bstr\_t valField1

pRstAuthors->MoveFirst

Si (!pRstAuthors->EndOfFile)

Mientras !pRstAuthors->EndOfFile

valField1 = pRstAuthors->Fields->GetItem"CODIGO"->Value

pRstAuthors->MoveNext

FinMientras

SiNo

FinSi

FinSubProceso

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

SubProceso CONECTAR(\* CODIGO)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t cavariableReservadallaR

\_bstr\_t contrincanteR

\_bstr\_t strCnnCadenaCon

\_RecordsetPtr pRstAuthors = NULL

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

Si (FAILED(hr))

FinSi

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

code = result

\_bstr\_t strSQL"SELECT [CAvariableReservadaLLAS],[APODO] FROM [JUEGOS] WHERE DISPONIBLE=1 AND [CODIGO]="

strSQL += code

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

\_bstr\_t valField1

pRstAuthors->MoveFirst

Si (!pRstAuthors->EndOfFile)

Mientras !pRstAuthors->EndOfFile

cavariableReservadallaR = pRstAuthors->Fields->GetItem"CAvariableReservadaLLAS"->Value

contrincanteR = pRstAuthors->Fields->GetItem"APODO"->Value

static std::string strcontrincanteR

ErrorAlConectarResult = false

Ocuparcode

Si (ErrorAlConectar)

FinSi

codigo = codigoConectar

pRstAuthors->MoveNext

FinMientras

SiNo

FinSi

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

FinSubProceso

SubProceso OCUPAR(\_BSTR\_T CODE)

\_ConnectionPtr pConn = NULL

\_bstr\_t strConCadenaCon

\_bstr\_t CodigoActualcodigo

HRESULT hr = S\_OK

CoInitializeNULL

try

hr = pConn.CreateInstance(\_\_uuidof(Connection))

Si (FAILED(hr))

CoUninitialize

FinSi

hr = pConn->OpenstrCon, "", "", 0

Si (FAILED(hr))

CoUninitialize

FinSi

\_bstr\_t strSQL"UPDATE JUEGOS SET [DISPONIBLE]=0,[BUSCANDO]=0,FECHA=NULL WHERE [CODIGO]="

strSQL += code + "

DELETE FROM JUEGOS WHERE [CODIGO]='" + codigo + "'"

pConn->ExecutestrSQL, NULL, adExecuteNoRecords

pConn->Close

myCharArray = \_com\_util::ConvertBSTRToStringcode

::SysFreeStringcode

codigo[0] = myCharArray[1]

codigo[1] = myCharArray[2]

codigo[2] = myCharArray[3]

codigo[3] = myCharArray[4]

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

FinSubProceso

SubProceso OCUPADO(\* CODIGO)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t strCnnCadenaCon

\_RecordsetPtr pRstAuthors = NULL

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

Si (FAILED(hr))

FinSi

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

code = result

\_bstr\_t strSQL"SELECT [DISPONIBLE], FROM [JUEGOS] WHERE [CODIGO]="

strSQL += code

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

\_bstr\_t valField1

pRstAuthors->MoveFirst

Si (!pRstAuthors->EndOfFile)

Mientras !pRstAuthors->EndOfFile

valField1 = pRstAuthors->Fields->GetItem"DISPONIBLE"->Value

std::string disponiblevalField1

Si (disponible.compare(0) == 0)

MITURNO = true

SiNo

FinSi

pRstAuthors->MoveNext

FinMientras

SiNo

FinSi

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

FinSubProceso

SubProceso PRESSKEY()

//SetForegroundWindow(hWnd)

FinSubProceso

SubProceso CARGARCAVARIABLERESERVADALLASMULTIPLAYER(\* APODO, \* CONTRINCANTE, ALEATORIO)

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0

PrintColorXy"%s", " \n", 31, 7, 2, 15

PrintColorXy"%s", " \n", 31, 8

PrintColorXy"%s", " \n", 31, 9

PrintColorXy"%s", " \n", 31, 10

PrintColorXy"%s", " \n", 31, 11

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy"%s", "\xBA", 31, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy" TURNO DE:%s", (turno == MITURNO ? apodo : contrincante), 30, 31, 15, 0

PrintColorXy"%s", "\xBA", 63, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

Hacer

tecla = 0

AsyncTurno = true

Esperar Tecla

Si (tecla == FLECHA)

