Unit Unit1;

{$mode objfpc}{$H+}

Interface

{Aparte de las unidades utilizadas de forma natural por las FORMS de pascal, se agrega la unidad

"LCLType", que permite el obtener las teclas presionadas por el usuario en un formato intuitivo y mas

facil de leer en el programa, junto a la unidad "MMSystem", que da paso a poder usar el comaundo "PlaySound",

el cual se utiliza para reproducir sonidos a partir de un archivo de audio.}

Uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, StdCtrls,

ComCtrls, Menus, Grids, crt, LCLType, ValEdit, CheckLst,MMSystem;

Type

{ TProjecT }

//La clase que representa al "ProjecT" (Projecto Tetris) dentro del panorama de objetos de pascal.

TProjecT = Class(TForm)

BuscarTop5: TLabel;

ComboBox1: TComboBox;

LeyendaPartida1: TLabel;

ListaPartidas1: TListBox;

PaisUsuarioLabel: TLabel;

NombreRealJugadorLabel: TLabel;

Partidas1: TPanel;

VolverAtras: TLabel;

BuscarUsuario1: TLabel;

LeyendaPartida: TLabel;

ListaPartidas: TListBox;

Partidas: TPanel;

UsuarioABuscar: TEdit;

EmpezarAJugar: TPanel;

ClaveIS: TEdit;

ClaveR: TEdit;

Clave\_Inicio1: TLabel;

Clave\_Registro: TLabel;

MostrarContraseniaLabel: TLabel;

Paises: TComboBox;

CorreoR: TEdit;

Correo\_Registro: TLabel;

EnviarDatos: TButton;

Extras: TLabel;

DownButton: TImage;

Capt\_porMov: TLabel;

Cpt\_PorTmpo: TLabel;

CambiarDeModo: TPanel;

ExplicacionDelModo1: TLabel;

BloquesPiezas: TImageList;

img\_LogoTetris: TImage;

Img\_GameOver: TImage;

Jugador: TLabel;

Jugar: TLabel;

Panel1: TPanel;

Tmp\_MusicaMenu: TTimer;

VolverAlMenu: TLabel;

ReqCaractEspeciales: TLabel;

Req4Iguales: TLabel;

ReqTamanio: TLabel;

ReqNoEnie: TLabel;

ReqNum: TLabel;

ReqMin: TLabel;

RequerimientosClave: TPanel;

ReqMayus: TLabel;

MensajeErrorJugar: TLabel;

GameOver: TTabSheet;

Panel10: TPanel;

TiempoLabel: TLabel;

NombreJugadorLabel: TLabel;

MovimientosLabel: TLabel;

PuntosPasadosLabel: TLabel;

Pausa: TImage;

PuntajeLabel: TLabel;

TableroPanel: TPanel;

Tablero: TImage;

Img\_DirectionalArrow: TImage;

Img\_SandClock: TImage;

LabelCantidades: TLabel;

Panel9: TPanel;

Tetris: TTabSheet;

Tmp\_EfectoPasoPuntos: TTimer;

Tmp\_Ctrl\_EfectoDerrota: TTimer;

Tmp\_Ctrl\_TiempoJuego: TTimer;

Tmp\_CaidaFilas: TTimer;

Tmp\_GameCicle: TTimer;

Tmp\_Gravedad: TTimer;

VolverAJugar: TLabel;

VolverAtras1: TLabel;

x1: TPanel;

PorMovimientos: TPanel;

PorTiempo: TPanel;

UpButton: TImage;

Img\_SelecPiezas: TImageList;

IngresePiezas: TLabel;

IngresePiezas1: TLabel;

iniciarsesion1: TButton;

Iniciodeseion: TLabel;

Buscar\_Jugador: TLabel;

Label2: TLabel;

Mensaje\_Usuario1: TLabel;

Mensaje\_Usuario2: TLabel;

Mensaje\_Usuario3: TLabel;

ErrorInicioSesion: TLabel;

Mostrarclave: TCheckBox;

NombreR: TEdit;

NyP\_Registro: TLabel;

MenuInicial: TTabSheet;

