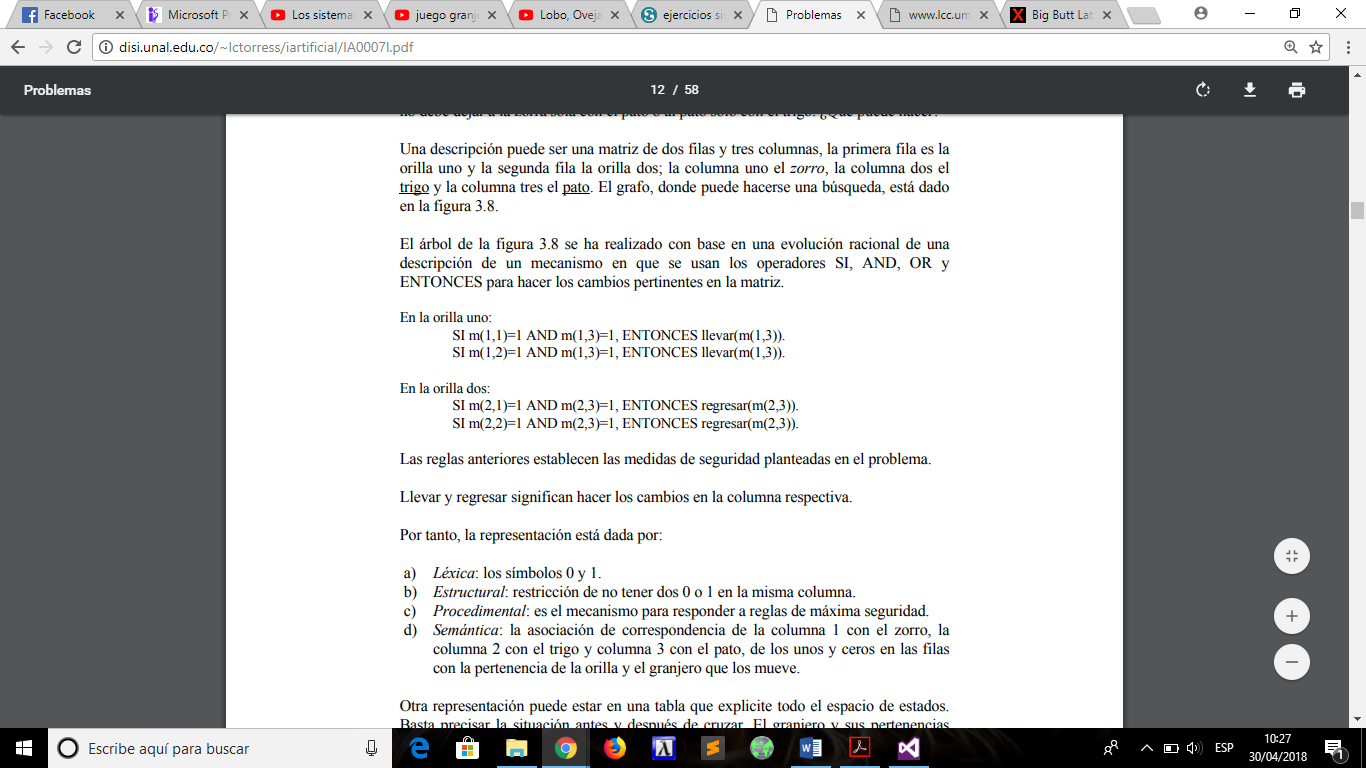
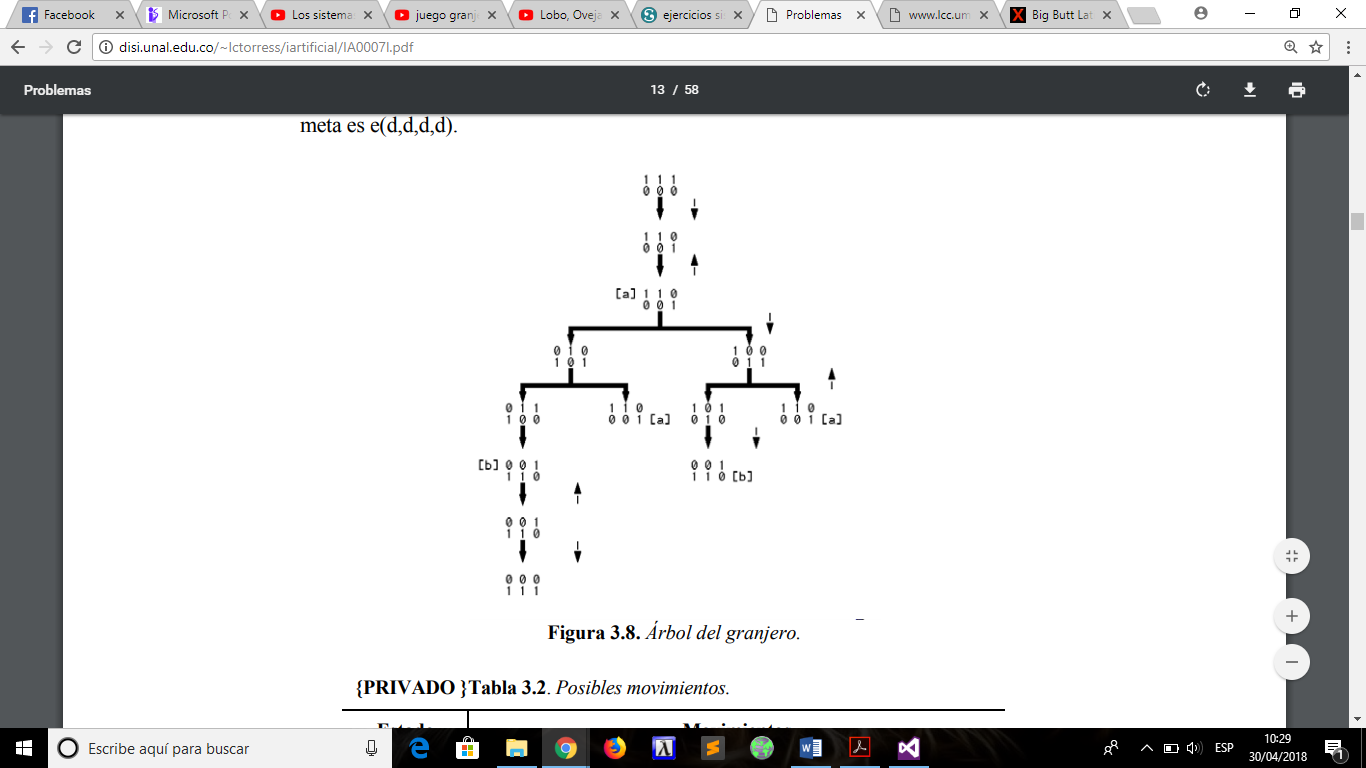
