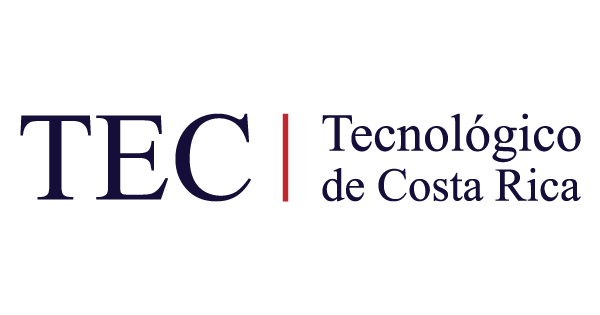
**Instituto Tecnológico de Costa Rica**



**Lenguajes de Programación**

**Proyecto programado #4**

**Desarrollado por:**

**Darío Vargas**

**Daniel Villatoro Cantarero**

**Daniel Calderón Díaz**

**II Semestre**

**2021**

**Tabla de contenidos**

[Manual de usuario](#_heading=h.3rdcrjn) **3**

[Compilación](#_heading=h.26in1rg) 3

[Ejecución](#_heading=h.lnxbz9) 3

[Uso](#_heading=h.35nkun2) 4

[Descripción del problema](#_heading=h.1ksv4uv) **4**

[Diseño programa](#_heading=h.44sinio) **5**

[Decisiones de diseño](#_heading=h.2jxsxqh) 5

[Algoritmos usados](#_heading=h.z337ya) 5

[Librerías Usadas](#_heading=h.3j2qqm3) **8**

[Resultados](#_heading=h.1y810tw) **9**

[Bitácora](#_heading=h.4i7ojhp) **10**

# Manual de usuario

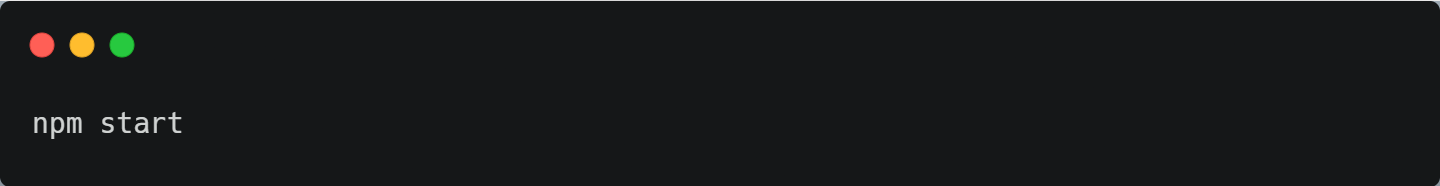
## Compilación

El presente proyecto fue programada la lógica de la solución en Node js y la interfaz gráfica en .Net Core en el sistema operativo Windows 10 de 64 bits. Para su compilación es necesario descargar [Node.js](https://nodejs.org/es/) en su versión 16.13.0 y de [Visual Studio](https://my.visualstudio.com/Downloads?q=visual%20studio%202017&wt.mc_id=o~msft~vscom~older-downloads) en su versión 2017. Además, es necesario de la instalación de localtunnel mediante el siguiente comando: **npm install -g localtunnel**.

## Ejecución

## Encender el servidor

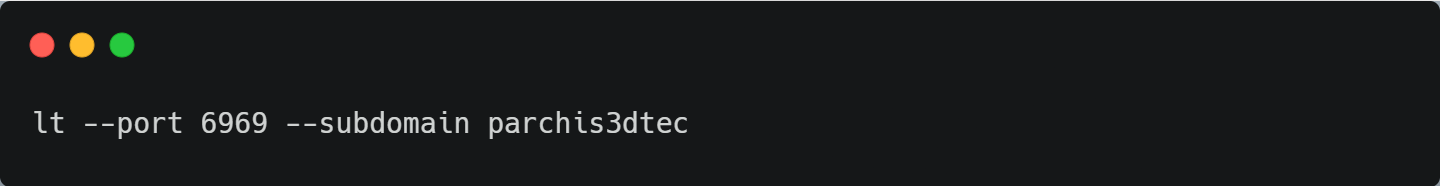
Para encender el servidor se hará mediante node js mediante el siguiente comando:



Este comando encenderá el servidor y lo conectará con la lógica del juego, comunicará al cliente y enviará y recibirá consultas para la funcionalidad del juego.