Panel7: TPanel;

Panel8: TPanel;

Pieza1: TPanel;

Pieza2: TPanel;

Pieza3: TPanel;

Pieza4: TPanel;

Pieza5: TPanel;

Pieza6: TPanel;

Pieza7: TPanel;

Pieza8: TPanel;

PiezaMostrar1: TPanel;

PiezaMostrar2: TPanel;

PiezaMostrar3: TPanel;

PiezaMostrar4: TPanel;

PiezaMostrar5: TPanel;

Registro: TTabSheet;

InicioSesion: TTabSheet;

PaginaExtras: TTabSheet;

Paginas: TPageControl;

Pais\_Registro: TLabel;

MenuPrincipal: TPanel;

Panel2: TPanel;

Panel3: TPanel;

Panel4: TPanel;

Panel5: TPanel;

Panel6: TPanel;

registrarse: TLabel;

Salir: TLabel;

DiagramaDeGantt: TTabSheet;

SeleccionPiezas: TTabSheet;

SeleccionModos: TTabSheet;

InfoJugador: TTabSheet;

top5: TTabSheet;

Timer1: TTimer;

GenerarTop5: TLabel;

Tmp\_SelecPiezas: TTimer;

UsuarioIS: TEdit;

UsuarioR: TEdit;

Usuario\_Inicio1: TLabel;

GenerarDiagramaDeGantt: TLabel;

Usuario\_Registro: TLabel;

x10: TPanel;

x5: TPanel;

Procedure UsuarioABuscarClick(Sender: TObject);

Procedure VolverAtrasClick(Sender: TObject);

Procedure Buscar\_JugadorClick(Sender: TObject);

Procedure CambiarDeModoClick(Sender: TObject);

Procedure PaisesChange(Sender: TObject);

Procedure DownButtonClick(Sender: TObject);

Procedure EmpezarAJugarClick(Sender: TObject);

Procedure EnviarDatosClick(Sender: TObject);

Procedure ExtrasClick(Sender: TObject);

Procedure FormCreate(Sender: TObject);

Procedure FormKeyDown(Sender: TObject; Var Key: Word; Shift: TShiftState);

Procedure FormKeyUp(Sender: TObject; Var Key: Word; Shift: TShiftState);

Procedure IngresePiezas1Click(Sender: TObject);

Procedure iniciarsesion1Click(Sender: TObject);

Procedure IniciodeseionClick(Sender: TObject);

Procedure JugarClick(Sender: TObject);

Procedure MostrarclaveClick(Sender: TObject);

Procedure Pieza1Click(Sender: TObject);

Procedure Pieza2Click(Sender: TObject);

Procedure Pieza3Click(Sender: TObject);

Procedure Pieza4Click(Sender: TObject);

Procedure Pieza5Click(Sender: TObject);

Procedure Pieza6Click(Sender: TObject);

Procedure Pieza7Click(Sender: TObject);

Procedure Pieza8Click(Sender: TObject);

Procedure PiezaMostrar1Click(Sender: TObject);

Procedure PiezaMostrar2Click(Sender: TObject);

Procedure PiezaMostrar3Click(Sender: TObject);

Procedure PiezaMostrar4Click(Sender: TObject);

Procedure PiezaMostrar5Click(Sender: TObject);

Procedure PorMovimientosClick(Sender: TObject);

Procedure PorMovimientosMouseEnter(Sender: TObject);

Procedure PorMovimientosMouseLeave(Sender: TObject);

Procedure PorTiempoClick(Sender: TObject);

Procedure PorTiempoEnter(Sender: TObject);

Procedure PorTiempoExit(Sender: TObject);

Procedure PorTiempoMouseEnter(Sender: TObject);

Procedure PorTiempoMouseLeave(Sender: TObject);

Procedure registrarseClick(Sender: TObject);

Procedure RequerimientosClaveClick(Sender: TObject);

Procedure SeleccionPiezasShow(Sender: TObject);

Procedure TableroClick(Sender: TObject);

Procedure Timer1Timer(Sender: TObject);

