INSTITUTO TECNOLÓGICO DE ESTUDIOS SUPERIORES DE OCCIDENTE

**Proyecto de programación estructurada**

**“Minijuegos”**

|  |  |
| --- | --- |
| Materia: | Programación Estructurada |
| Participantes: | Luis David Bucio Macias.  Pablo Gerardo Zarate Félix |
| Profesor: | Francisco Cervantes |
| Fecha: | 09/12/19 |

Documento del Proyecto Final

Documento de definición

# Integrantes

LUIS DAVID BUCIO MACÍAS

PABLO GERARDO ZARATE FELIX.

# Resumen del proyecto

Para este proyecto buscamos realizar un programa donde haya dos juegos, uno de batalla naval y otro de buscaminas, en el cual buscaremos incorporar nuestros conocimientos del curso, así como cosas que hayamos investigado por nuestra cuenta para mejorar el juego.

Esperamos que realizar este proyecto reafirme nuestro conocimiento y nos ponga en contacto con otro tipo de herramientas que no alcanzaríamos a ver durante el curso. Como equipo tenemos los conocimientos y la organización necesaria para sacar adelante este proyecto.

# Contexto y terminologías del proyecto

1. Objetivo del juego de Batalla Naval:

Hundir los 9 barcos de tu oponente antes que él acabe con los tuyos.

Cada jugador tiene 2 tableros compuesto por 10 filas y 10 columnas.

Hundir todos los barcos del enemigo antes de que él lo haga.

Cada jugador tiene 9 barcos que ocuparan determinadas casillas en el tablero.

• 1 portaaviones: ocupa 4 casillas

• 3 submarinos/ Acorazados: ocupan 3 casillas. Son aleatorios, es decir, puedes tener 3 submarinos y 0 acorazados, 2 submarinos y 1 acorazado...etc.

• 3 destructores: ocupan 2 casillas.

• 2 fragatas: ocupan 1 casilla.

Terminología y movimientos:

• Agua: cuando disparas sobre una casilla donde no está colocado ningún barco enemigo, disparas al agua.

• Tocado: Cuando disparas en una casilla en la que está ubicado un barco enemigo que ocupa 2 o más casillas y destruyes sólo una parte del barco, le has tocado

• Hundido: Cuando disparas correctamente a todas las casillas de un barco

Reglas. -

• Número de jugadores: 2

• Una vez posicionas tus barcos y comienzas la partida, no podrás volver a cambiarlos de posición.

• Podrás disparar en cualquier casilla del tablero, salvo en las que ya has disparado.

• No puedes deshacer disparos ni propios ni de tus oponentes.

• Es un juego por turnos: haces tu disparo, si es "agua" el turno pasa a tu oponente; si "tocas" y/o hundes un barco enemigo, vuelves a disparar.

• La partida acaba cuando un jugador ha hundido la flota completa del enemigo.

2.- Objetivos juego de Buscaminas

El objetivo del juego es despejar un campo de minas sin detonar ninguna.

Reglas. -

* El juego consiste en despejar todas las casillas de una pantalla que no oculten una mina.
* Algunas casillas tienen un número, el cual indica la cantidad de minas que hay en las casillas circundantes. Así, si una casilla tiene el número 3, significa que de las ocho casillas que hay alrededor (si no es en una esquina o borde) hay 3 con minas y 5 sin minas. Si se descubre una casilla sin número indica que ninguna de las casillas vecinas tiene mina y éstas se descubren automáticamente.
* Si se descubre una casilla con una mina se pierde la partida.
* Se puede poner una marca en las casillas que el jugador piensa que hay minas para ayudar a descubrir las que están cerca.

Referencias:

https://es.wikipedia.org/wiki/Buscaminas

https://www.retos.com/juego/reglas?id=6

A nivel de código tendrá muchas similitudes con juegos que utilizan coordenadas par localización de objetivos o trampas por ejemplo como Memorama, ajedrez, etc.

# Alcance del Proyecto

## Lo que sí se incluye

Batalla Naval:

* Modo prueba del multijugador.
* Juego funcional de batalla naval.
* *Mapa en base a coordenadas.*
* *3 tamaños de mapa*
* *3 dificultades*
* *Contador de intentos*
* *Jugar contra la maquina*
* *Jugar contra otro jugador*
* *Barcos de 1x1*
* *ASCII ART*

*Buscaminas*

* *Juego funcional de Buscaminas*
* *Mapa en base a coordenadas*
* *3 tamaños de mapa*
* *3 dificultades*
* *Ganas al revelar todas las casillas que no tengan bomba*
* *Pierdes al revelar una casilla con bomba*
* *ASCII ART*

## Lo que no se incluye

Batalla Naval:

* Longitud de barcos (varias casillas con un solo barco)
* Que la maquina te ataque
* En multiplayer no decides que ip, solo se puede en una máquina
* No es automático el juego en el aspecto de quien recibe y envía

Buscaminas:

* La opción de poner una flecha como en el juego original donde creas que haya una mina
* Números en las casillas cercanas a una bomba como el juego original

## Riesgos posibles y medidas de prevención

Creemos que lo más complicado será desarrollar el sistema de Servidor Cliente, ya que es un tema bastante avanzado y que no vimos en el curso por lo que tendremos que investigar todo por nuestra cuenta.

También creemos que se nos puede complicar el sistema de posicionamiento de los barcos con base en coordenadas.