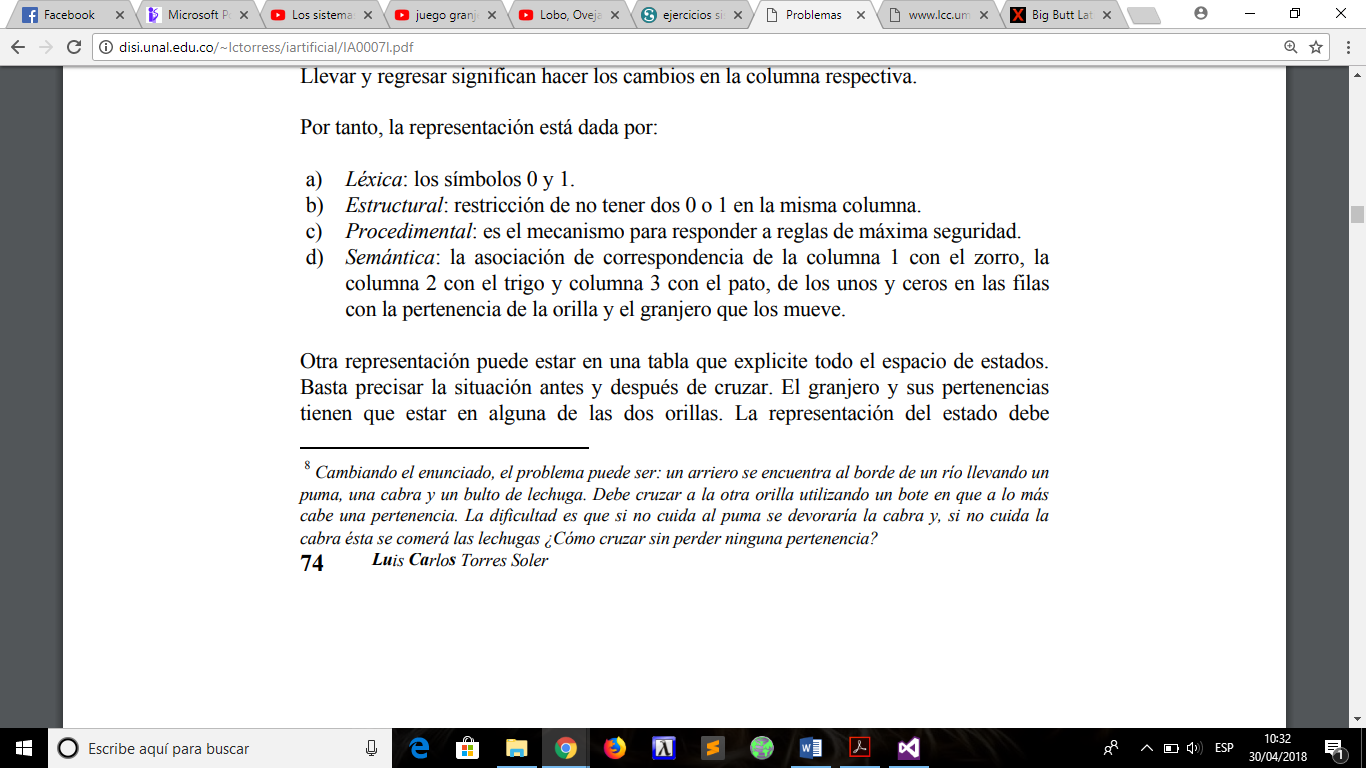
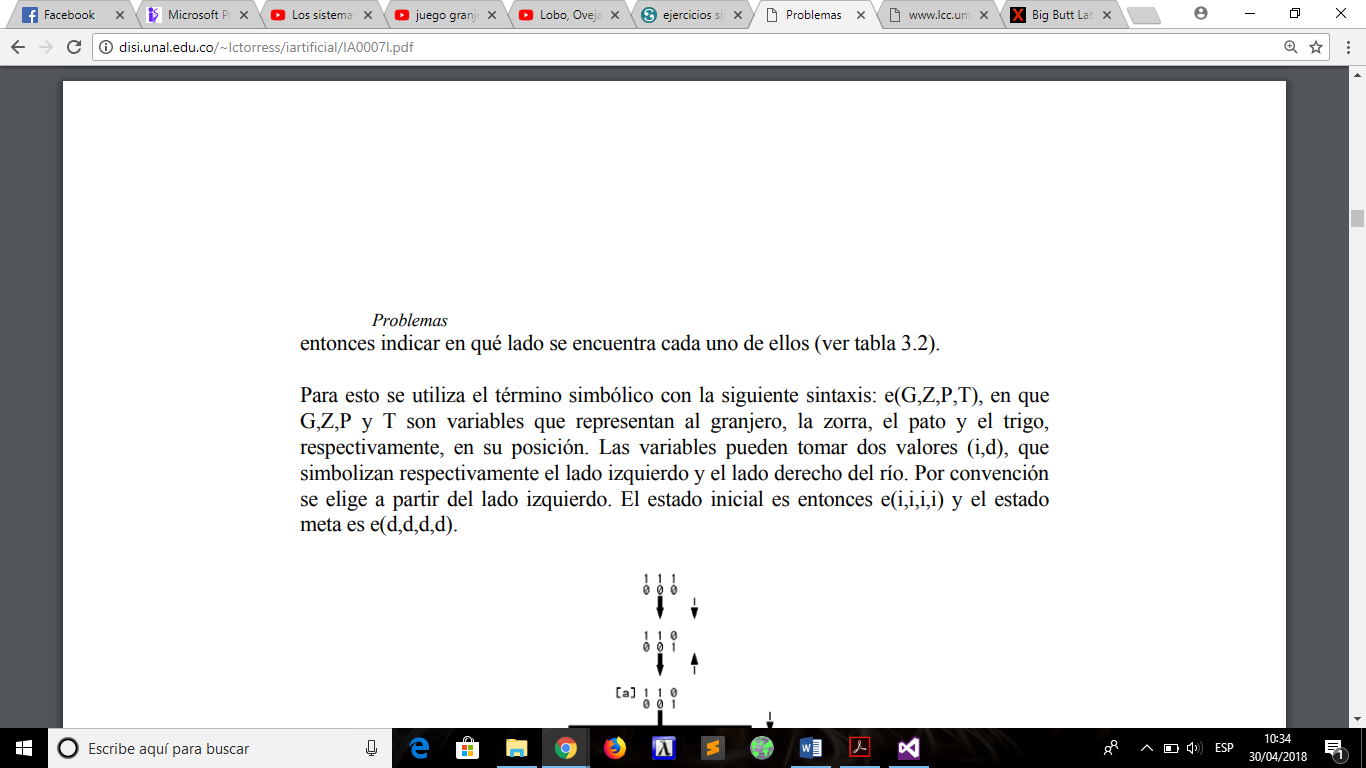
Ejercicio 2: El Granjero