Esperar Tecla

Segun tecla Hacer

ARRIBA:

Si (seleccion == 0 || seleccion == 1 || seleccion == 2)

seleccion += 6

SiNo

seleccion -= 3

FinSi

DERECHA:

Si (seleccion == 5 || seleccion == 2 || seleccion == 8)

seleccion -= 2

SiNo

seleccion=seleccion+1

FinSi

ABAJO:

Si (seleccion == 6 || seleccion == 7 || seleccion == 8)

seleccion -= 6

SiNo

seleccion += 3

FinSi

IZQUIERDA:

Si (seleccion == 6 || seleccion == 3 || seleccion == 0)

seleccion += 2

SiNo

seleccion=seleccion-1

FinSi

FinSegun

SiNo

Si (tecla == ENTER)

Si (turno != MITURNO || !puedeTirar)

MessageBoxANULL, "No es tu turno...", "Atención!", MB\_OK

SiNo

Si (cavariableReservadallas[seleccion] == 0)

cavariableReservadallas[seleccion] = MITURNO ? 1 : 2

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, 15, 0

PrintColorXy"%s", "\xBA", 31, 31, 15, 0

PrintColorXy" TURNO DE:%s", contrincante, 30, 31, 15, 0

PrintColorXy"%s", "\xBA", 63, 31, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, 15, 0

ErrorAlConectarResult = false

EnviarTiro

Si (ErrorAlConectarResult)

FinSi

puedeTirar = false

SiNo

MessageBoxANULL, "CavariableReservadalla Ocupada", "Atención!", MB\_OK

FinSi

FinSi

SiNo

Si (tecla == ESC)

FinSi

FinSi

FinSi

Para int i = 0 Hasta Con Paso 1

Segun i Hacer

0:

tempx = 0

1:

tempx = 11

2:

tempx = 22

3:

tempx = 0

4:

tempx = 11

5:

tempx = 22

6:

tempx = 0

7:

tempx = 11

8:

tempx = 22

FinSegun

Si (i >= 6)

temp = 13

SiNo

Si (i <= 2)

temp = 1

SiNo

Si (i >= 3 y i <= 5)

temp = 7

FinSi

FinSi

FinSi

Segun cavariableReservadallas[i] Hacer

0:

PrintColorXy"%s", " \n", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", " \n", (31 + tempx), 7 + temp

PrintColorXy"%s", " \n", (31 + tempx), 8 + temp

PrintColorXy"%s", " \n", (31 + tempx), 9 + temp

PrintColorXy"%s", " \n", (31 + tempx), 10 + temp

1:

PrintColorXy"%s", " \\ \\ / /", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", " \\ V / ", (31 + tempx), 7 + temp

PrintColorXy"%s", " > < ", (31 + tempx), 8 + temp

PrintColorXy"%s", " / . \\ ", (31 + tempx), 9 + temp

PrintColorXy"%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp

2:

PrintColorXy"%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", "| | | |", (31 + tempx), 7 + temp

PrintColorXy"%s", "| | | |", (31 + tempx), 8 + temp

PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp=PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp-1

PrintColorXy"%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp

FinSegun

FinPara

Hasta Que (Gano(1))

FinSubProceso

SubProceso ENVIARTIRO( ALEATORIO)

std::string res = "'"

Para each(int t in cavariableReservadallas)

res += ","

res += std::to\_stringt

FinPara

res += "' WHERE [CODIGO]='"

Si (!aleatorio)

res += codigo

SiNo

res += codigoConectar

FinSi

res += "'"

\_bstr\_t strCavariableReservadallas

\_ConnectionPtr pConn = NULL

\_bstr\_t strConCadenaCon

HRESULT hr = S\_OK

CoInitializeNULL

try

hr = pConn.CreateInstance(\_\_uuidof(Connection))

Si (FAILED(hr))

CoUninitialize

FinSi

hr = pConn->OpenstrCon, "", "", 0

Si (FAILED(hr))

CoUninitialize

FinSi

res.data), res.length(,

NULL, 0

BSTR wsdata = ::SysAllocStringLenNULL, wslen

::MultiByteToWideCharCP\_ACP, 0,

res.data), res.length(,

wsdata, wslen

strCavariableReservadallas = wsdata

\_bstr\_t strSQL"UPDATE JUEGOS SET [TURNO]=~[TURNO],[CAvariableReservadaLLAS]="

strSQL += strCavariableReservadallas

pConn->ExecutestrSQL, NULL, adExecuteNoRecords

pConn->Close

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

FinSubProceso

SubProceso SENDSTRING( H, \*TEXT)