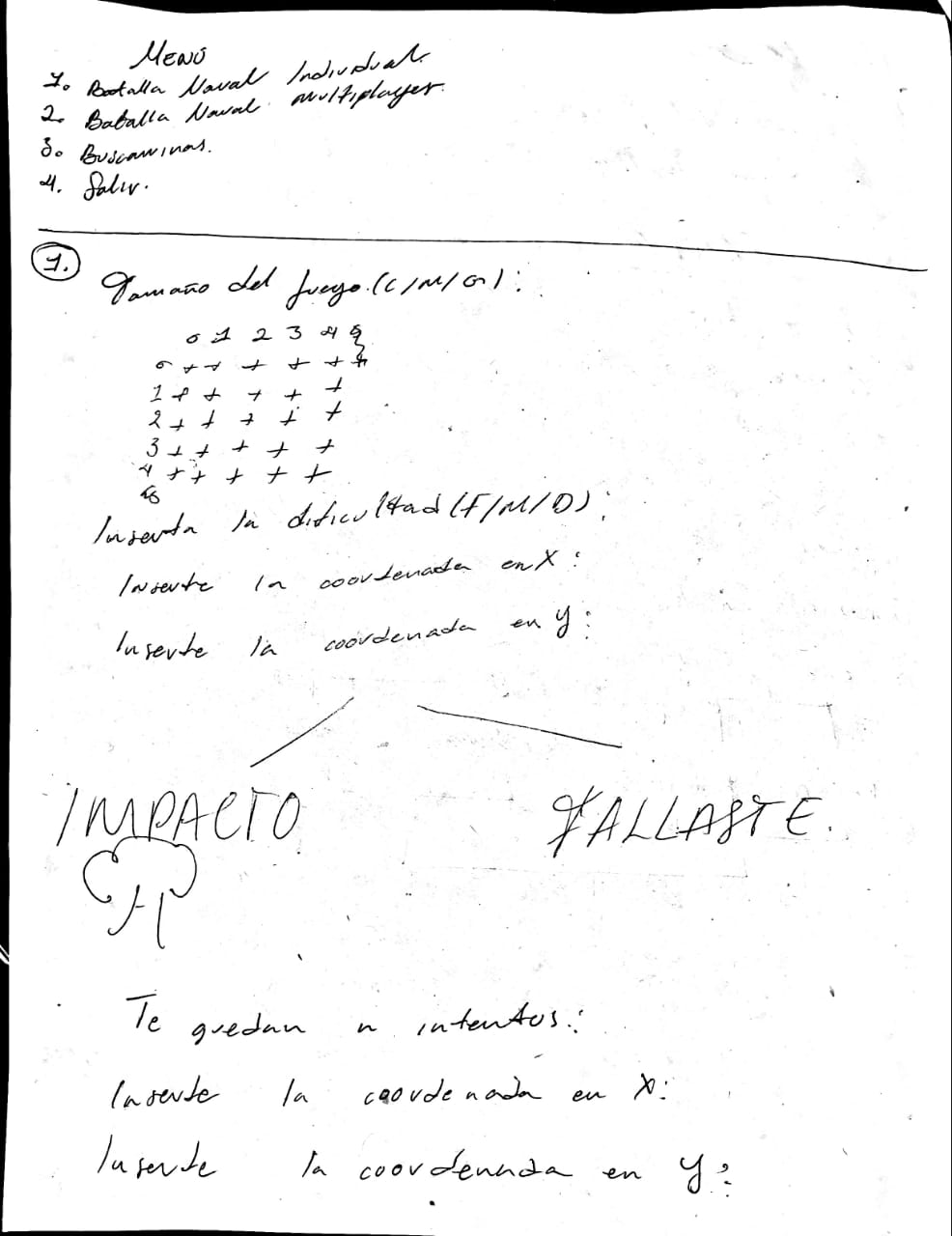
Las prevenciones que tomaremos será investigar bien los temas, mediante páginas de internet, videos, y asesorías con el maestro, así como ordenarnos para que luego no nos falte el tiempo.

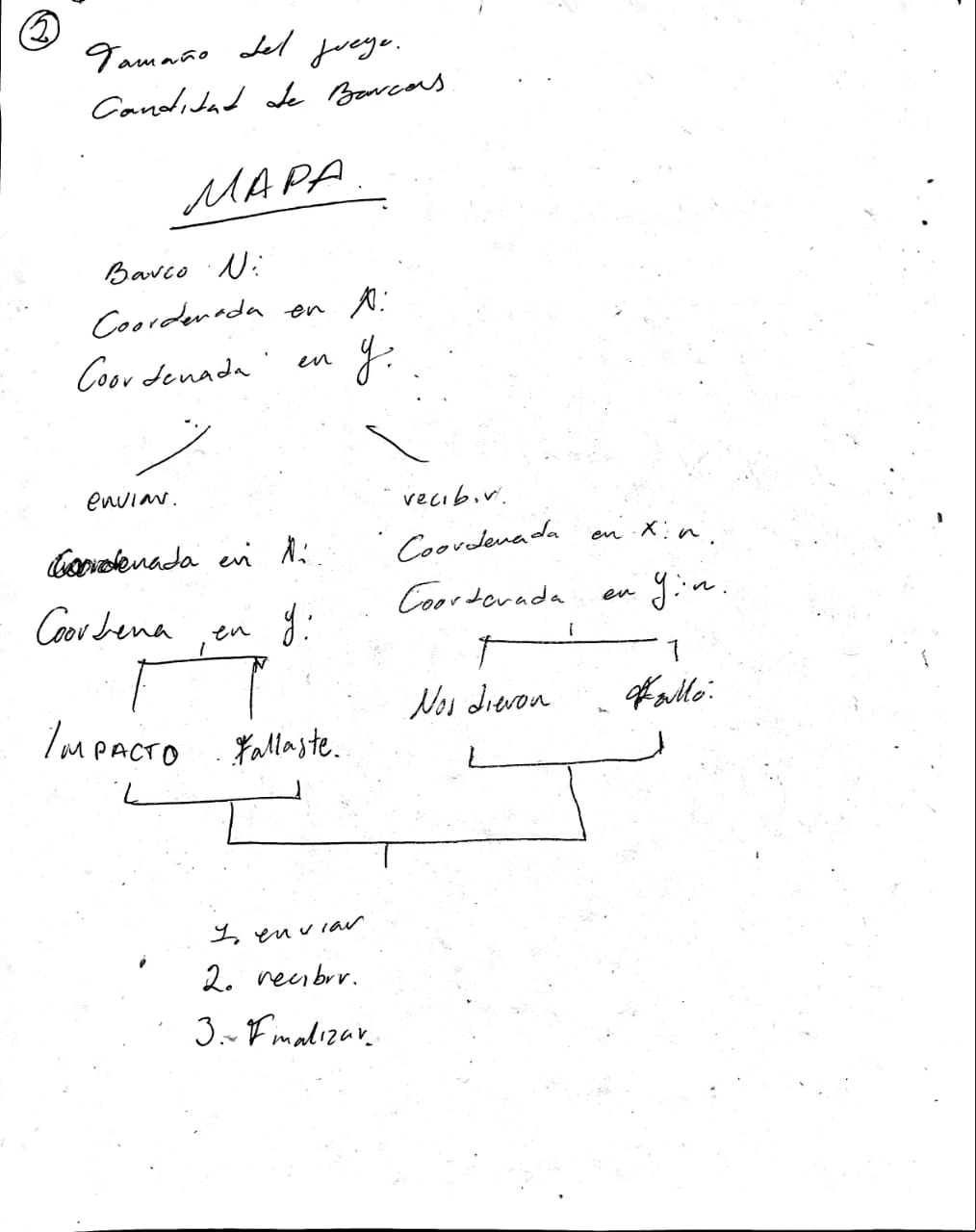
**6.- Hipótesis de solución y borradores de interfaz**

**Análisis del problema:** Al finalizar nuestro programa esperamos haber creado 2 juegos, uno de batalla naval y otro de buscaminas, en el cual el usuario pueda ingresar la dificultad del juego, la cual afectará la cantidad de barcos o minas en el espacio y además el usuario podrá ingresar el tamaño del espacio. Así mismo también que el juego de batalla naval pueda jugarse entre dos personas y no solo contra la máquina.