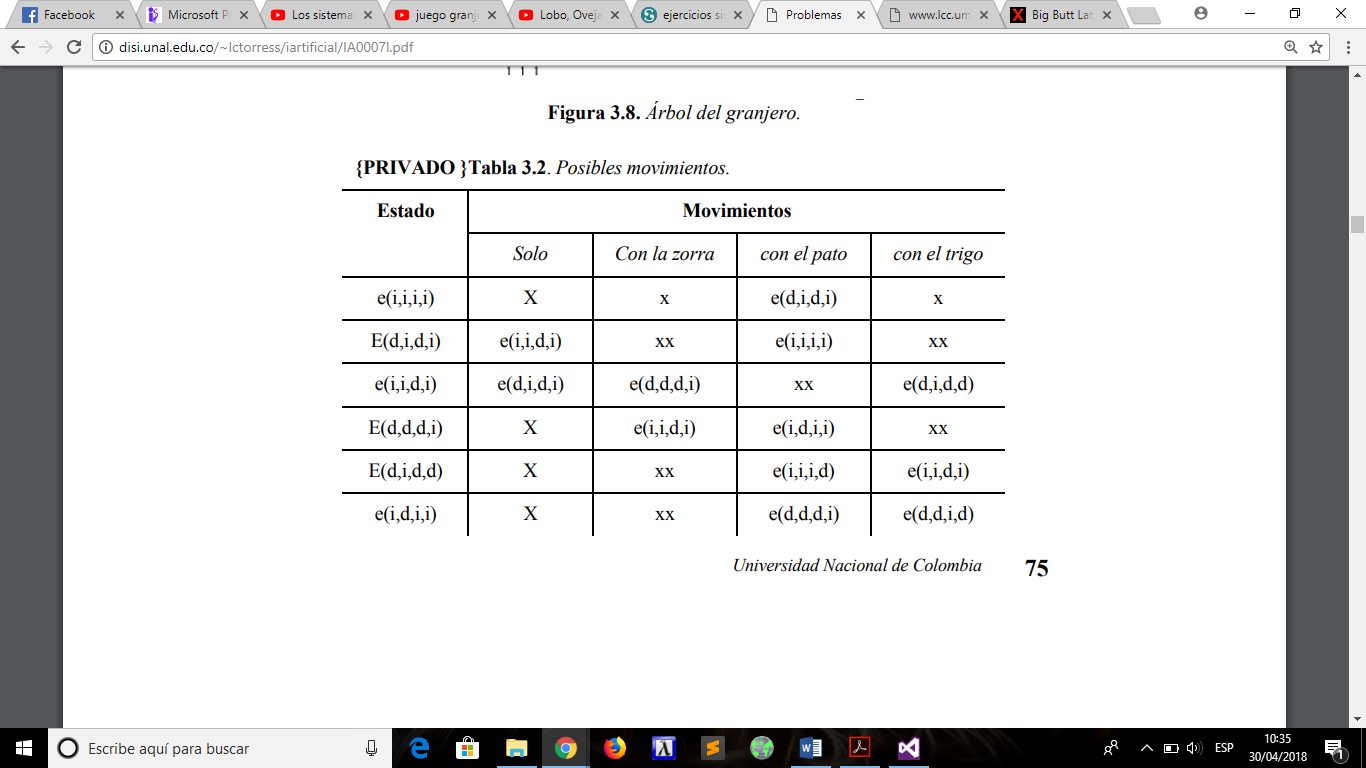
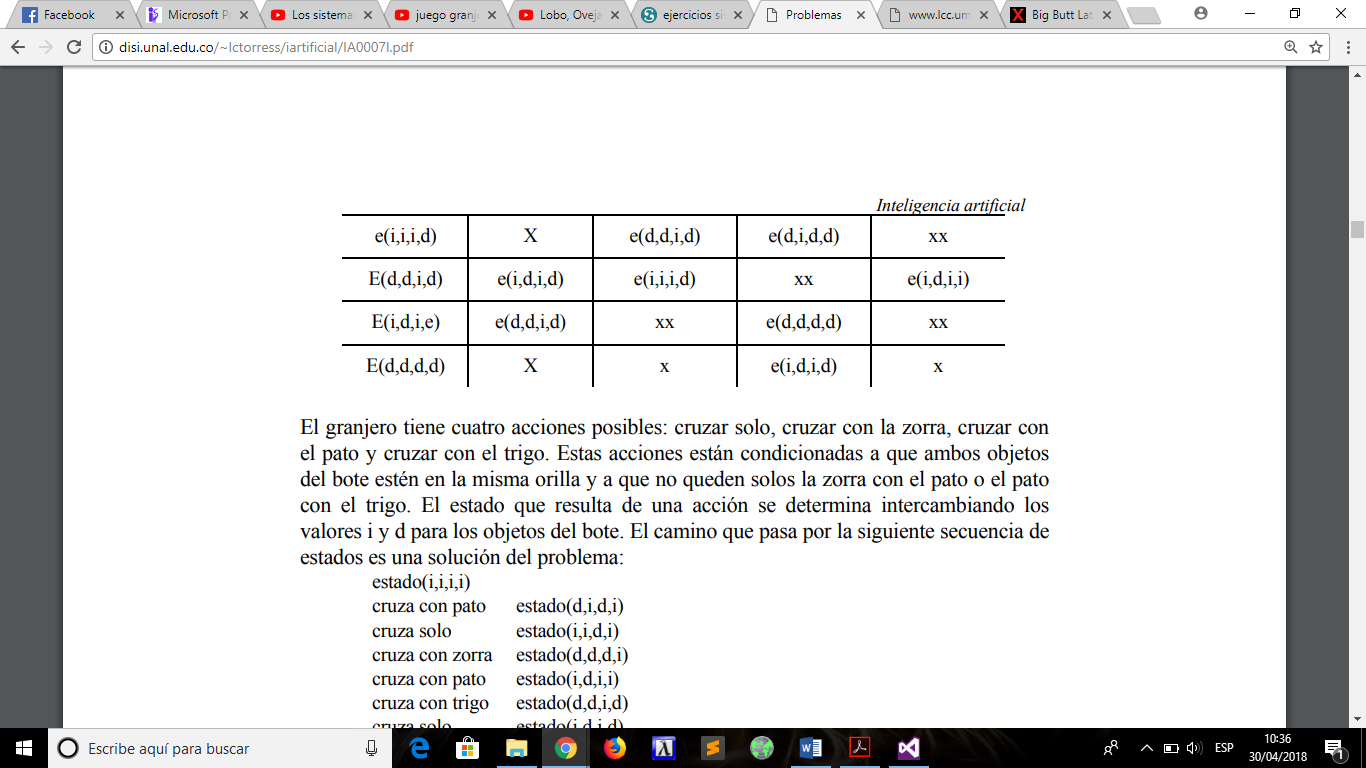
Diseñe un sistema de producción que resuelva el siguiente problema: “Un granjero quiere cruzar un río llevando consigo a una zorra, un ganso y un saco de trigo. Por desgracia su bote es tan pequeño que solo puede transportar a una de sus pertenencias en cada viaje. Además, la zorra se puede comer al ganso si están solos, asimismo el ganso se puede comer al trigo. De esta manera, el granjero no puede dejar a la zorra sola con el ganso o al ganso solo con el trigo.”