PostMessageh, WM\_CHAR, text[0], 0

FinSubProceso

SubProceso SETCURSOR( H, X, Y)

PostMessageh, WM\_LBUTTONDOWN, MK\_LBUTTON, MAKELPARAM(x, y)

PostMessageh, WM\_LBUTTONUP, MK\_LBUTTON, MAKELPARAM(x, y)

FinSubProceso

SubProceso ACTUALIZARTIROSVZPLAYER( NEWTURNO, ALEATORIO)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t strCnnCadenaCon

\_RecordsetPtr pRstAuthors = NULL

std::string::variableReservadaze\_type sz

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

Si (FAILED(hr))

FinSi

strcpyresult, "\'"

Si (!aleatorio)

strcatresult, codigo

SiNo

strcatresult, codigoConectar

FinSi

strcatresult, "\'"

code = result

\_bstr\_t strSQL"SELECT [CAvariableReservadaLLAS] FROM [JUEGOS] WHERE [CODIGO]="

strSQL += code

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

\_bstr\_t valField1

pRstAuthors->MoveFirst

Si (!pRstAuthors->EndOfFile)

Mientras !pRstAuthors->EndOfFile

valField1 = pRstAuthors->Fields->GetItem"CAvariableReservadaLLAS"->Value

std::string disponiblevalField1

Para each(char c in disponible)

Si (c == 0 || c == 1 || c == 2)

i=i+1

FinSi

FinPara

pRstAuthors->MoveNext

FinMientras

FinSi

FinSubProceso

catch \_com\_error & ce

CoUninitialize

Para int i = 0 Hasta Con Paso 1

Segun i Hacer

0:

tempx = 0

1:

tempx = 11

2:

tempx = 22

3:

tempx = 0

4:

tempx = 11

5:

tempx = 22

6:

tempx = 0

7:

tempx = 11

8:

tempx = 22

FinSegun

Si (i >= 6)

temp = 13

SiNo

Si (i <= 2)

temp = 1

SiNo

Si (i >= 3 y i <= 5)

temp = 7

FinSi

FinSi

FinSi

Segun cavariableReservadallas[i] Hacer

0:

PrintColorXy"%s", " \n", (31 + tempx), 6 + temp, 15, 0

PrintColorXy"%s", " \n", (31 + tempx), 7 + temp

PrintColorXy"%s", " \n", (31 + tempx), 8 + temp

PrintColorXy"%s", " \n", (31 + tempx), 9 + temp

PrintColorXy"%s", " \n", (31 + tempx), 10 + temp

1:

PrintColorXy"%s", " \\ \\ / /", (31 + tempx), 6 + temp, 15, 0

PrintColorXy"%s", " \\ V / ", (31 + tempx), 7 + temp

PrintColorXy"%s", " > < ", (31 + tempx), 8 + temp

PrintColorXy"%s", " / . \\ ", (31 + tempx), 9 + temp

PrintColorXy"%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp

2:

PrintColorXy"%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, 15, 0

PrintColorXy"%s", "| | | |", (31 + tempx), 7 + temp

PrintColorXy"%s", "| | | |", (31 + tempx), 8 + temp

PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp=PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp-1

PrintColorXy"%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp

FinSegun

Si (turno != newturno)

turno = newturno

puedeTirar = true

FinSi

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy"%s", "\xBA", 30, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy"TURNO DE:%s", (turno == MITURNO ? apodo : contrincante), 31, 31, 15, 0

PrintColorXy"%s", "\xBA", 63, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

FinPara

FinSubProceso

SubProceso VZALEATORIO()

turno = true

Para int i = 0 Hasta 2 Con Paso 1

apodo[i] = ' '

FinPara

contrincante = "Jugador 2"

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 25, 15, 0

PrintColorXy"%s", "\xBA", 30, 26

PrintColorXy"%s", " ", 31, 26, 0, 15

PrintColorXy"%s", "\xBA", 64, 26, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 27, 15, 0

PrintColorXy"%s", "Escribe un apodo original

)", 30, 23, 15, 0

Mientras tecla != ENTER

Si (tecla == ESC)