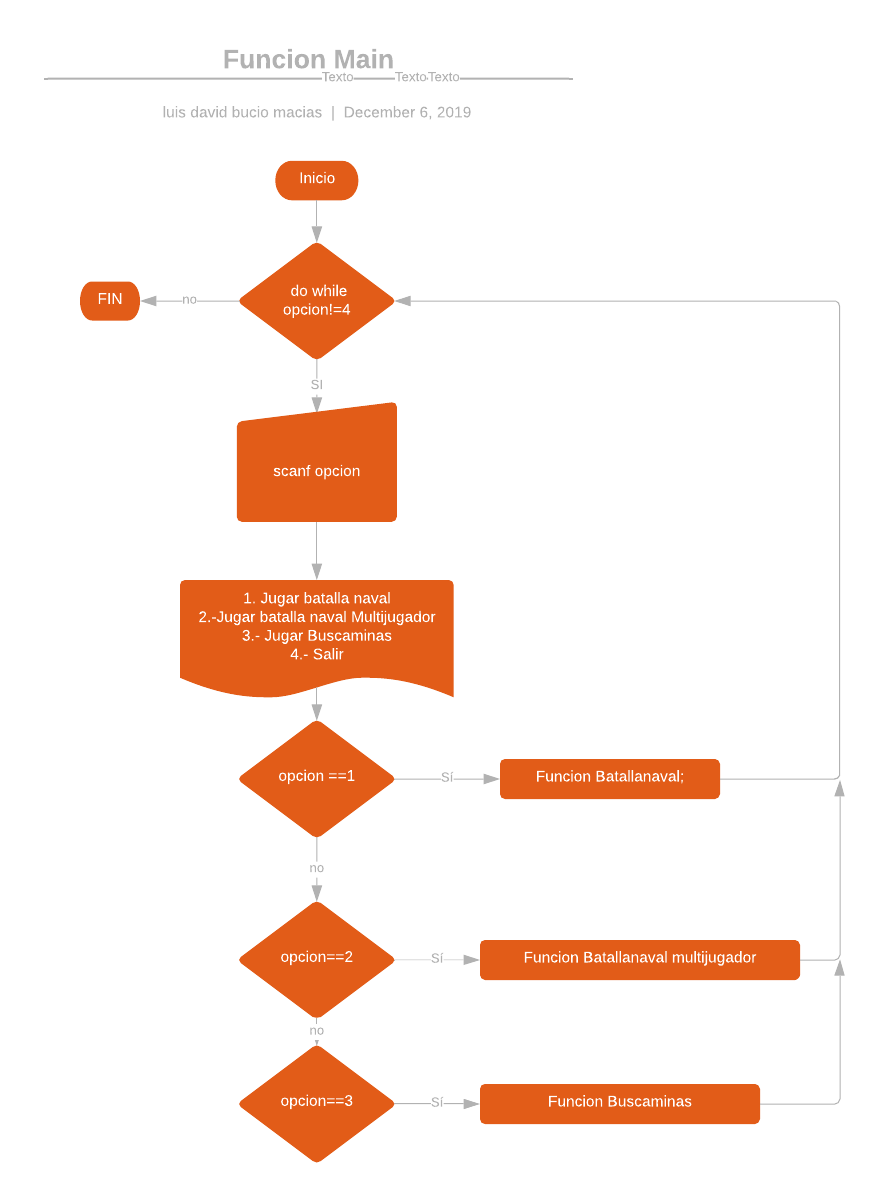
**Solución propuesta:** La solución de este problema la pensamos hacer separando cada cosa del juego mediante funciones y llamarlas en la función main para que todo sea más ordenado y más fácil al momento de juntar todo y crear el juego, así como tomar las medidas adecuadas para que no nos falte tiempo. Aparte para el problema del cliente, servidor la solución será dedicar bastante tiempo en leer y entender la documentación de las librerías externas que vamos a ocupar

