**Instituto Tecnológico de Costa Rica**

**Escuela de Ingeniería de Computadores**

Algoritmos y Estructuras de Datos I Gr 3

Prof. [Leonardo](https://tecdigital.tec.ac.cr/dotlrn/community-member?user_id=397864) Araya Martínez

Proyecto No.1

Estudiantes:

Giancarlo Vega Marín

Julián Duarte Astúa

I Semestre 2023

**Descripción del problema**

Connect dots es un juego multiplayer de ***n*** jugadores. El juego consiste en una malla de puntos donde el jugador puede unir dichos puntos mediante líneas por turno. Si el jugador logra cerrar un cuadrado en su turno, puede seguir jugando hasta que agregue una línea que no cierre un cuadrado. El objetivo del juego es cerrar cuadrados para obtener puntos. El jugador que logre cerrar más cuadrados al final de la partida será el ganador.

![Aplicación

Descripción generada automáticamente con confianza media]()

Todos los cuadrados tienen el mismo valor. Cuando un jugador cierra un cuadrado, a este se le asignará algún identificador que permita saber cuál fue el jugador que lo cerró.

Dots es un juego multiplayer. Hay un servidor central que consiste en una aplicación en Java que escucha las conexiones entrantes por un Socket. Cada aplicación cliente se ejecuta en una computadora y se conecta por socket al servidor. Cuando el juego va a iniciar, el servidor recibe la petición del cliente y lo ingresa en una cola. En el momento que haya dos o más jugadores, empieza el juego. Pueden jugar varios jugadores a la vez.

El servidor lleva el control completo del juego. Los clientes únicamente grafican lo que el servidor les envía. Los clientes a su vez envían las acciones que realicen al servidor, el cual se encarga de mantener el estado del juego completo. Toda la comunicación entre cliente y servidor es en formato JSON.

![Diagrama

Descripción generada automáticamente]()

Adicionalmente, un cliente tendrá un control elaborado por los estudiantes que permitirá al jugador seleccionar dónde colocará las líneas.

**Diagrama de clases**

**Servidor**

Tabla

Descripción generada automáticamente con confianza media

**Cliente**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Descripción de las estructuras de datos desarrolladas**

**Corridas de ejemplo con capturas de pantalla y explicaciones en prosa**

**Código Arduino de Control de Botones y LEDs**

**Inclusión de Biblioteca:**



Incluye la biblioteca Softwares erial para permitir la comunicación serie en pines digitales arbitrarios.

**Declaración de objetos y variables:**

Texto, Carta

Descripción generada automáticamente

Declara un objeto ‘Serial1’ para la comunicación serie y asigna pines para LEDs y el LED adicional para indicar la presión de cualquier botón

**Configuración inicial en el método ‘setup’:**

Texto

Descripción generada automáticamente

Inicia la comunicación serie, configura pines como entradas o salidas, y activa la resistencia de pull-up para pines configurados como botones.

**Función ‘blinkLED’:**

Texto

Descripción generada automáticamente

Función para hacer parpadear un LED en el pin especifico

**Bucle Principal en el Método ‘loop’:**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Bucle principal que verifica el estado de cada botón. Si se presiona un botón, realiza acciones especificas y enciende LEDs correspondientes. Luego, apaga los LEDs al final del bucle

**Comunicación serie y Control de LEDs:**

Texto, Carta

Descripción generada automáticamente

Ejemplo para el botón “Arriba”. Envía un mensaje por ‘Serial1’, hace parpadear el LED correspondiente y enciende LEDs adicionales para indicar la acción.

**Apagado de LEDs:**

Texto

Descripción generada automáticamente

Apaga el LED de botón y el LED general al final del bucle.