Procedure Tmp\_Ctrl\_EfectoDerrotaTimer(Sender: TObject);

Procedure Tmp\_Ctrl\_TiempoJuegoTimer(Sender: TObject);

Procedure Tmp\_CaidaFilasTimer(Sender: TObject);

Procedure Tmp\_EfectoPasoPuntosTimer(Sender: TObject);

Procedure Tmp\_GameCicleTimer(Sender: TObject);

Procedure Tmp\_GravedadTimer(Sender: TObject);

Procedure Tmp\_MusicaMenuTimer(Sender: TObject);

Procedure GenerarDiagramaDeGanttClick(Sender: TObject);

Procedure UpButtonClick(Sender: TObject);

Procedure VolverAJugarClick(Sender: TObject);

Procedure VolverAlMenuClick(Sender: TObject);

Procedure x10Click(Sender: TObject);

Procedure x1Click(Sender: TObject);

Procedure x5Click(Sender: TObject);

Private

Public

End;

{Un tipo de dato utilizado para las variables que vayan a guardar datos individuales obtenidos

al leer una linea de un archivo secuencial, siendo que cada indice sera un dato separado de

la linea completa de informacion}

InfoHolder = Array [1..8] Of string [80];

//Tipo de dato utilizado unicamente para ordenar mejor la fecha y hora en la que se realiza una partida

Fecha\_Hora\_Completa = Record

Dia, mes, anio: word;

Hora: word;

Minutos, segundos, Milis: word;

End;

//El tipo de registro base para la variable que guarda los datos importantes de cada partida jugada

Partida = Record

Codigo: String[10];

Jugador: ShortString;

FechaYHora: Fecha\_Hora\_Completa;

Puntos: Integer;

End;

Const

//El codigo de color del "Azul" que se decidio usar en varias partes del codigo

Azul = $00C08000;

{Un arreglo constante que guarda cada una de las piezas disponibles en el juego, escritas en orden numerico.

Cada pieza esta declarada de tal forma que cada una se encuentra dividida en "Laminas"}

Piezas: Array [1..8,1..4,1..4] Of integer = (

((1,1,1,1),(0,0,0,0),(0,0,0,0),(0,0,0,0)),

((2,0,0,0),(2,0,0,0),(2,0,0,0),(2,0,0,0)),

((3,0,0,0),(3,0,0,0),(3,3,0,0),(0,0,0,0)),

((0,4,0,0),(0,4,0,0),(4,4,0,0),(0,0,0,0)),

((0,5,0,0),(5,5,0,0),(5,0,0,0),(0,0,0,0)),

((0,6,6,0),(0,6,6,0),(0,0,0,0),(0,0,0,0)),

((7,7,0,0),(0,7,7,0),(0,0,0,0),(0,0,0,0)),

((0,8,0,0),(8,8,8,0),(0,0,0,0),(0,0,0,0))

);

Var

{Variable que representa toda la FORM del juego}

ProjecT: TProjecT;

{Las Variables a continuacion son aquellas que no pueden ser locales al ser utilizadas

en mas de un procedimiento aislado, por lo que es necesario tenerlas disponibles de

forma global.}

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Variables globales relacionados a los archivos del juego \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

//Declaracion de los archivos secuenciales que se van a usar en el Programa.

Jugadores: Text;

Juegos: Text;

//Se declara una variable que va a guardar los datos de cada partida.

Partida\_Actual: Partida;

//La variable que va a guardar el nombre de usuario del jugador mientras tenga iniciada la sesion.

Nombre\_Usuario: ShortString;

//Variable que va a guardar cada elemento de una linea de los archivos secuenciales. Funciona en conjunto con la variable "Linea".

Holders: InfoHolder;

//Una variable auxiliar que guarda cada linea de los archivos secuenciales para su lectura.