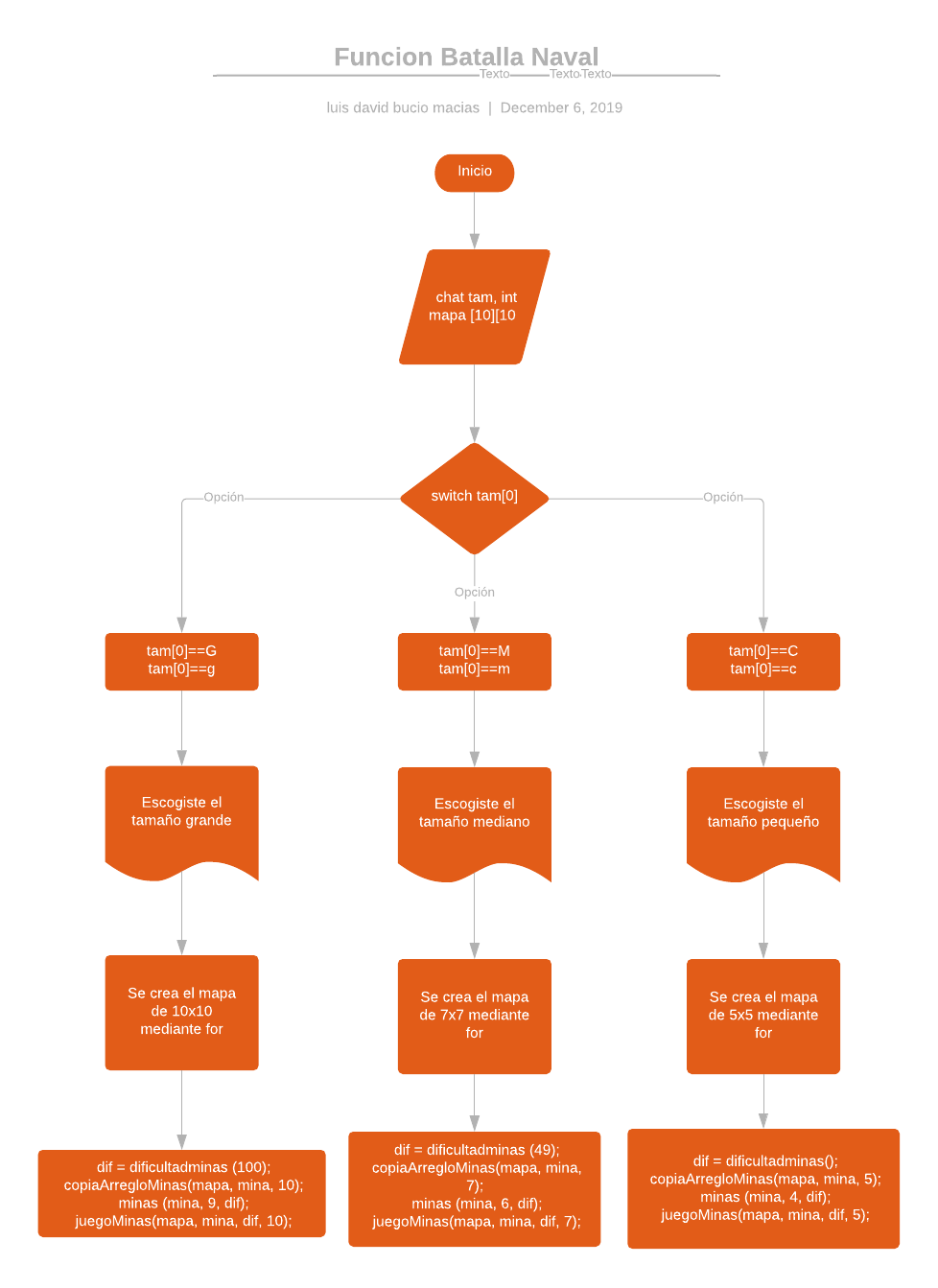
**Borradores de interfaz:**

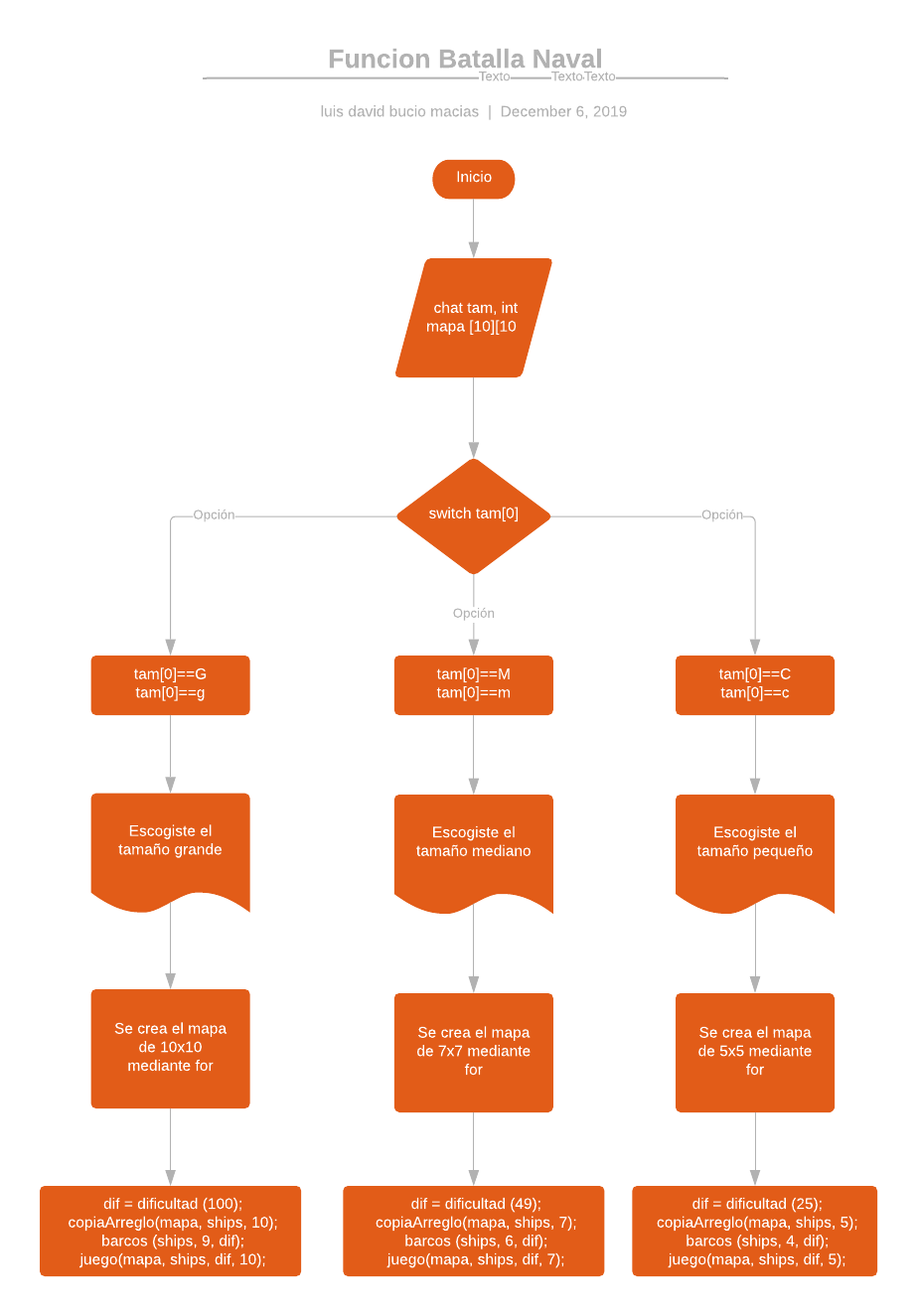


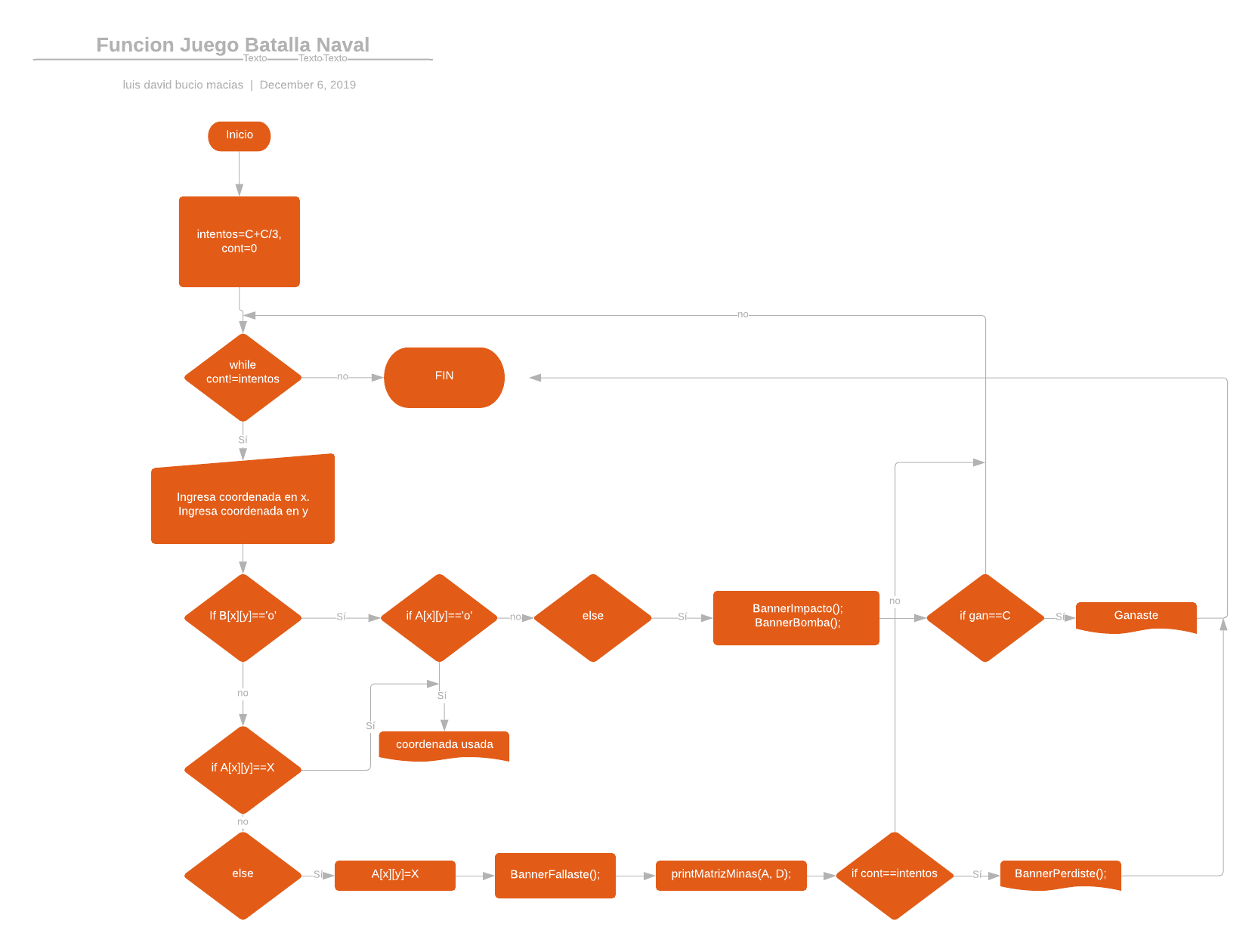
****

**Diagramas de flujo**

****

****

****



# 

# 

# Herramientas y recursos necesarios

* Librerias stdio.h, stdlib.h, time.h, winsock2.h y la libreria externa ws2\_32

● Ciclos:

○ For para llenar nuestros arreglos

○ For Para recorrer nuestras matrices

* Condicionales
* Funciones
* ASCII art

|  |  |  |  |
| --- | --- | --- | --- |
| **Herramienta o Recurso** | **Descripción y utilidad en el proyecto** | **Links a tutoriales y documentación** | **¿Qué harán para aprender a usarlo?** |
| **Librería stdio.h y stdlib.h** | **Básicamente es la librería principal que se suele usar en C para las funciones como printf y scanf** | [**https://es.wikipedia.org/wiki/Stdlib.h**](https://es.wikipedia.org/wiki/Stdlib.h)  [**https://es.wikipedia.org/wiki/Stdio.h**](https://es.wikipedia.org/wiki/Stdio.h) | **Creemos que ya sabemos usar esta librería por todo lo que hemos visto en el curso** |
| **Librería time.h** |  | [**https://es.wikipedia.org/wiki/Time.h**](https://es.wikipedia.org/wiki/Time.h)  [**https://www.tutorialspoint.com/c\_standard\_library/time\_h.htm**](https://www.tutorialspoint.com/c_standard_library/time_h.htm)  [**https://www.geeksforgeeks.org/time-h-header-file-in-c-with-examples/**](https://www.geeksforgeeks.org/time-h-header-file-in-c-with-examples/) | **Viendo videos de YouTube de cómo se usa en diferentes casos y leyendo documentación de esta librería en internet.** |
| **Winsock2.h y ws2\_32** | **Librería necesaria para el uso de sockets y así poder enlazar nuestros dispositivos para el intercambio de datos.** | [**https://gist.github.com/piscisaureus/906386**](https://gist.github.com/piscisaureus/906386)  [**https://docs.microsoft.com/es-mx/windows/win32/api/winsock2/nf-winsock2-wsasetblockinghook**](https://docs.microsoft.com/es-mx/windows/win32/api/winsock2/nf-winsock2-wsasetblockinghook)  **http://cursos.iteso.mx/mod/resource/view.php?id=1104513** | **En esta librería no hay muchos videos por lo que nos tocara leer mucha documentación de esta.** |
| **Funciones** | **Haremos funciones para cada cosa que haga el programa para tener más ordenado el juego y el programa mismo.** | [**https://es.wikibooks.org/wiki/Programación\_en\_C/Uso\_de\_funciones**](https://es.wikibooks.org/wiki/Programación_en_C/Uso_de_funciones)  [**https://www.mheducation.es/bcv/guide/capitulo/8448148681.pdf**](https://www.mheducation.es/bcv/guide/capitulo/8448148681.pdf)  **http://cursos.iteso.mx/pluginfile.php/1342871/mod\_resource/content/2/Sesión%2016.pdf** | **Creemos que esta ya sabemos usarla bastante bien, por lo visto en este curso y en anteriores, pero de todos modos veremos algunos videos para casos específicos** |