## Encender el tunel

Para la ejecución del proyecto primero se debe encender el tunel en una consola de comandos mediante el siguiente comando:



Esto nos permitirá abrir un puerto y obtener un dominio el cual servirá para conectar varios clientes al servidor de manera remota.

## Iniciar la interfaz gráfica

De momento, al ser un proyecto en desarrollo se deberá ejecutar la interfaz gráfica mediante Visual Studio. Para ello se deberá abrir el archivo llamado 3DParchis.sln y dar click en el ícono de ejecutar proyecto.

## Uso

# Descripción del problema

El problema que se resolverá en el presente proyecto es crear un servidor gaming en el cual se pueda jugar el juego de mesa parchís. El objetivo del proyecto es desarrollar una herramienta web de un juego en línea utilizando el paradigma orientado a objetos, por este motivo se separará la lógica y datos (Back-end) del juego de la solución web o interfaz (Front-end). Para el back-end se utilizará Node js y como front-end se utilizará .Net Core.

# Diseño programa

## Decisiones de diseño Diagrama del juego parchis

Como se muestra en la anterior ilustración, el juego estará compuesto de una clase principal llamada “Parchis”, la cual está dividida entre el “Jugador” y “Tablero”. El jugador tiene un dado, el nombre y fichas. Las fichas son tres, tiene cuatro colores y cuatro estados:

* **En casa:** Cuando la ficha está en la casa o posición inicial.
* **En tablero:** Cuando la ficha esta jugado.
* **En pasillo:** Cuando la ficha llega al pasillo.
* **Gane:** Cuando la ficha ha finalizado el recorrido.