Linea: String;

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Contadores, acumuladores o controladores universales \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

{Son variables que cambian de valores a traves de todo el programa para evitar el uso de variables individuales para cada

procedimiento o trozo de codigo}

//Los contadores y acumulador universal.

i, j, k, cont, acum: Integer;

//Un control booleano utilizado en trozos de codigo donde no es necesario guardar el valor de verdad despues de usarse una cantidad determinada de veces.

control: Boolean;

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Variables usadas en la seccion de extras (Diagrama de Gantt, Historial de jugador y Top 5) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Variables usadas en la seccion de seleccion de piezas \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

//Un arreglo que guarda las piezas que se van a usar en partida.

PiezasAUsar: Array[1..5] Of integer;

//Un arreglo de imagenes que iran a representar visualmente la seleccion de piezas del usuario.

Img\_Piezas\_Seleccionadas: Array [1..5] Of TImage;

//Un arreglo de imagenes que representa la seleccion de piezas disponibles para el usuario.

Img\_Piezas\_ASeleccionar: Array[1..8] Of TImage;

//Un arreglo booleano que permite conocer cuales son las piezas de la seleccion que el usuario ha elegido.

PiezaSeleccionada: Array[1..8] Of boolean;

//Una variable que acumula la cantidad de piezas que ha seleccionado el usuario.

CantSelect: 0..5;

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Variables usadas en la seccion de seleccion de modos \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

//Variables que van a guardar La cantidad de tiempo o de movimientos elegidos por el usuario. .

TiempoJuego,LimitMov: Integer;

//Un booleano que permite saber si se ha seleccionado un tipo de juego. Sirve para el control visual.

Seleccionado: Boolean;

//Un rango de 0 a 2 que va a diferenciar los modos de juego, donde 1: por movimientos; 2: por tiempo; 0: No se ha seleccionado ninguno aun.

ModoJuego: 0..2;

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Variables usadas en la seccion de la partida/juego \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

//Arreglo de los bloques a dibujar en pantalla.

Blocks: Array [1..12,1..9] Of Timage;

{El tablero que va a contener los valores logicos (Numericos) de cada pieza.

De este depende el funcionamiento de cada partida.

Se declara con un indice extra de [1..2] al necesitarse dos tableros logicos en partida,

uno que contanga las piezas que ya han caido anteriormente, y otro que contenga la pieza

que cae y es manipulada por el usuario}

Board: Array [1..12,1..9,1..2] Of integer;

//Variables usadas durante la partida.

Fin\_Juego,PiezaCaida,Gravedad,CaidaFilas,pauseBool: Boolean;

PosPiezaInf,Puntaje: Integer;

TeclaActual: Char;

Movimientos,Tiempo,PuntosTotales: Integer;

CondicionDerrota: 1..2;

Implementation

{$R \*.lfm}

{ TProjecT }

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Sector de manejo de archivos \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

{Procedimientos que asignan los archivos Jugadores/Juegos, y se verifica si existen,

y si no lo hacen, se crean y se escriben sus cabeceras en su primera linea.}

Procedure Verificar\_ArchivoJugadores;

Begin

Assign (Jugadores, 'Jugadores.TXT');

If Not (FileExists ('Jugadores.TXT')) Then

Begin

Rewrite (Jugadores);

Writeln(Jugadores,'[NombreDeUsuario][Contrasenia][Nombre][Correo][Pais]');

close(Jugadores);

End;

End;

Procedure Verificar\_ArchivoJuegos;

Begin

Assign (Juegos, 'Juegos.TXT');

If Not (FileExists ('Juegos.TXT')) Then

Begin

Rewrite (Juegos);

Writeln(Juegos,'[Codigo][Usuario][Fecha][Hora][Puntaje][ModoDeJuego][Movimientos][Tiempo]');

close(Juegos);

End;

End;

{Procedimiento que se encarga de particionar cada linea de los archivos.

Recibe la variable "linea" que ya ha leido el archivo deseado, y envia el arreglo "holders"

con cada elemento del "registro" escrito en uno de sus indices.}

Procedure FileReading(P\_Linea:String; Var P\_Holders: InfoHolder);

Begin

control := false;

j := 1;

//Se limpian los holders al tener que realizar el procedimiento una vez por linea.

