UNIVERSIDAD DEL BÍO-BÍO

# Proyecto semestral:
# Batalla naval en lenguaje C

**Nombres:** Alain Moraga Vargas

**Asignatura:** Sistemas operativos

**Fecha entrega:** 25/03/2020

**Profesor:** Carlos Faúndez

## Descripción del proyecto:

El juego Battleship consiste en que dos jugadores deben tratar de hundir barcos enemigos, para ello el juego se representa como una matriz donde sus filas se representan por letras y columnas con números. Sí un jugador dice una coordenada y en la matriz enemiga hay un barco, entonces este se hunde (HIT!), caso contrario es una falla (MISS!). Cada matriz debe ser de 5x5 y se ubican 5 barcos de dimensión 1 (solo ocupará una celda). El juego termina cuando los 5 barcos de un jugador están hundidos.

## Propuesta de solución:

Para resolver este proyecto se propone crear un servidor y dos clientes mediante tuberías con nombre conocidas como FIFO.

El FIFO receptor actuará como servidor, preparará las matrices de ambos clientes inicializándolas y agregando barcos a ellas en forma aleatoria. También manejará la lógica del juego y notificará a los clientes si un ataque fue exitoso o si fallo, así como también los barcos que le faltan por derribar a un determinado cliente.

El FIFO emisor actuara como cliente, en este caso existen dos FIFOS emisores representando al cliente 1 y 2. Los clientes recibirán sus matrices desde el servidor y las podrán visualizar.

Respecto a la sincronización de los ataques se utilizaron semáforos, para que un cliente no pueda atacar mientras otro aún está en su turno.

UNIVERSIDAD DEL BÍO-BÍO

## Descripción de funciones

| Función | Parámetros | Descripción |
|---|---|---|
| void inicializar_mapa(int mapa[5][5]); | Recibe las matrices del cliente 1 y 2. | Permite la inicialización de los mapas en 0 y evitar que las matrices contengan basura, además de representar los mapas no tienen barcos. |
| void generar_barcos(int mapa[5][5]); | Recibe las matrices del cliente 1 y 2. | Permite generar barcos en posiciones aleatorias dentro del mapa. Dichos barcos se representarán con un 1 en la matriz. |
| void mostrar_mapa(int mapa[5][5]); | Recibe las matrices del cliente 1 y 2. | Permite visualizar las matrices de los clientes. |
| void procesar_coordenadas (char *cadena, int mapa[5][5]) | Recibe las matrices de los clientes y un buffer vacío. | Permite guardar las coordenadas de las matrices en una cadena para su posterior envío a los clientes. |
| void removerCaracteres (char *cadena); | Recibe una cadena de caracteres de un cliente. | Permite la eliminación de caracteres especiales para poder trabajar solo con números asociados a las coordenadas de ataque. |
| int retornar_x(char *cadena);<br>int retornar_y(char *cadena); | Recibe una cadena de caracteres de la cliente previamente filtrada. (sin caracteres) | Permite obtener la coordenada x,y para realizar un ataque sobre una matriz. |
| void error(char* errorInfo);<br>void doSignal(int semid, int numSem);<br>void doWait(int semid, int numSem);<br>void initSem(int semid, int numSem, int valor); | Reciben parámetros para el funcionamiento de semáforos. | Permiten el correcto funcionamiento de los turnos de los clientes mediante la sincronización que proveen. |