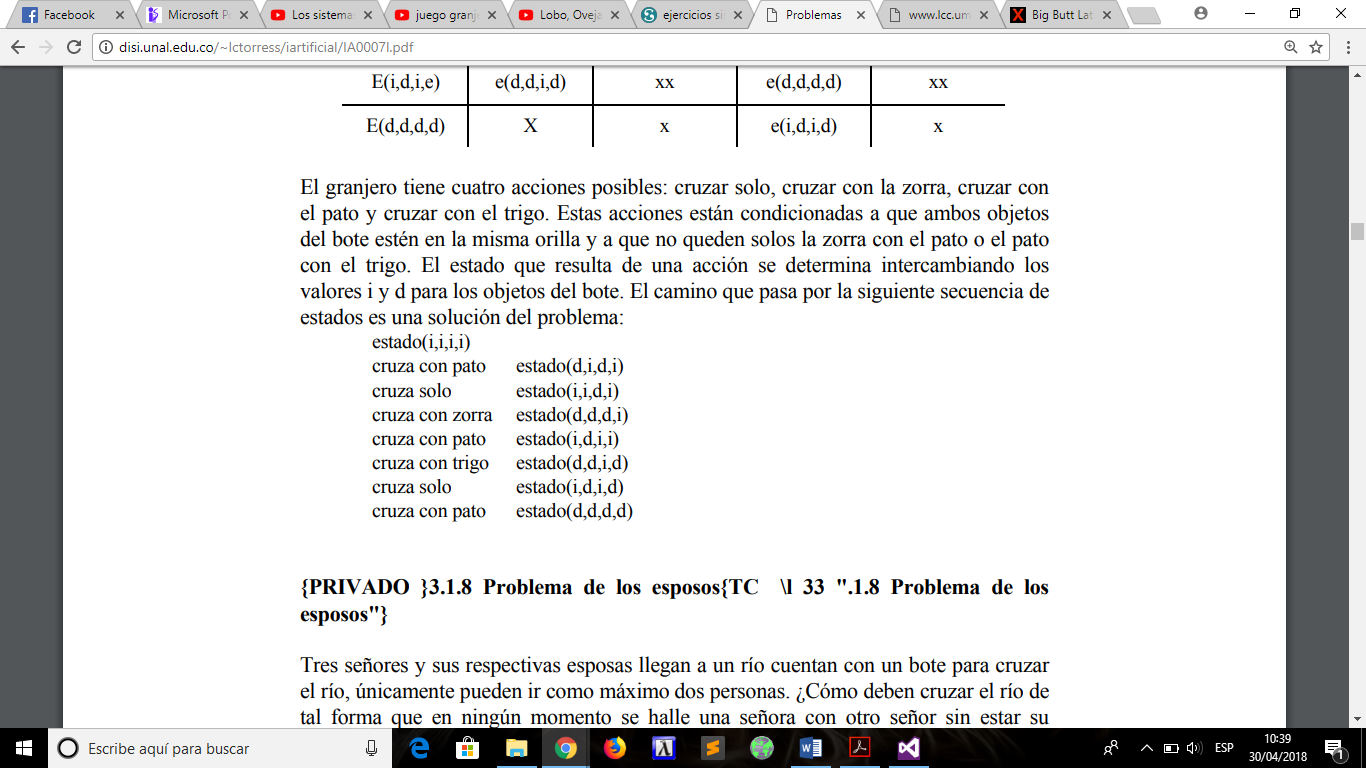
**Solución teórica:**









**Solución Practica:**

El siguiente sistema, está elaborado en C# con Visual Studio. Es representado gráficamente y contiene las condiciones que establece el problema marcadas como reglas. El ejercicio es interactivo y permite probar la imaginación del usuario.

Código fuente:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Juego\_Granjero

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

public void perdisteOrilla1()

{

if ((listOrilla1.Items.Count == 3 && listOrilla1.Items[1] == "Ganso" && listOrilla1.Items[2] == "Trigo" &&

listOrilla1.Items[0] == "Zorra") || (listOrilla1.Items.Count == 3 && listOrilla1.Items[2] == "Ganso" &&

listOrilla1.Items[0] == "Trigo" && listOrilla1.Items[1] == "Zorra") || (listOrilla1.Items.Count == 3 &&

listOrilla1.Items[0] == "Ganso" && listOrilla1.Items[1] == "Trigo" && listOrilla1.Items[2] == "Zorra"))

{

MessageBox.Show("Perdiste");

}

if ((listOrilla1.Items.Count == 2 && listOrilla1.Items[1] == "Ganso" && listOrilla1.Items[0] == "Trigo" ||

listOrilla1.Items.Count == 2 && listOrilla1.Items[0] == "Ganso" && listOrilla1.Items[1] == "Trigo"))

{

MessageBox.Show("Perdiste");

}

if ((listOrilla1.Items.Count == 2 && listOrilla1.Items[1] == "Ganso" && listOrilla1.Items[0] == "Zorra" ||

listOrilla1.Items.Count == 2 && listOrilla1.Items[0] == "Ganso" && listOrilla1.Items[1] == "Zorra"))

{

MessageBox.Show("Perdiste");

}

}

public void perdisteOrilla2()

{

if ((listOrilla2.Items.Count == 2 && listOrilla2.Items[1] == "Ganso" && listOrilla2.Items[0] == "Trigo" ||

listOrilla2.Items.Count == 2 && listOrilla2.Items[0] == "Ganso" && listOrilla2.Items[1] == "Trigo"))

{

MessageBox.Show("Perdiste");

}

if ((listOrilla2.Items.Count == 2 && listOrilla2.Items[1] == "Ganso" && listOrilla2.Items[0] == "Zorra" ||

listOrilla2.Items.Count == 2 && listOrilla2.Items[0] == "Ganso" && listOrilla2.Items[1] == "Zorra"))

{

MessageBox.Show("Perdiste");

}

}

public void perdisteBote()

{

if ((listBote.Items.Count == 2 && listBote.Items[0] == "Ganso" && listBote.Items[1] == "Trigo") ||

(listBote.Items.Count == 2 && listBote.Items[1] == "Ganso" && listBote.Items[0] == "Trigo") ||

(listBote.Items.Count == 2 && listBote.Items[0] == "Ganso" && listBote.Items[1] == "Zorra") ||

(listBote.Items.Count == 2 && listBote.Items[1] == "Ganso" && listBote.Items[0] == "Zorra"))

{

MessageBox.Show("Perdiste");

}

if (listBote.Items.Count > 2)

{

MessageBox.Show("No puedes subir a más de dos a la vez");

}

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

listBote.Items.Add(listOrilla1.SelectedItem.ToString());

listOrilla1.Items.Remove(listOrilla1.SelectedItem.ToString());

perdisteOrilla1();

perdisteBote();

}

catch { }

}

private void button4\_Click(object sender, EventArgs e)

{

try

{

if (listBote.Items[0].ToString() == "Granjero" || listBote.Items[1].ToString() == "Granjero")

{

for (int i = 0; i < listBote.Items.Count; i++)

{

listOrilla2.Items.Add(listBote.Items[i].ToString());

}

listBote.Items.Clear();

}

else

{

MessageBox.Show("No hay quien maneje el bote");

}

if (listOrilla2.Items.Count == 4)

{

MessageBox.Show("Felicidades, has ganado");

}

}

catch { }

}

private void button3\_Click(object sender, EventArgs e)

{

try

{

listOrilla1.Items.Add(listBote.SelectedItem.ToString());

listBote.Items.Remove(listBote.SelectedItem.ToString());

perdisteOrilla1();

}

catch { }

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

listBote.Items.Add(listOrilla2.SelectedItem.ToString());

listOrilla2.Items.Remove(listOrilla2.SelectedItem.ToString());

perdisteOrilla2();

perdisteBote();