For k:=1 To 8 Do

P\_Holders[k] := '';

{Se lee toda la linea en busca de los caracteres de control de los archivos (Los corchetes [])

pues su aparicion indica que inicia o termina un elemento, respectivamente. Una vez se guarda

un elemento en un indice de los holders, se pasa al siguiente indice.}

For i:=2 To length(P\_Linea) Do

Begin

If (P\_Linea[i-1]='[')Then

Begin

control := true;

End;

If (P\_Linea[i]=']') Then

Begin

control := false;

j := j+1;

End;

If (control= true) Then

P\_Holders[j] := Holders[j]+P\_Linea[i];

End;

End;

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Sector de La primera inicializacion \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

{Al crearse la FORM que representa al programa, se pasa a la primera pagina, se inicia

la musica del menu, y se llaman a los procedimientos que verifican los archivos}

Procedure TProjecT.FormCreate(Sender: TObject);

Begin

Paginas.ActivePageIndex := 0;

PlaySound('sounds/Menu.wav',1,SND\_ASYNC);

Verificar\_ArchivoJugadores;

Verificar\_ArchivoJuegos;

End;

//El temporizador que hace sonar la musica del menu en Repeticion.

//Activo desde el principio con un intervalo de 6000 milisegundos (Duracion del audio).

Procedure TProjecT.Tmp\_MusicaMenuTimer(Sender: TObject);

Begin

PlaySound('sounds/Menu.wav',1,SND\_ASYNC);

End;

//Inicio y registro de sesión

Procedure IniciodeSesion (Var Salv\_Usuario: shortstring);

Var

Encontrado\_clave, Encontrado\_usuario: Boolean;

Begin

i := 0;

Encontrado\_clave := False;

Encontrado\_usuario := False;

reset (Jugadores);

//Se verifica cada registro dentro del archivo

//para verificar si existe el usuario y si existe, si se corresponde con su contrasenia

//TODO mandar puntero a la segunda linea.

While (Not (seekEOF(Jugadores))) And (encontrado\_Usuario=false) Do

Begin

Readln(Jugadores,Linea);

FileReading(Linea,Holders);

//Se verifica que exista el usuario

If Holders[1] = ProjecT.UsuarioIS.Text Then

Begin

Encontrado\_usuario := true;

//Se verifica que la contrasenia se corresponda con el usuario

If Holders[2] = ProjecT.ClaveIS.Text Then

Begin

Encontrado\_Clave := true;

End;

End;

End;

If Encontrado\_usuario=false Then

ProjecT.ErrorInicioSesion.caption := 'Este usuario no existe'

Else

If Encontrado\_Clave=false Then

ProjecT.ErrorInicioSesion.caption := 'Clave incorrecta';

If (Encontrado\_usuario= true) And (Encontrado\_clave= True) Then

Begin

Project.ErrorInicioSesion.font.color := clwhite;

ProjecT.ErrorInicioSesion.caption := 'Inicio de sesion exitoso';

Salv\_Usuario := Holders[1];

ProjecT.Timer1.Enabled := true;

End;

ProjecT.ErrorInicioSesion.visible := true;

Close (jugadores);

End;

Procedure RegistrodeJugadores;

Const

DotDomains: Array [1..5] Of string = ('.com','.ru','.net','.org','.ve');

Var

auxArroba: String[50];

Encontrado, Clave\_Min, Clave\_enie, Clave\_Esp, Clave\_May, Cad\_Repe,clave\_Num, arroba,dotcom: Boolean;

Aux\_Cad: String [10];

Cont\_CadRepe, i, k: Integer;

Begin

{\*\*\*\*\*\*Inicializando las variables\*\*\*\*\*\*\*}

Clave\_enie := false;

Clave\_Esp := False;

Clave\_May := False;

Clave\_Min := False;

Clave\_Num := False;

Cad\_Repe := False;

arroba := False;

dotcom := false;

Cont\_CadRepe := 1;

Aux\_Cad := '';

i := 0;

k := 0;

{\*\*\*\*\*\*Verificando que el nombre del Usuario\_Registro se encuentre disponible\*\*\*\*\*\*\*}

Encontrado := False;

reset (jugadores);

While (Not (seekEOF(Jugadores))) And (encontrado=false) Do

Begin

Readln(Jugadores,Linea);

FileReading(Linea,Holders);

//el holder 1 contiene el nombre de usuario.