# Resultados

## Contenido del paquete entregable (.zip)

En el zip estarán las siguientes cosas:

El archivo .c del proyecto donde vendrá el código que hicimos.

El .exe del proyecto el cual servirá para abrir el programa e iniciarlo para poder jugar cualquiera de los dos juegos.

## Instrucciones de compilación y uso

Para compilar el programa se necesita abrir el .c en algún ID y usar un compilador, en caso de que esto no sea posible, tendremos que ingresar directamente por consola a donde este almacenado el archivo y después se va a ingresar el comando “gcc NombreDelArchivo.c –o NombreDelArchivo –lws2\_32”

Se utilizó una librería externa llamada ws2\_32, este se debe agregar yéndose a las opciones del compilador y estando ahí en las librerías del compilador agregar en la zona de Librerías -l lo siguiente “Ws2\_32” sin las comillas.

1. Al iniciar el programa (el .exe) aparecerá un menú donde el usuario podrá escoger entre 4 opciones:
2. Jugar batalla naval individual
3. Jugar batalla naval multijugador,
4. Jugar buscaminas
5. Salir
6. El usuario deberá ingresar el numero alado de la opción para ingresar a cada opción. Una vez que el usuario ingrese el numero mediante el teclado el programa ejecutara el respectivo juego. (Abajo se muestra el funcionamiento de cada juego)

* Batalla Naval Individual
  + El usuario deberá configurar su juego, ingresando el tamaño y la dificultad.
  + Posteriormente, el usuario deberá ingresar las coordenadas que desea atacar, a lo largo del juego se imprimirán mensajes dependiendo de si pierda, impacta algo o si su tiro falla.
  + El juego finaliza con la impresión del mensaje “FIN DEL JUEGO”
* Batalla Naval multijugador
  + En el caso del multijugador, la cantidad de barcos y el tamaño del mapa son predefinidos.
  + El usuario deberá ingresar sus barcos y posteriormente tendrá que coordinarse con su compañero para decidir quién envía y quien recibe coordenadas.
  + Dependiendo del resultado se imprimirá en la pantalla del usuario los impactos, los tiros fallidos y en la pantalla del contrincante si le impactó o falló.
* Buscaminas:
  + El usuario configurara su juego, ingresando el tamaño y la dificultad de este.
  + Luego, el usuario ira ingresando coordenadas mediante el teclado y a lo largo del juego se imprimirán mensajes diciendo si le diste o no a una mina.
  + El juego finaliza al darle a una mina o al revelar todas las casillas que no tienen mina. Dependiendo de lo que pase se mostrara un mensaje diciendo perdiste o ganaste y al finalizar un mensaje con “FIN DEL JUEGO”
* Salir
  + Finaliza el juego.