}

catch { }

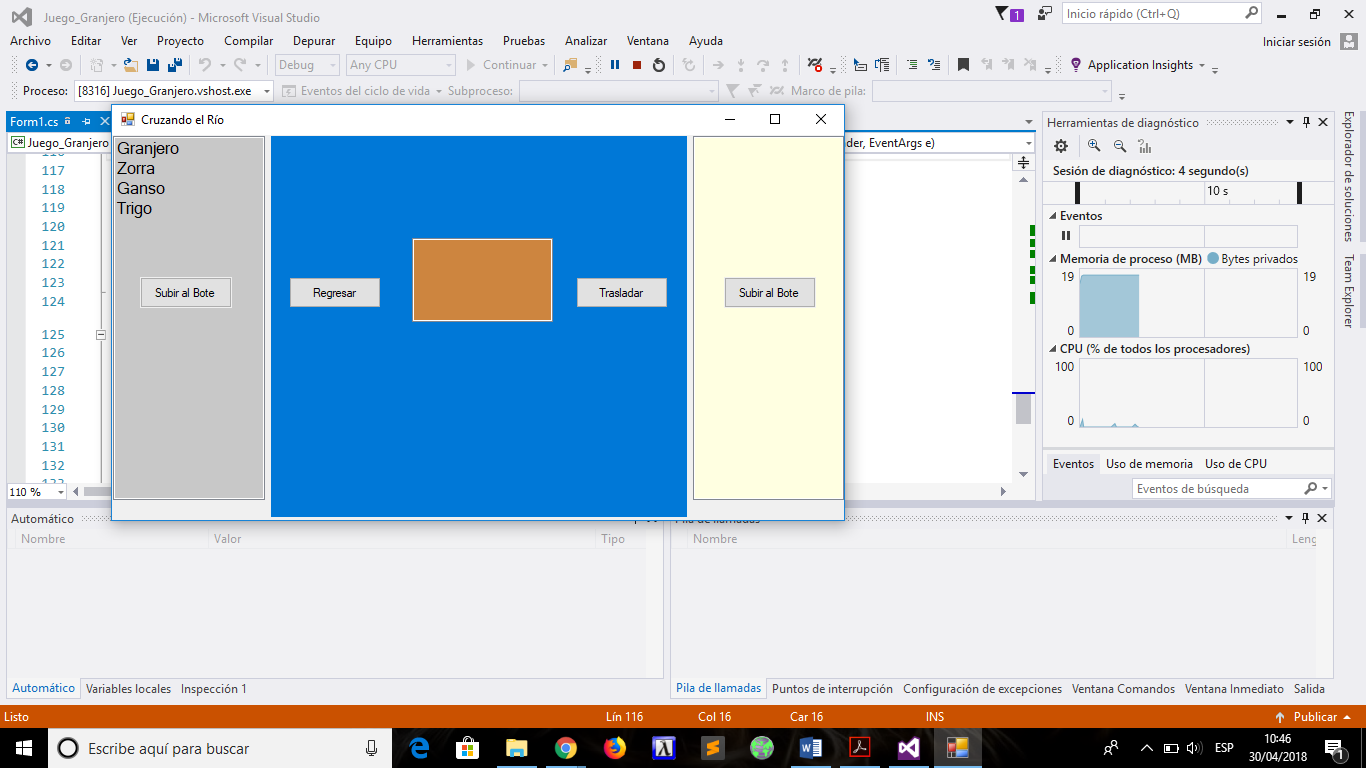
}

}

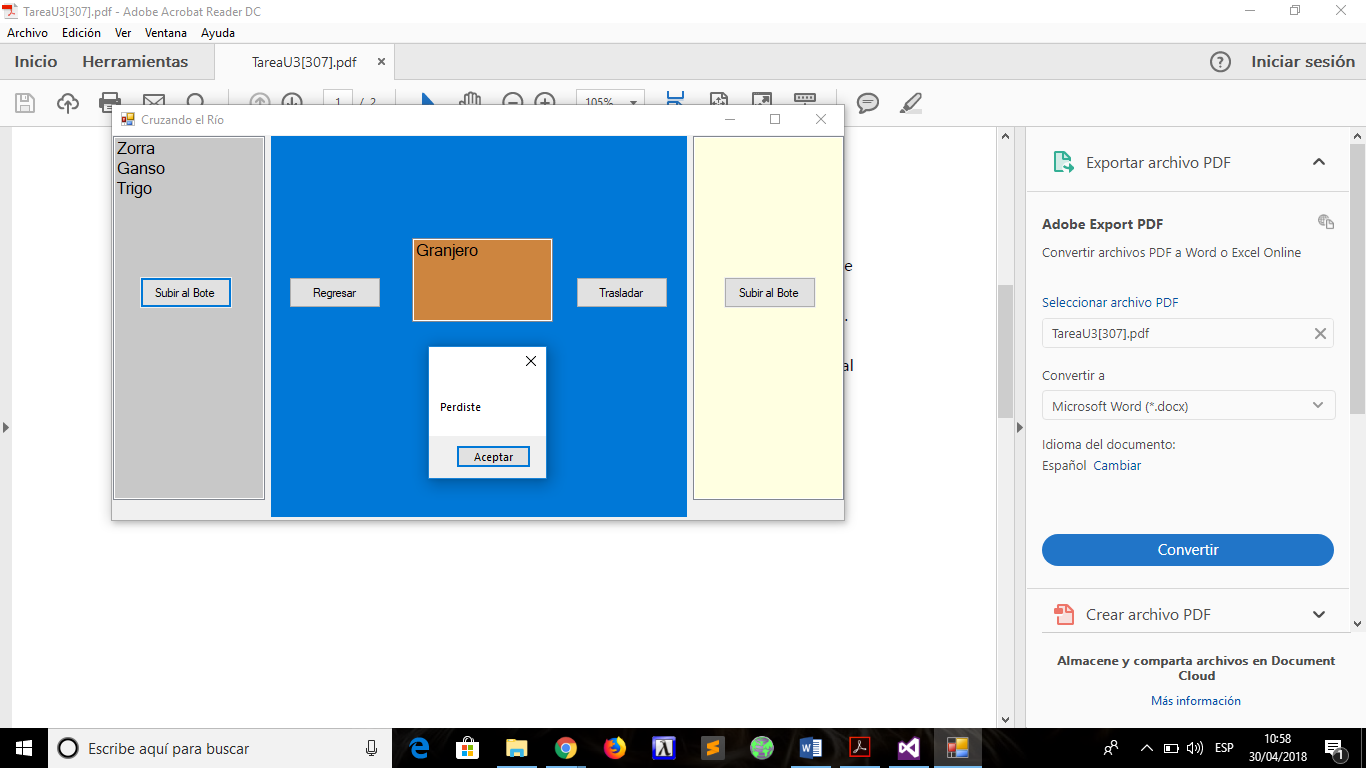
}

Resultados:

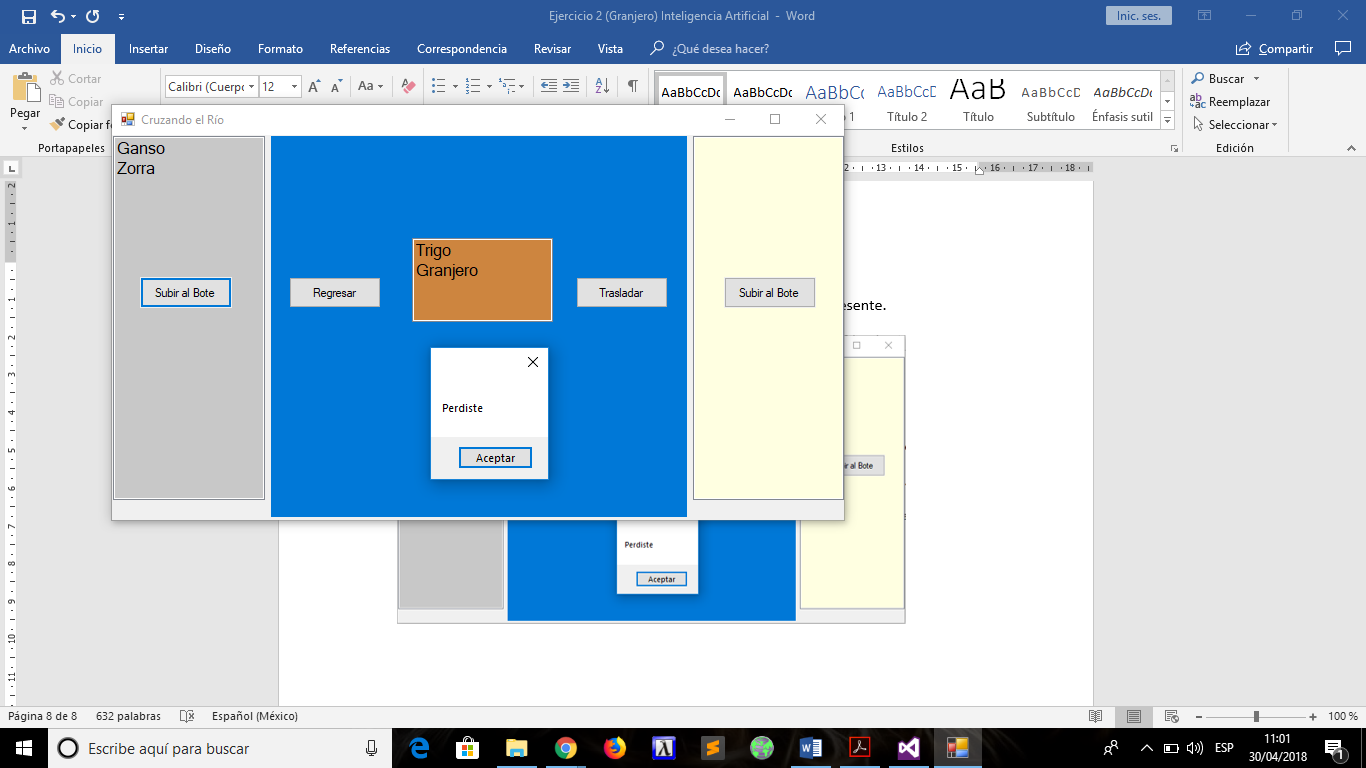
La interfaz que permitirá al usuario interactuar con el problema, será la presente. En la orilla 1, se ubican los 4 personajes y es marcada de color gris. El río es marcado con azul. El bote es representado por el rectángulo café ubicado dentro del río y la orilla 2 o destino es la marcada con el color beige.