If Holders[1] = ProjecT.UsuarioR.Text Then

Encontrado := true;

End;

{ #todo : Agregar Mensajes para cada error.}

If (Encontrado= true) Then

Begin

ProjecT.Mensaje\_Usuario1.visible := true;

ProjecT.Mensaje\_Usuario1.Caption := 'No disponible';

End

Else

ProjecT.Mensaje\_Usuario1.visible := false;

For i:= 1 To length (ProjecT.CorreoR.Text) Do

If (ProjecT.CorreoR.Text[i] = '@') Then

Arroba := True;

AuxArroba := '';

If Arroba = true Then

Begin

For i:=Pos('@',ProjecT.CorreoR.text) To length (ProjecT.CorreoR.Text) Do

Begin

AuxArroba := AuxArroba+ProjecT.CorreoR.Text[i];

End;

For i:=1 To 5 Do

Begin

If (pos(DotDomains[i],Lowercase(auxArroba)))<>0 Then

dotcom := true;

End;

End;

If (Arroba = false) Or (DotCom = false) Then

Begin

ProjecT.Mensaje\_Usuario2.Caption := 'Correo invalido';

ProjecT.Mensaje\_Usuario2.visible := true;

End

Else

ProjecT.Mensaje\_Usuario2.visible := false;

{ #todo : Agregar sets o conjuntos en constantes para no repetirlos aqui }

For k:=1 To length (ProjecT.ClaveR.Text) Do

Begin

//Clave minuscula

If (ProjecT.ClaveR.Text[k] In ['a'..'z']) And (Clave\_Min=false)Then

Clave\_Min := true;

//Clave mayuscula

If (ProjecT.ClaveR.Text[k] In ['A'..'Z']) And (Clave\_May= false)Then

Clave\_May := True;

//Caracteres especiales

If (ProjecT.ClaveR.Text[k] In ['=', '\*', '-', '\_', '.']) And (Clave\_Esp=false) Then

Clave\_Esp := true;

//

If (ProjecT.ClaveR.Text[k] In ['1'..'9']) Then

clave\_Num := true;

//

If (ProjecT.ClaveR.Text[k] In ['a'..'z', 'A'..'Z', '=', '\*', '-','\_', '.', '0'..'9']) Then

Begin

If (ProjecT.ClaveR.Text[k]=Aux\_Cad) Then

Cont\_CadRepe := Cont\_CadRepe + 1;

If (ProjecT.ClaveR.Text[k]<>Aux\_Cad) Then

Begin

Cont\_CadRepe := 1;

Aux\_Cad := ProjecT.ClaveR.Text[k];

End;

End;

If (Cont\_CadRepe >= 4) Then

Cad\_Repe := True;

End;

//Enie

If (((pos('ñ',ProjecT.ClaveR.Text)) <> 0) Or ((pos('Ñ',ProjecT.ClaveR.Text)) <> 0)) And (Clave\_enie=false) Then

Clave\_enie := true;

If (clave\_Num=false) Or (Clave\_Min=false) Or (Clave\_May=false) Or (Clave\_Esp=false) Or (Clave\_enie=true) Or (Cad\_Repe =true) Or (length (ProjecT.ClaveR.Text) > 10) Or (length (ProjecT.ClaveR.Text) < 8) Then

Begin

ProjecT.Mensaje\_Usuario3.Caption := 'No cumple con los requisitos';

ProjecT.Mensaje\_Usuario3.visible := true;

ProjecT.RequerimientosClave.Height := 330;

ProjecT.RequerimientosClave.Caption := '';

End

Else

ProjecT.Mensaje\_Usuario3.visible := false;

If (length (ProjecT.ClaveR.Text) > 10) Or (length (ProjecT.ClaveR.Text) < 8) Then

Project.ReqTamanio.font.color := clred

Else

Project.ReqTamanio.font.color := azul;