ERRORES:

Batalla naval individual:

* Hay un error que conocemos si el usuario ingresa una coordenada más alta a la del tablero mostrada se contara como fallo y se le restara las oportunidades.

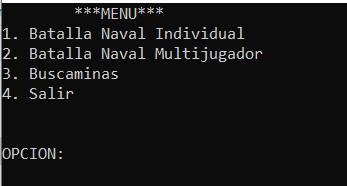
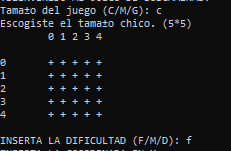
Batalla naval multijugador:

* EL programa puede fallar si los usuarios no se sincronizan a la hora de enviar y recibir coordenadas, de igual forma, si el usuario ingresa coordenadas incorrectas, esto se marcará como un fallo.
* Puedes ingresar muchas veces la misma coordenada, de hacer esto tendrías que bombardear el mismo lugar 20 veces hasta que te aparezca que ganaste

Buscaminas:

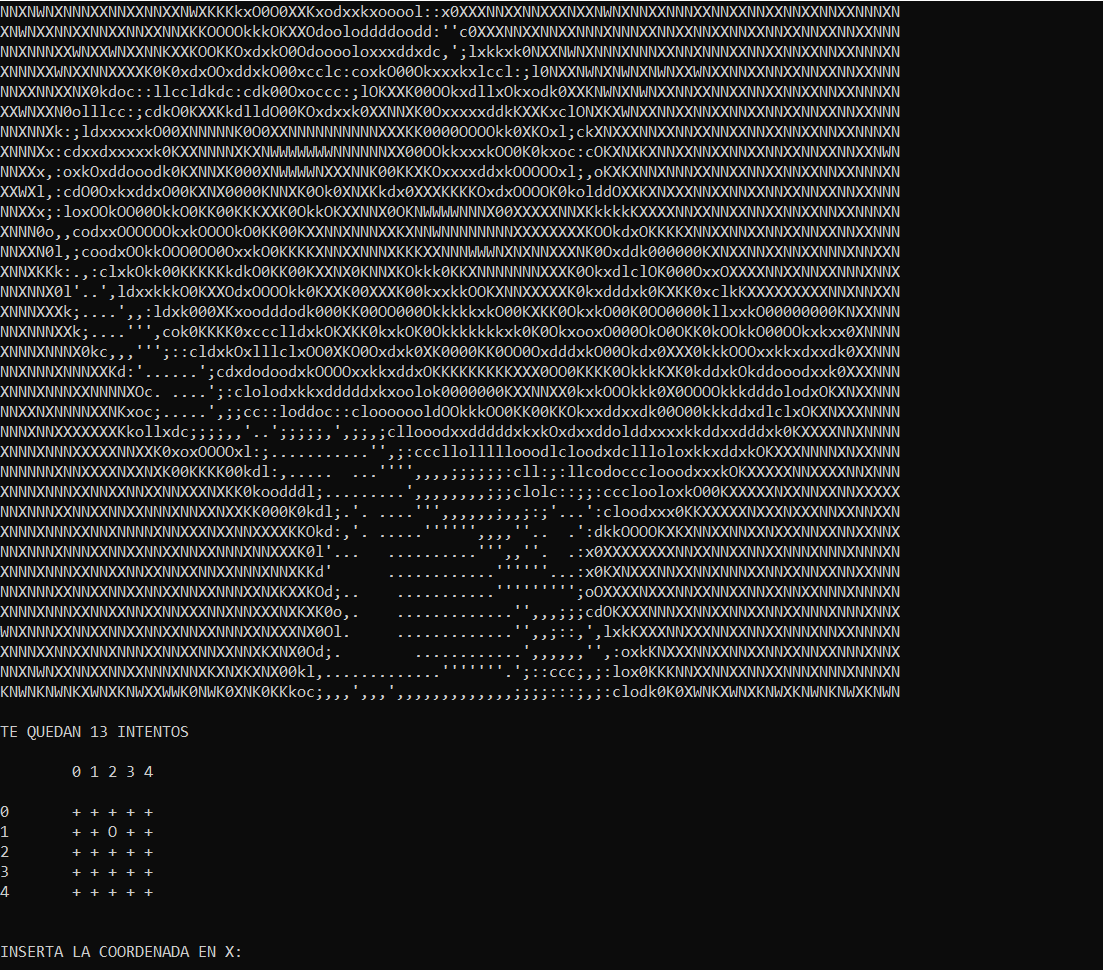
* Hay un error que conocemos si el usuario ingresa una coordenada más alta a la del tablero mostrada que no le diste a ninguna mina, y esto puede llegar a afectar a la condición de victoria.

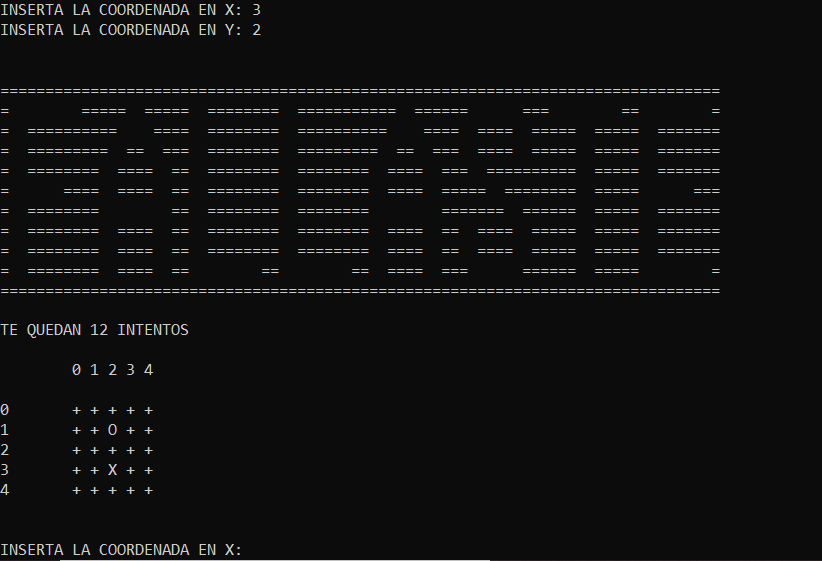
## Ejemplos de Uso

El menú donde se le da la opción al usuario.