El sistema contiene todas las reglas posibles en forma de validaciones, al infringir una regla (condición), se mostrará un mensaje “perdiste”, pero el juego permitirá corregir el error y encontrar una solución, esto hasta solucionar el problema.

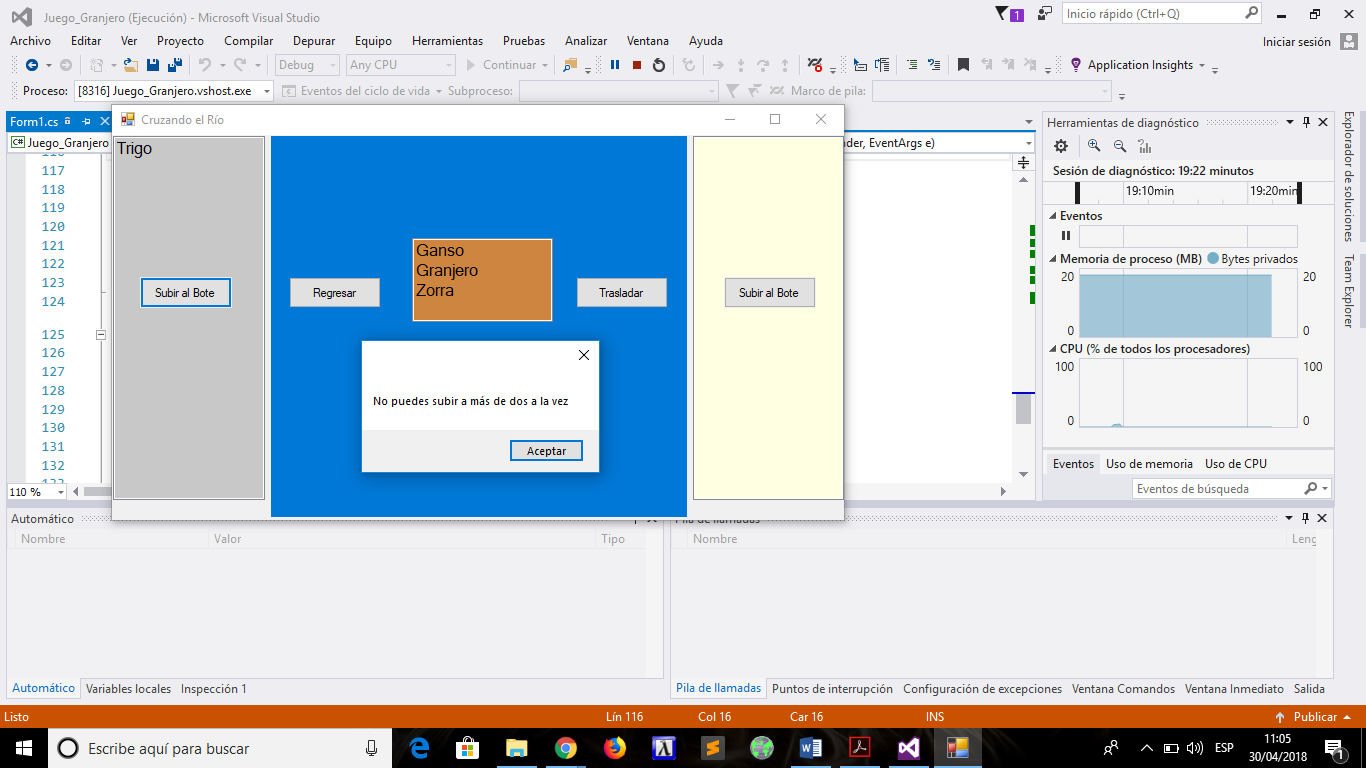
Mostrando las reglas establecidas (Condiciones)…

Si subimos primero al granjero al bote, pierdes y arroja el mensaje presente.

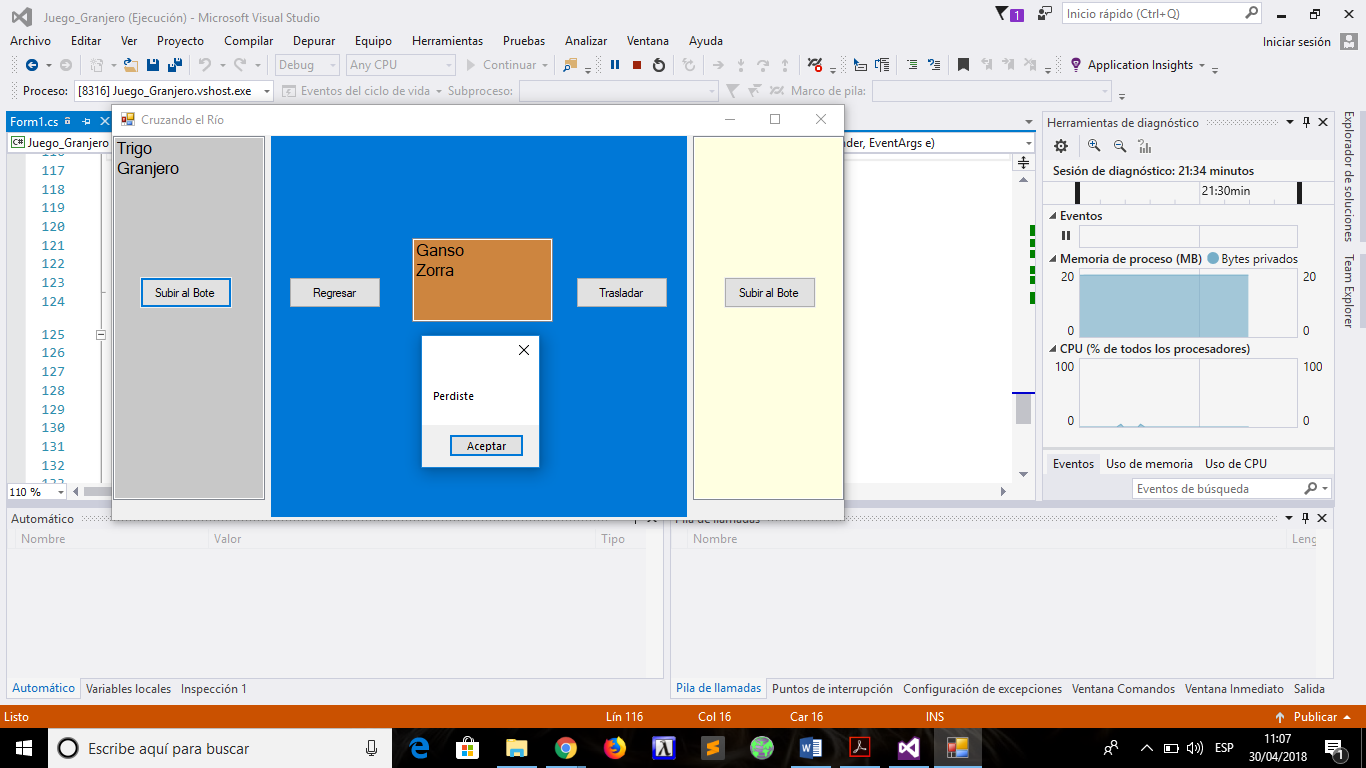
Si subes ya sea al trigo y al granjero al bote, al dejar en la orilla 1 al ganso y la zorra, pierdes. Lo mismo ocurriría si subes a la zorra y al granjero al bote y dejas en la orilla 1 al Ganso y al Trigo.