FinSi

Si (tecla != ENTER y tecla != FLECHA y tecla != ESC)

i=i+1

PrintColorXy"%s", apodo, 31, 26, 0, 15

FinSi

FinMientras

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 25, 15, 0

PrintColorXy"%s", "\xBA", 30, 26

PrintColorXy"%s", " ", 31, 26, 0, 15

PrintColorXy"%s", "\xBA", 64, 26, 15, 0

PrintColorXy"%s", "\xBA", 30, 27

PrintColorXy"%s", ">>Conectando con el servidor...<<", 31, 27, 0, 15

PrintColorXy"%s", "\xBA", 64, 27, 15, 0

PrintColorXy"%s", "\xBA", 30, 28

PrintColorXy"%s", " ", 31, 28, 0, 15

PrintColorXy"%s", "\xBA", 64, 28, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 29, 15, 0

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0

AsyncAnimation = true

sprintfcode, "%d", RandRange(0, 9)

TempCode = appendCharToCharArrayTempCode, code[0]

sprintfcode, "%d", RandRange(0, 9)

TempCode = appendCharToCharArrayTempCode, code[0]

Mientras ExisteCodigoTempCode

TempCode = ""

sprintfcode, "%d", RandRange(0, 9)

TempCode = appendCharToCharArrayTempCode, code[0]

sprintfcode, "%d", RandRange(0, 9)

TempCode = appendCharToCharArrayTempCode, code[0]

FinMientras

PrintColorXy"%s", ">>>>>Registrando Jugador<<<<<<", 31, 27, 0, 15

codigo[0] = TempCode[0]

codigo[1] = TempCode[1]

codigo[2] = TempCode[2]

codigo[3] = TempCode[3]

ErrorAlConectarResult = false

EscribirAleatoriocodigo, apodo

Si (ErrorAlConectarResult)

FinSi

AsyncAnimation = false

10Esperar 0 segundos

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0

AsyncAnimation = true

AsyncAleatorioActive = true

ErrorAlConectarResult = false

Mientras !ConectarAleatoriocodigo && !AleatorioOcupadocodigo && !ErrorAlConectarResult

Esperar Tecla

Si (tecla == ESC)

AsyncAleatorioActive = false

FinSi

FinMientras

Si (ErrorAlConectarResult)

FinSi

AsyncAnimation = false

AsyncAleatorioActive = false

20Esperar 0 segundos

RedibujarMarcos

CargarCavariableReservadallasMultiplayerAleatorioapodo, contrincante

FinSubProceso

SubProceso ESCRIBIRALEATORIO(\* CODIGO, \* APODO, DISPONIBLE, TURNO, BUSCANDO, \* CAVARIABLERESERVADALLAS)

\_bstr\_t strCodigo

\_bstr\_t strAge

\_bstr\_t strDOB

\_bstr\_t strSalary

\_bstr\_t strBuscando

\_bstr\_t strCavariableReservadallas

\_ConnectionPtr pConn = NULL

\_bstr\_t strConCadenaCon

HRESULT hr = S\_OK

CoInitializeNULL

try

hr = pConn.CreateInstance(\_\_uuidof(Connection))

Si (FAILED(hr))

CoUninitialize

FinSi

hr = pConn->OpenstrCon, "", "", 0

Si (FAILED(hr))

CoUninitialize

FinSi

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

strCodigo = result

strcpyresult2, ",\'"

strcatresult2, apodo

strcatresult2, "\',"

strAge = result2

result3 = appendCharToCharArrayresult3, turno

strcatresult3, ","

strDOB = result3

result4 = appendCharToCharArrayresult4, disponible

strcatresult4, ","

strSalary = result4

result5 = appendCharToCharArrayresult5, buscando

strcatresult5, ","

strBuscando = result5

strcpyresult6, "\'"

strcatresult6, cavariableReservadallas

strcatresult6, "\')"

strCavariableReservadallas = result6

\_bstr\_t strSQL"INSERT INTO JUEGOS(CODIGO,APODO,TURNO,DISPONIBLE,BUSCANDO,CAvariableReservadaLLAS) Values("

strSQL += strCodigo + strAge + strDOB + strSalary + strBuscando + strCavariableReservadallas

pConn->ExecutestrSQL, NULL, adExecuteNoRecords

pConn->Close

FinSubProceso

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

FinSubProceso

SubProceso CONECTARALEATORIO(\* CODE)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t contrincanteR