Cuando impactas un barco en batalla naval individual

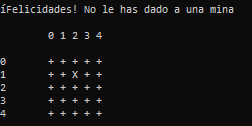
Escoger el tamaño y la dificultad





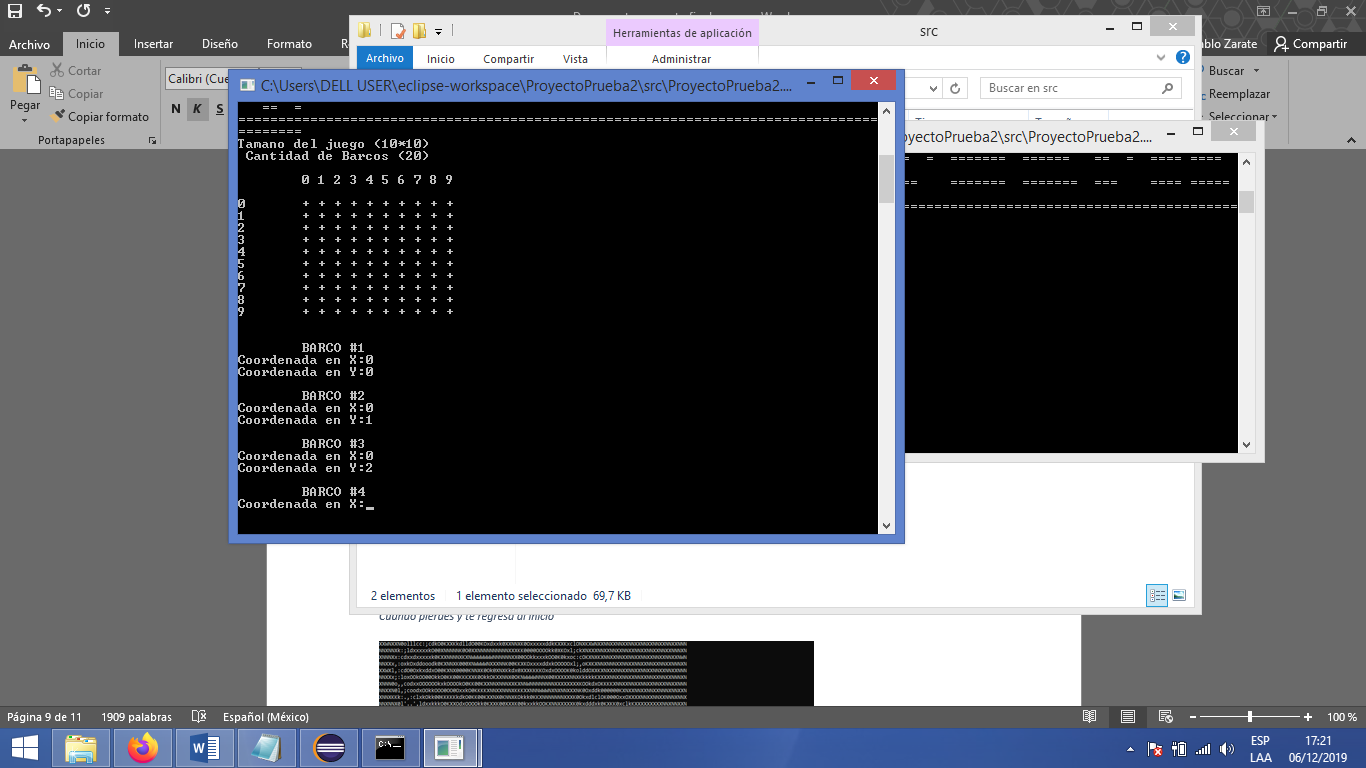
Cuando pierdes y te regresa al inicio

Cuando fallas el disparo en Batalla Naval Individual