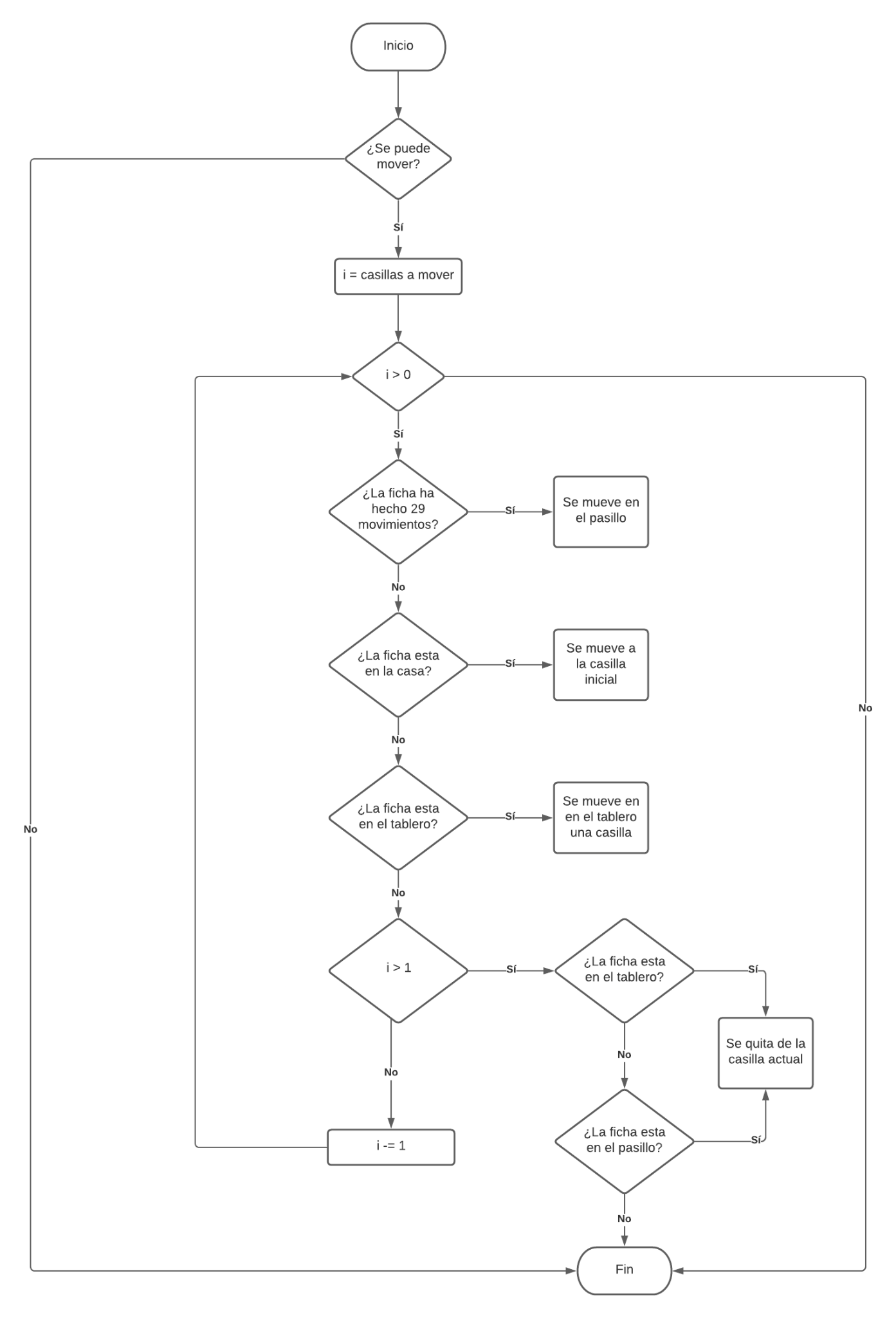
Los estados mencionados ayudan a que se ubique la ficha durante todo el recorrido en el pasillo. Por otro lado, el tablero está hecho de dos elementos más, el “Pasillo” que es donde las fichas recorren hasta llegar a la meta del juego y las “Casillas” que es donde las fichas pueden posicionarse. Las casillas tienen diferentes tipos:

* **Inicio:** Las casillas donde comienzan las fichas
* **Seguras:** Las casillas donde no se pueden comer fichas.
* **Normales:** Las casillas donde se pueden comer fichas.

Los tipos de casillas ayudarán a que las fichas “conozcan” a que ficha moverse cuando salen de la casa y comer a otra cuando esté otra ficha en una casilla normal y no pueda comerla cuando esté en una casilla segura.

## Algoritmos usados

A continuación se mostrará una figura donde se muestra el algoritmo de movimiento de la ficha:



Como se puede observar en la anterior figura explica como se realizó el algoritmo de movimiento de las fichas. Primero, se verifica si es válida el movimiento de la ficha en el tablero, luego en un ciclo for mueve la ficha la cantidad de veces indicada, cabe recalcar que al sacar una ficha de la casa se posicionará en una casilla en específico según su color. La ficha se quitará de la casilla actual siempre y cuando la cantidad de movimientos sobrantes sea mayor a uno.

# Librerías Usadas

| **Librería** | **Descripción** |
| --- | --- |
| clpfd | Restricción de los números del tablero. |
| java.awt | Componentes de interfaz de java. |
| java.util | Arrays y números aleatorios. |
| javax.swing | Componentes de intefaz de java |
| org.jpl7 | API para comunicarse con prolog. |

# Resultados

| **Objetivo** | **Resultado** | **Comentario** |
| --- | --- | --- |
| Cliente |  |  |
| Área de juego |  |  |
| Autenticación |  |  |
| Crear partida |  |  |
| Unirse al juego |  |  |
| Juego |  |  |
| Estadísticas |  |  |
| Ranking |  |  |

# Bitácora

<https://github.com/Dasperless/3D-Parchis>