\_bstr\_t strCnnCadenaCon

\_RecordsetPtr pRstAuthors = NULL

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

Si (FAILED(hr))

FinSi

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

code = result

\_bstr\_t strSQL"SELECT [APODO],[CODIGO] FROM [JUEGOS] WHERE [DISPONIBLE]=1 AND [BUSCANDO]=1 AND DATEDIFF(minute, [FECHA], SYSDATETIME())<=5 AND [CODIGO]!="

strSQL += code

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

\_bstr\_t valField1

pRstAuthors->MoveFirst

Si (!pRstAuthors->EndOfFile)

Mientras !pRstAuthors->EndOfFile

contrincanteR = pRstAuthors->Fields->GetItem"APODO"->Value

code = pRstAuthors->Fields->GetItem"CODIGO"->Value

static std::string strcontrincanteR

static std::string str2code

codigoConectar[0] = tempcode[0]

codigoConectar[1] = tempcode[1]

codigoConectar[2] = tempcode[2]

codigoConectar[3] = tempcode[3]

ErrorAlConectarResult = false

pRstAuthors->MoveNext

FinMientras

SiNo

pressKey

FinSi

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

FinSubProceso

SubProceso OCUPARALEATORIO(\* CODE)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t codeConectar

\_bstr\_t contrincanteR

\_bstr\_t strCnnCadenaCon

\_RecordsetPtr pRstAuthors = NULL

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

Si (FAILED(hr))

FinSi

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

code = result

strcpyresult1, "\'"

strcatresult1, codigoConectar

strcatresult1, "\'"

codeConectar = result1

\_bstr\_t strSQL"UPDATE JUEGOS SET DISPONIBLE=0,BUSCANDO=0,FECHA=NULL WHERE [CODIGO]="

strSQL += codeConectar + "

DELETE FROM JUEGOS WHERE [CODIGO]=" + code + "

"

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

turno = true

MITURNO = false

FinSubProceso

catch \_com\_error & ce

ErrorAlConectar

CoUninitialize

FinSubProceso

SubProceso ALEATORIOOCUPADO(\* CODE)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t disponible

\_bstr\_t strCnnCadenaCon

\_RecordsetPtr pRstAuthors = NULL

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

Si (FAILED(hr))

pressKey

FinSi

strcpyresult, "\'"

strcatresult, codigo

strcatresult, "\'"

code = result

\_bstr\_t strSQL"SELECT [DISPONIBLE] FROM [JUEGOS] WHERE [BUSCANDO]=0 AND [DISPONIBLE]=0 AND [FECHA] IS NULL AND [CODIGO]="

strSQL += code

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

\_bstr\_t valField1

pRstAuthors->MoveFirst

Si (!pRstAuthors->EndOfFile)

Mientras !pRstAuthors->EndOfFile

turno = true

MITURNO = true

pRstAuthors->MoveNext

FinMientras

SiNo

pressKey

FinSi

FinSubProceso

catch \_com\_error & ce

ErrorAlConectar

pressKey

CoUninitialize

PressKey

FinSubProceso

SubProceso ASYNCALEATORIO(\* DATA)

Mientras 0 == 0

Si (AsyncAleatorioActive)

HRESULT hr = S\_OK

try

CoInitializeNULL

\_bstr\_t code

\_bstr\_t strCnnCadenaCon

\_bstr\_t valField1

\_RecordsetPtr pRstAuthors = NULL

hr = pRstAuthors.CreateInstance\_\_uuidof(Recordset)

std::string strcodigoConectar

Si (str.compare(XXXX) == 0)

codigoConectar[0] = codigo[0]

codigoConectar[1] = codigo[1]

codigoConectar[2] = codigo[2]

codigoConectar[3] = codigo[3]

codigoConectar[4] = codigo[4]

FinSi

strcpyresult, "\'"

strcatresult, codigoConectar

strcatresult, "\'"

code = result

\_bstr\_t strSQL"SELECT [TURNO] FROM [JUEGOS] WHERE [CODIGO]="

strSQL += code

pRstAuthors->OpenstrSQL, strCnn, adOpenStatic, adLockReadOnly, adCmdText

pRstAuthors->MoveFirst

Si (!pRstAuthors->EndOfFile)