If (clave\_Num=false) Then

ProjecT.ReqNum.font.Color := clred

Else

ProjecT.ReqNum.font.Color := azul;

If (Clave\_Min=false) Then

ProjecT.ReqMin.font.Color := clred

Else

Project.Reqmin.font.color := azul;

If (Clave\_May=false) Then

ProjecT.ReqMayus.font.Color := clred

Else

Project.ReqMayus.font.color := azul;

If (Clave\_Esp=false) Then

ProjecT.ReqCaractEspeciales.font.Color := clred

Else

Project.ReqCaractEspeciales.font.color := azul;

If (Clave\_enie=true) Then

ProjecT.ReqNoEnie.font.Color := clred

Else

Project.ReqNoEnie.font.color := azul;

If (Cad\_Repe =true) Then

ProjecT.Req4Iguales.font.Color := clred

Else

Project.Req4Iguales.font.color := azul;

close (Jugadores);

//Se cumplieron todos los requisitos y se escribe dentro del archivo de jugadores.

If (Clave\_Min=true) And (Clave\_May=true) And (Clave\_Esp=true) And (Clave\_enie=false) And (Cad\_Repe =false) And (arroba= true) And (Encontrado = false) And (length (ProjecT.ClaveR.Text) <= 10) And (length (ProjecT.ClaveR.Text) >= 8)Then

Begin

Append (jugadores);

write (Jugadores,'[',ProjecT.UsuarioR.Text,']');

Write(Jugadores, '[',ProjecT.ClaveR.Text,']');

Write (Jugadores, '[',ProjecT.NombreR.Text,']');

Write (Jugadores, '[',ProjecT.CorreoR.Text,']');

Write (Jugadores,'[',ProjecT.Paises.Text ,']');

Writeln (Jugadores, '');

close (Jugadores);

ProjecT.Paginas.ActivePageIndex := 0;

End;

End;

Procedure TProjecT.registrarseClick(Sender: TObject);

Begin

//Se limpian los cuadros de escritura del usuario

UsuarioR.caption := '';

NombreR.caption := '';

CorreoR.caption := '';

ClaveR.caption := '';

RequerimientosClave.Font.color := clwhite;

ReqTamanio.font.color := clwhite;

ReqNum.font.Color := clwhite;

ReqMin.font.Color := clwhite;

ReqMayus.font.Color := clwhite;

ReqCaractEspeciales.font.Color := clwhite;

ReqNoEnie.font.Color := clwhite;

Req4Iguales.font.Color := clwhite;

Req4Iguales.font.color := clwhite;

Paises.Caption := 'Seleccione su pais';

Paginas.ActivePageIndex := 1;

End;

Procedure TProjecT.RequerimientosClaveClick(Sender: TObject);

Begin

If RequerimientosClave.Height = 24 Then

Begin

RequerimientosClave.caption := '';

RequerimientosClave.Height := 330;

End

Else

Begin

RequerimientosClave.caption := 'Mostrar Requerimientos para la clave ';

RequerimientosClave.Height := 24;

End;

End;

Procedure TProjecT.Timer1Timer(Sender: TObject);

Begin

Timer1.Enabled := false;

Paginas.PageIndex := 0;

End;

Procedure TProjecT.GenerarDiagramaDeGanttClick(Sender: TObject);

Begin

Paginas.ActivePageIndex := 4;

End;

Procedure TProjecT.EnviarDatosClick(Sender: TObject);

Begin

Verificar\_ArchivoJugadores;

RegistrodeJugadores;

End;

Procedure TProjecT.ExtrasClick(Sender: TObject);

Begin

Paginas.PageIndex := 3;

End;

Procedure TProjecT.iniciarsesion1Click(Sender: TObject);

Begin

IniciodeSesion (Nombre\_Usuario);

NombreJugadorLabel.Caption := 'JUGADOR: ' + Nombre\_Usuario;

Jugador.Caption := Nombre\_Usuario;

Iniciodeseion.Caption := 'Cerrar sesion';

MensajeErrorJugar.visible := false;

End;

Procedure TProjecT.IniciodeseionClick(Sender: TObject);