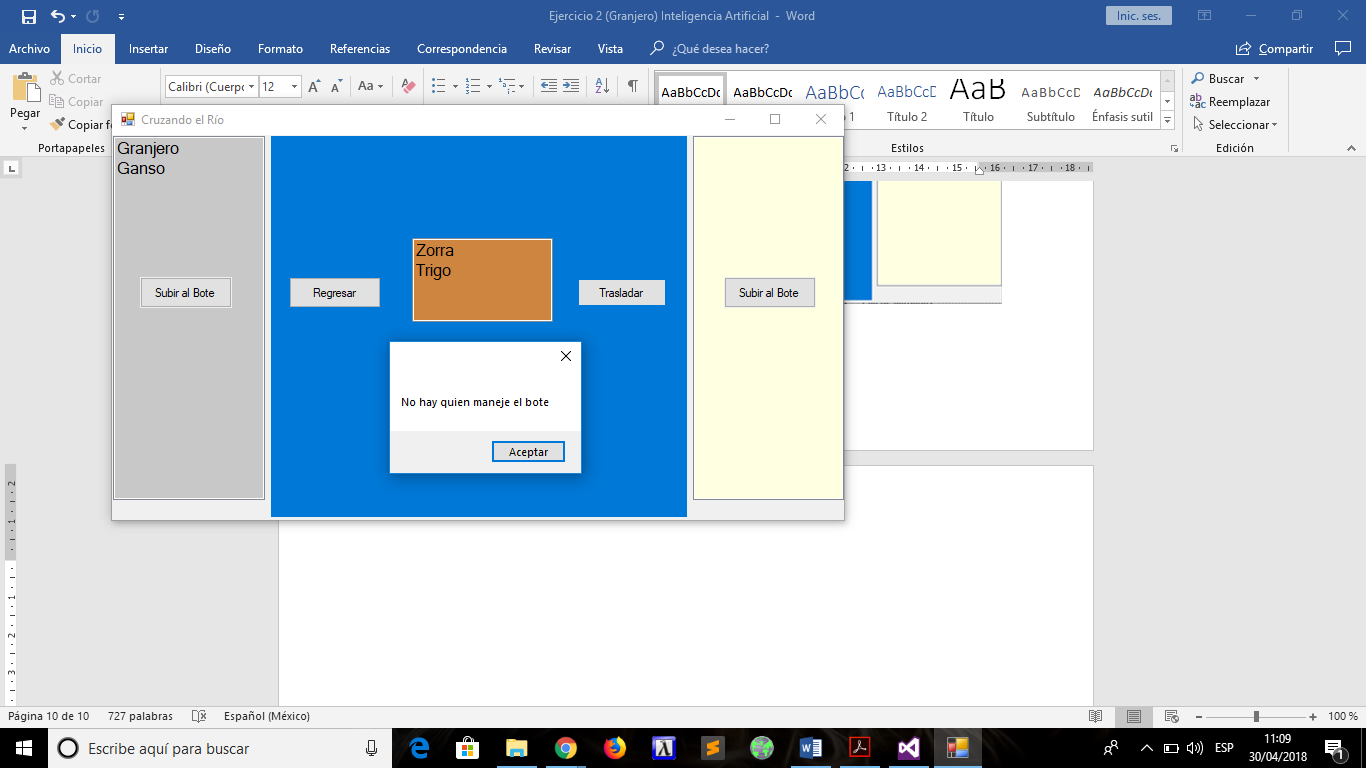
No puedes subir a mas de dos a la vez incluyendo al granjero. Al subir a mas de dos, muestra el presente mensaje.



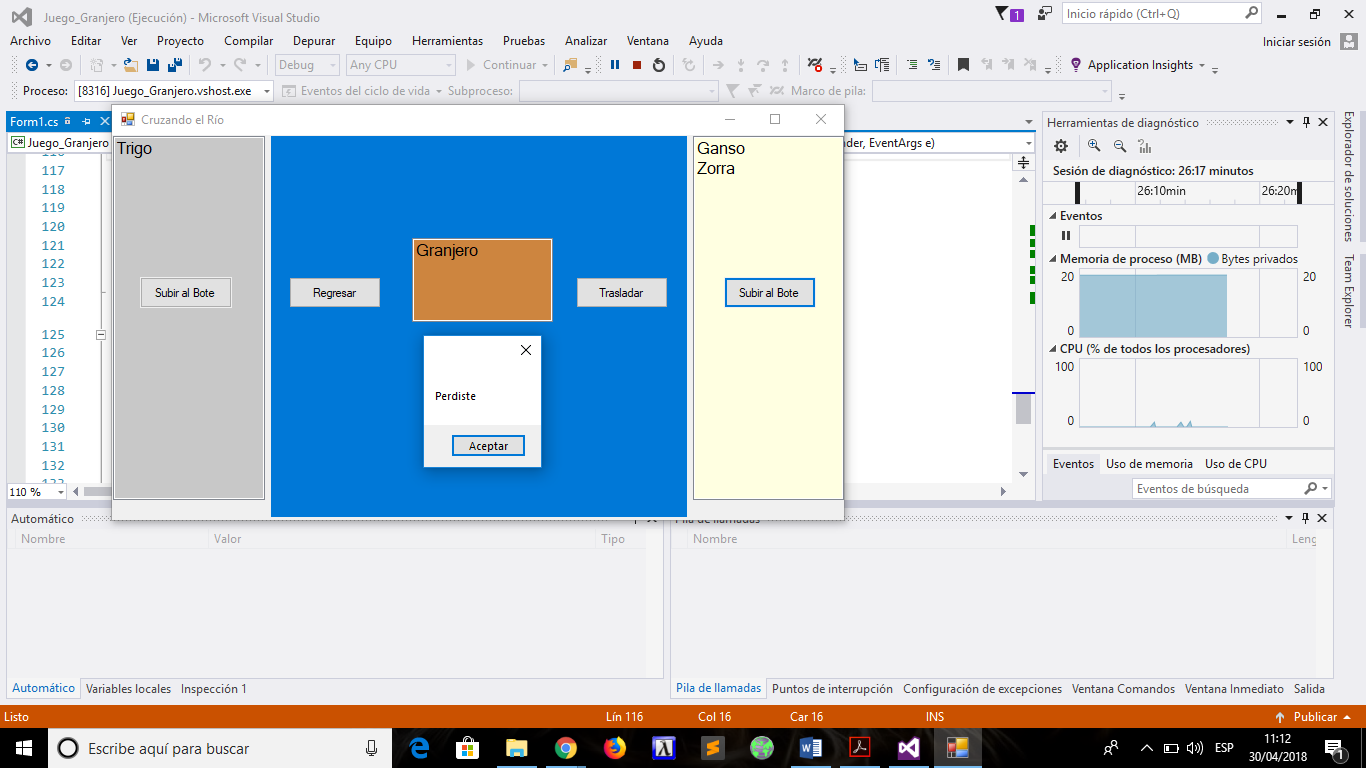
Si dejas en el bote al ganso y a la zorra, pierdes. Lo mismo pasaría si dejas en el bote al trigo y al ganso.