Mientras !pRstAuthors->EndOfFile

valField1 = pRstAuthors->Fields->GetItem"TURNO"->Value

std::string disponiblevalField1

std::string::variableReservadaze\_type sz

Si (turno != newturno)

Si (!ActualizarTirosVzPlayer(newturno true))

AsyncAleatorioActive = false

FinSi

FinSi

pRstAuthors->MoveNext

FinMientras

FinSi

FinSi

catch \_com\_error & ce

ErrorAlConectarResult = true

ErrorAlConectar

FinMientras

CoUninitialize

SiNo

2000Esperar 2 segundos

FinSi

FinSubProceso

SubProceso CARGARCAVARIABLERESERVADALLASMULTIPLAYERALEATORIO(\* APODO, \* CONTRINCANTE)

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCB\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 6, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 7, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 8, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 9, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 10, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 11, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 12, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 13, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 14, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 15, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 16, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 17, 15, 0

PrintColorXy"%s", "\xCC\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCE\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xB9", 30, 18, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 19, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 20, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 21, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 22, 15, 0

PrintColorXy"%s", "\xBA \xBA \xBA \xBA", 30, 23, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCA\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 24, 15, 0

PrintColorXy"%s", " \n", 31, 7, 2, 15

PrintColorXy"%s", " \n", 31, 8

PrintColorXy"%s", " \n", 31, 9

PrintColorXy"%s", " \n", 31, 10

PrintColorXy"%s", " \n", 31, 11

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy"%s", "\xBA", 30, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy"TURNO DE:%s", (turno == MITURNO ? apodo : contrincante), 32, 31, 15, 0

PrintColorXy"%s", "\xBA", 63, 31, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, (turno == MITURNO ? 9 : 15), (turno == MITURNO ? 15 : 0)

Hacer

tecla = 0

ErrorAlConectarResult = false

AsyncAleatorioActive = true

Si (ErrorAlConectarResult)

FinSi

Esperar Tecla

Si (tecla == FLECHA)

Esperar Tecla

Segun tecla Hacer

ARRIBA:

Si (seleccion == 0 || seleccion == 1 || seleccion == 2)

seleccion += 6

SiNo

seleccion -= 3

FinSi

DERECHA:

Si (seleccion == 5 || seleccion == 2 || seleccion == 8)

seleccion -= 2

SiNo

seleccion=seleccion+1

FinSi

ABAJO:

Si (seleccion == 6 || seleccion == 7 || seleccion == 8)

seleccion -= 6

SiNo

seleccion += 3

FinSi

IZQUIERDA:

Si (seleccion == 6 || seleccion == 3 || seleccion == 0)

seleccion += 2

SiNo

seleccion=seleccion-1

FinSi

FinSegun

SiNo

Si (tecla == ENTER)

Si (turno != MITURNO || !puedeTirar)

MessageBoxANULL, "No es tu turno...", "Atención!", MB\_OK

SiNo

Si (cavariableReservadallas[seleccion] == 0)

cavariableReservadallas[seleccion] = MITURNO ? 1 : 2

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 30, 30, 15, 0

PrintColorXy"%s", "\xBA", 30, 31, 15, 0

PrintColorXy"TURNO DE:%s", contrincante, 31, 31, 15, 0

PrintColorXy"%s", "\xBA", 63, 31, 15, 0

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 30, 32, 15, 0

ErrorAlConectarResult = false

EnviarTirotrue

Si (ErrorAlConectarResult)

FinSi

puedeTirar = false

SiNo

MessageBoxANULL, "CavariableReservadalla Ocupada", "Atención!", MB\_OK

FinSi

FinSi

SiNo

Si (tecla == ESC)

FinSi

FinSi

FinSi

Para int i = 0 Hasta Con Paso 1

Segun i Hacer

0:

tempx = 0

1:

tempx = 11

2:

tempx = 22

3:

tempx = 0

4:

tempx = 11

5:

tempx = 22

6:

tempx = 0

7:

tempx = 11

8:

tempx = 22

FinSegun

Si (i >= 6)

temp = 13

SiNo

Si (i <= 2)

temp = 1

SiNo

Si (i >= 3 y i <= 5)

temp = 7

FinSi

FinSi

FinSi

Segun cavariableReservadallas[i] Hacer

0:

PrintColorXy"%s", " \n", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", " \n", (31 + tempx), 7 + temp