Begin

//TODO colocar revision de cerrar sesion

If (Iniciodeseion.Caption = 'Iniciar sesion') Then

Begin

Paginas.ActivePageIndex := 2;

Project.ErrorInicioSesion.font.color := clred;

ProjecT.ErrorInicioSesion.visible := false;

End

Else

Begin

Jugador.Caption := 'Usuario';

Iniciodeseion.Caption := 'Iniciar sesion';

Nombre\_Usuario := '';

End;

End;

Procedure TProjecT.JugarClick(Sender: TObject);

Begin

If (Jugador.Caption <> 'Jugador') Then

Paginas.ActivePageIndex := 7

Else

Begin

MensajeErrorJugar.Caption := 'Debe iniciar sesion para poder jugar!!!';

MensajeErrorJugar.visible := true;

End;

End;

Procedure TProjecT.MostrarclaveClick(Sender: TObject);

Begin

If (Mostrarclave.Checked = true) Then

Begin

ClaveIS.PasswordChar := #0;

End

Else

Begin

ClaveIS.PasswordChar := '\*';

End;

End;

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Sector de extras\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Generacion de un diagrama de Gantt\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Buqueda de los datos de un jugador en especifico\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

Procedure TProjecT.Buscar\_JugadorClick(Sender: TObject);

Begin

Paginas.ActivePageIndex := 5;

End;

Procedure TProjecT.VolverAtrasClick(Sender: TObject);

Begin

Paginas.ActivePageIndex := 3;

End;

Procedure TProjecT.UsuarioABuscarClick(Sender: TObject);

Var

Cont\_Lineas, Cont\_Partidas: integer;

PartidaAux: String[200];

UsuarioBuscado: ShortString;

ListaStringPartidas: TstringList;

Begin

//Se inicializa el cuadro de texto para ingresar el usuario, y la lista de partidas.

UsuarioABuscar.Font.Color := clblack;

ListaPartidas.Font.color := clblack;

For i:=ListaPartidas.Items.count-1 Downto 0 Do

ListaPartidas.Items.delete(i);

Cont\_Partidas := 0;

UsuarioBuscado := UsuarioABuscar.text;

Reset(Juegos);

While (Not (SeekEOF(Juegos))) Do

Begin

Readln(Juegos,Linea);

FileReading(Linea,Holders);

If Holders[2] = UsuarioBuscado Then

Cont\_Partidas := Cont\_Partidas+1;

End;

closefile(Juegos);

If Cont\_Partidas>0 Then

Begin

ListaStringPartidas := TstringList.create;

Cont\_Lineas := 1;

Reset (Juegos);

While (Not (SeekEOF(Juegos))) And (Cont\_Lineas<=cont\_Partidas) Do

Begin

Readln(Juegos,Linea);

FileReading(Linea,Holders);

If Holders[2] = UsuarioBuscado Then

Begin

//Se agrega formato al holder que contiene la fecha. Si contiene menos de 10 caracteres (XXXX/XX/XX), se agregan los necesarios para alcanzar el 10 y evitar que se corran las cantidades que vienen despues de este texto.

If length(Holders[3]) < 10 Then

For i:=1 To (10 - length(Holders[3])) Do

Holders[3] := Holders[3]+' ';

//Lo mismo que con el anterior holder, pero con el que contiene la hora, siendo ahora 8 caracteres (XX:XX:XX)

If length(Holders[4]) < 8 Then

For i:=1 To (8 - length(Holders[4])) Do

Holders[4] := Holders[4]+' ';

PartidaAux := Holders[1]+' '+Holders[3]+' '+Holders[4]+' '+Holders[5];

ListaStringPartidas.Add(PartidaAux);

Cont\_Lineas := Cont\_Lineas+1;

End;

End;

ListaPartidas.Items.assign(ListaStringPartidas);

ListaStringPartidas.free;

closefile(Juegos);

End

Else

Begin

UsuarioABuscar.Font.Color := clred;

ListaPartidas.Font.color := clred;

ListaPartidas.Items.add('Este Usuario no existe o no tiene partidas registradas');

End;

End;