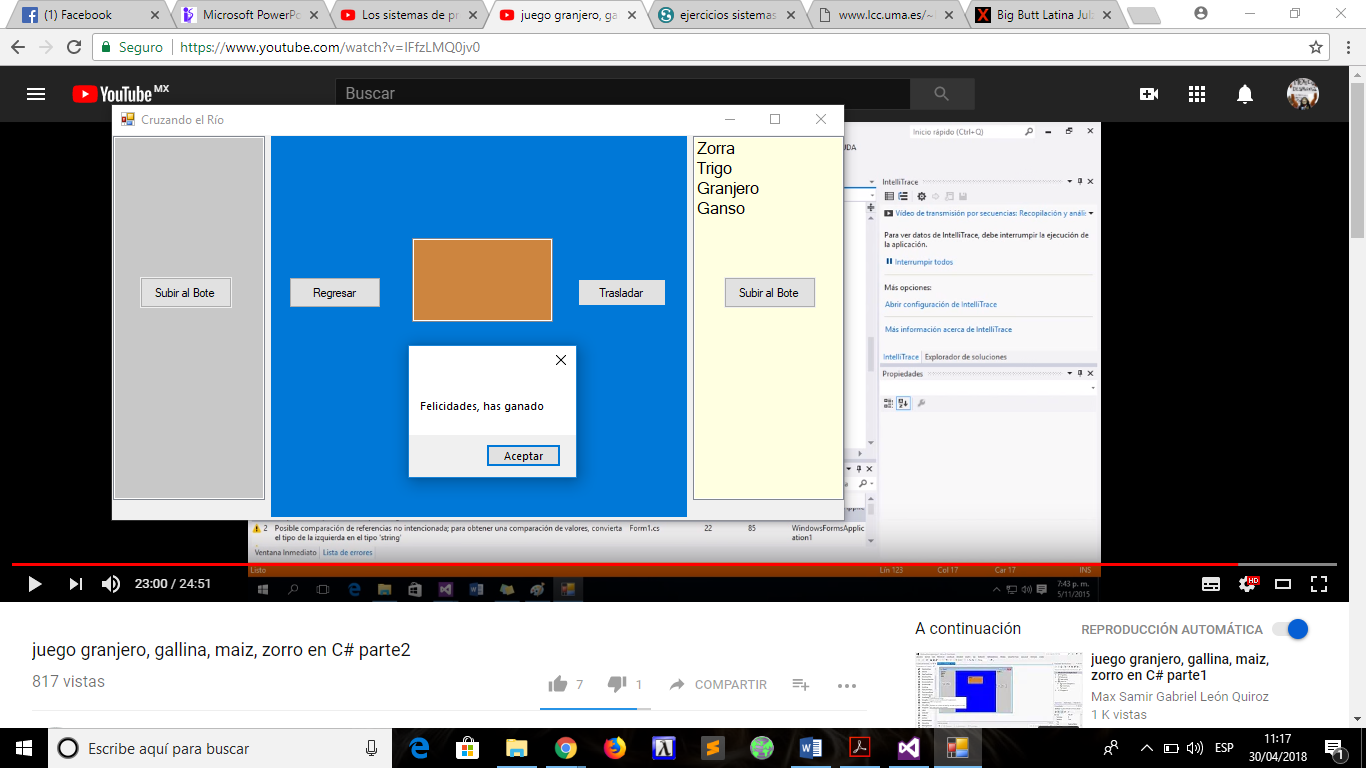
Si subes al bote a la Zorra y al Trigo y quieres cruzar del otro lado del río, arrojara el siguiente mensaje, pues el bote no tiene conductor.



Si en la orilla 2 dejas solos al ganso y a la zorra, pierdes. Lo mismo pasa si dejas solo al trigo y al ganso en la orilla 1 o la orilla 2.



Por último, al pasar en correcto orden a las pertenencias del granjero, se gana el juego y se soluciona el problema, arrojando el presente mensaje.



Conclusión de solución Practica:

De forma practica se llega al correcto orden que es;

1. Subir al ganso y al granjero al bote, trasladarlos a la orilla 2.
2. Dejar al ganso en orilla 2 y regresar al granjero a orilla 1.
3. Subir a la zorra al bote y al granjero, llevarlos a la orilla 2.
4. Subir al ganso y al granjero al bote, dejar a la zorra en la orilla 2.
5. Dejar al ganso en la orilla 1, subir al trigo y al granjero al bote.
6. Trasladar al granjero y al trigo a orilla 2.
7. Dejar al trigo y a la zorra en orilla 2 y subir al granjero al bote.
8. Regresar al granjero a orilla 1.
9. Subir al ganso al bote y al granjero.
10. Trasladar ambos a orilla 2.
11. Fin del juego “Felicidades”.

Segunda forma de Resolución al problema 2 con SLD-Draw:

%%% Ejercicio 4.b

% representaremos el estado del problema por una estructura estado/4:

%

% estado(Granjero,Zorra,Ganso,Trigo)

%

% donde cada componente representa el margen del rio en que se

% encuentra cada personaje (izq o der)

% al principio todos estan al mismo lado (izq)

estado\_inicial(glcc, estado(izq,izq,izq,izq)).

% al final deben estar todos al otro lado (der)

estado\_final(glcc, estado(der,der,der,der)).

% movimientos posibles

% cruza el granjero con la zorra.

mover(glcc,estado(X,X,Ganso,Trigo), estado(Y,Y,Ganso,Trigo)) :-

cruza(X,Y),

a\_salvo(estado(Y,Y,Ganso,Trigo)).

% cruza el granjero con el ganso.

mover(glcc,estado(X,Zorra,X,Trigo),estado(Y,Zorra,Y,Trigo)) :-

cruza(X,Y).

% no es necesario comprobar nada

% cruza el granjero con el Trigo.

mover(glcc,estado(X,Zorra,Ganso,X),estado(Y,Zorra,Ganso,Y)) :-

cruza(X,Y),

a\_salvo(estado(Y,Zorra,Ganso,Y)).

% cruza el granjero solo

mover(glcc, estado(X,Zorra,Ganso,Trigo), estado(Y,Zorra,Ganso,Trigo)) :-

cruza(X,Y),

a\_salvo(estado(Y, Zorra,Ganso,Trigo)).

cruza(izq,der).

cruza(der,izq).

% una configuracion es segura:

% 1) si el granjero y el ganso estan en el mismo lado del rio

a\_salvo(estado(X,\_,X,\_)).

% 2) si el granjero y el ganso estan en lados distintos,

% La zorra y el trigo deben estar con el granjero

a\_salvo(estado(X,X,Ganso,X)) :- X \= Ganso.

% Para resolver el problema, basta aplicar la plantilla de busqueda

% anterior con una profundidad maxima de 10:

%

% ?- resolver\_pmax(glcc,10,Sol)