PrintColorXy"%s", " \n", (31 + tempx), 8 + temp

PrintColorXy"%s", " \n", (31 + tempx), 9 + temp

PrintColorXy"%s", " \n", (31 + tempx), 10 + temp

1:

PrintColorXy"%s", " \\ \\ / /", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", " \\ V / ", (31 + tempx), 7 + temp

PrintColorXy"%s", " > < ", (31 + tempx), 8 + temp

PrintColorXy"%s", " / . \\ ", (31 + tempx), 9 + temp

PrintColorXy"%s", " /\_\_/ \\\_\_\\", (31 + tempx), 10 + temp

2:

PrintColorXy"%s", " / \_\_ \\ ", (31 + tempx), 6 + temp, (seleccion == i ? 2 : 15), (seleccion == i ? 15 : 0)

PrintColorXy"%s", "| | | |", (31 + tempx), 7 + temp

PrintColorXy"%s", "| | | |", (31 + tempx), 8 + temp

PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp=PrintColorXy"%s", "| `' |", 31 + tempx, 9 + temp-1

PrintColorXy"%s", " \\\_\_\_\_\_\_/ ", (31 + tempx), 10 + temp

FinSegun

FinPara

Hasta Que (Gano(1))

FinSubProceso

SubProceso ERRORALCONECTAR()

AsyncAleatorioActive = false

AsyncAnimation = false

AsyncCodigo = false

AsyncMarco = false

AsyncTurno = false

ErrorAlConectarResult = true

RedibujarMarcos4

PrintColorXy"%s", "\xC9\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBB", 15, 10, 15, 0

PrintColorXy"%s", "\xBA", 15, 11, 15, 4

PrintColorXy"%s", "Error al conectar con el servidor comprueba tu conexi\xA2n a internet... ", 16, 11, 15, 4

PrintColorXy"%s", "\xBA", 86, 11, 15, 4

PrintColorXy"%s", "\xC8\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xCD\xBC", 15, 12, 15, 0

PrintColorXy"%s", " ¶¶", 24, 13, 4, 14

PrintColorXy"%s", " ¶¶¶¶", 24, 14, 4, 14

PrintColorXy"%s", " ¶¶¶¶¶¶", 24, 15, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶", 24, 16, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶", 24, 17, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶", 24, 18, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶

¶¶¶", 24, 19, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶¶¶

¶¶¶", 24, 20, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶¶¶

¶¶¶", 24, 21, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶¶¶

¶¶¶", 24, 22, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶¶¶

¶¶", 24, 23, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶¶

¶¶", 24, 24, 4, 14

PrintColorXy"%s", " ¶¶

¶¶¶¶

¶¶", 24, 25, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶

¶¶¶", 24, 26, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶

¶¶¶", 24, 27, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶

¶¶¶", 24, 28, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶", 24, 29, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶¶

¶¶¶", 24, 30, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶¶¶

¶¶¶", 24, 31, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶¶¶

¶¶¶", 24, 32, 4, 14

PrintColorXy"%s", " ¶¶¶

¶¶¶", 24, 33, 4, 14

PrintColorXy"%s", "¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶", 24, 34, 4, 14

Esperar Tecla

RedibujarMarcos

conectado = false

puedeTirar = true

MITURNO = false

turno = false

FinSubProceso

Diagrama



Análisis

Este programa esta diseñado para jugar gato tanto indivual con la modalidad “Vz Cpu” como con amigos por medio de conexión a internet y una base de datos que matiene ambos juegos.

Tambien incluye una opcion para combatir con un oponente al azar.

Problemas

Hubo algunos problemas para enviar las consultas de la base de datos a el servidor directamente.

Problema donde se indicaba que era tu turno de tirar pero no se permitia hacerlo.

Complicaciones mateniendo los procesos asincrónicos para las animaciones en el fondo.

Problemas para imprimir correctamente los textos en las coordenadas especificas de la pantalla antes de que otro proceso llamara a el buffer de la consola.

Compilación

Se utilizo Visual Studio 2017 para la compilación y depuración.

Mantenimiento

Limpieza y renovación de la base de datos.

Correción de errores que podrían llegar a encontrarse mas adelante.

Funcionamiento

Para su correcto funcionamiento debe utilizarse el programa únicamente con las flechas de el teclado y las teclas Enter y Esc para aceptar las opciones y salir de de la función respectivamente.

Se requiere una conexión a internet para las modalidades con amigos o contra algun jugador aleatorio.