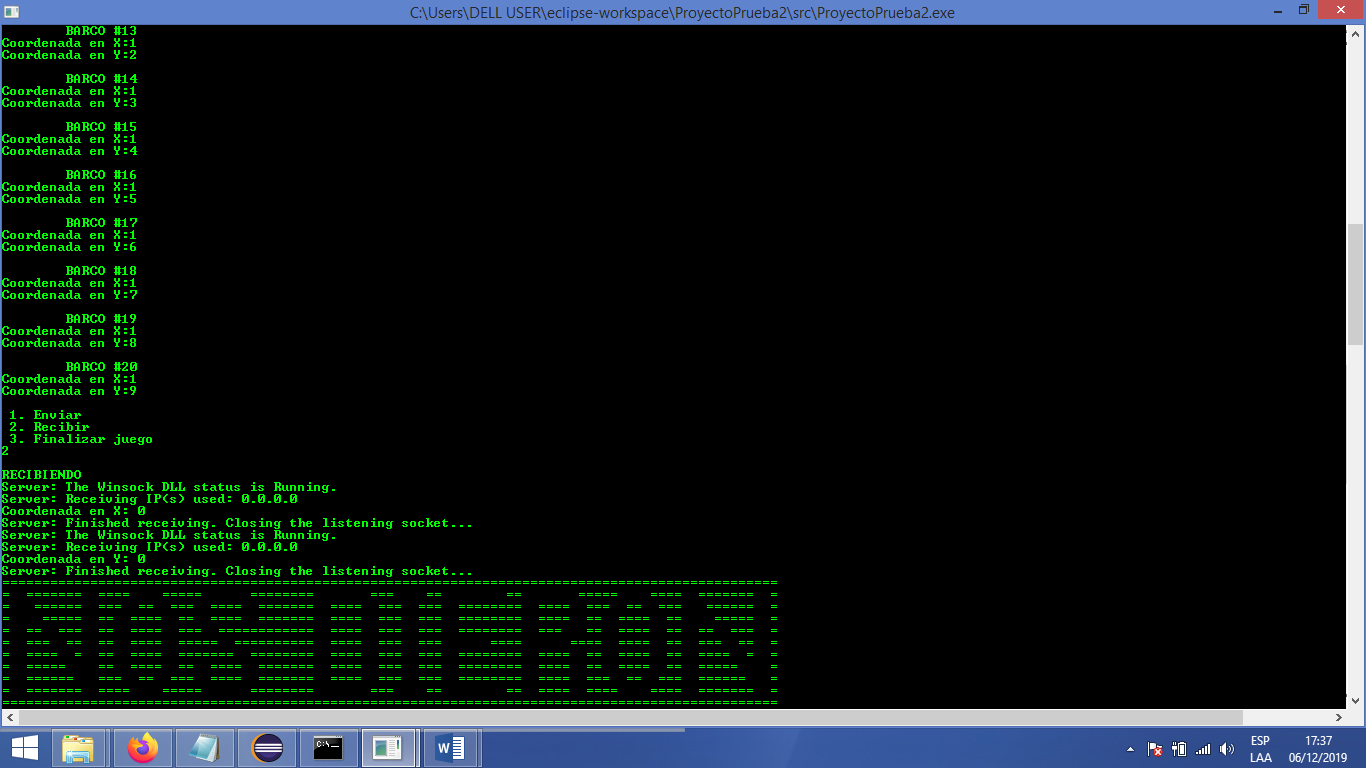


Cuando no le das a una mina

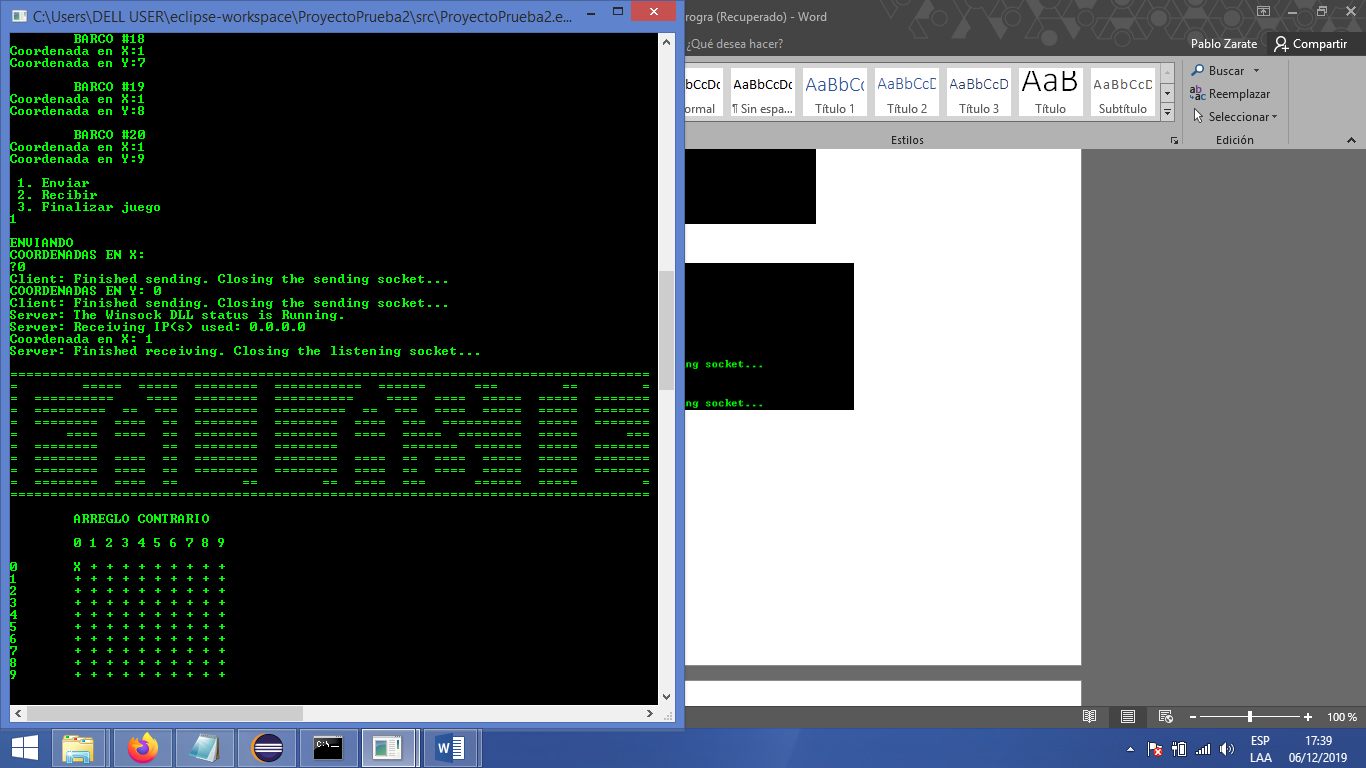
Cuando le das a una mina en buscaminas (Se muestran las posiciones de las bombas)



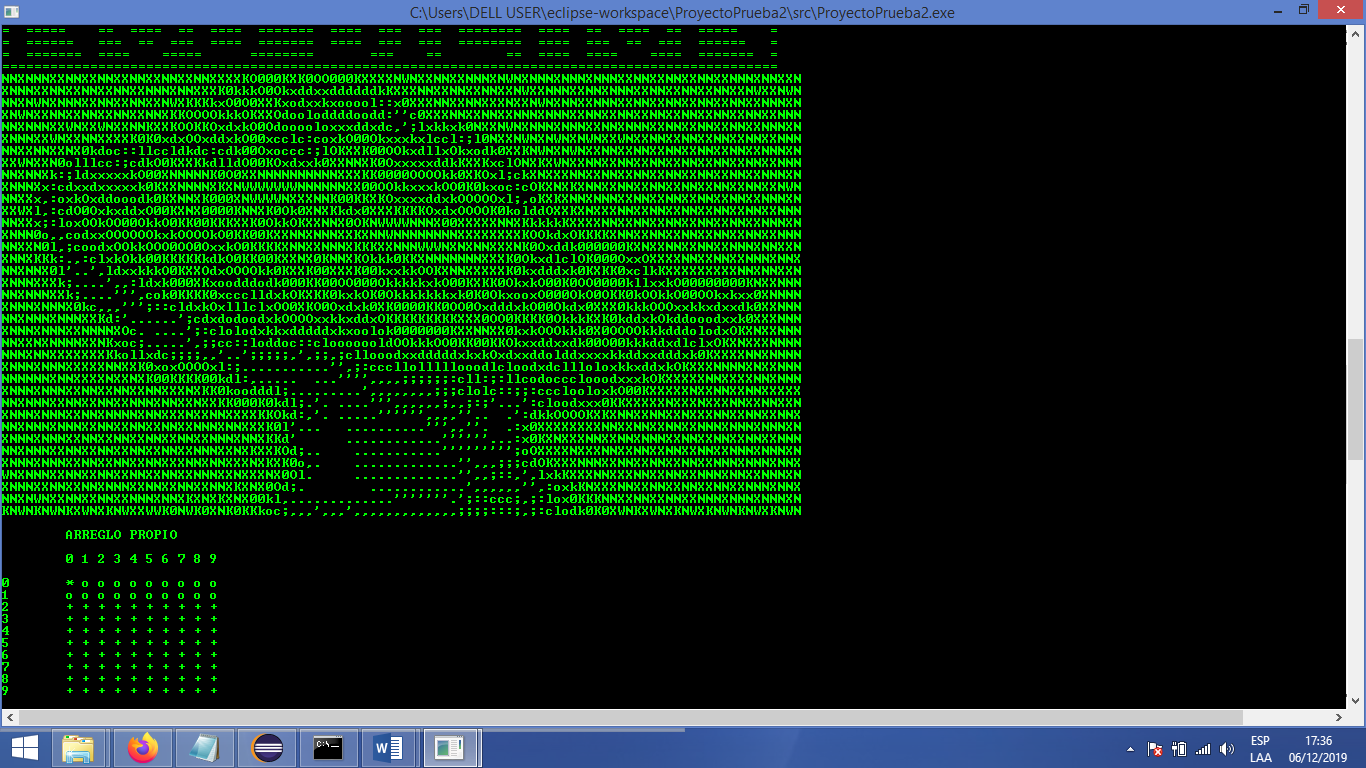
Insertando barcos



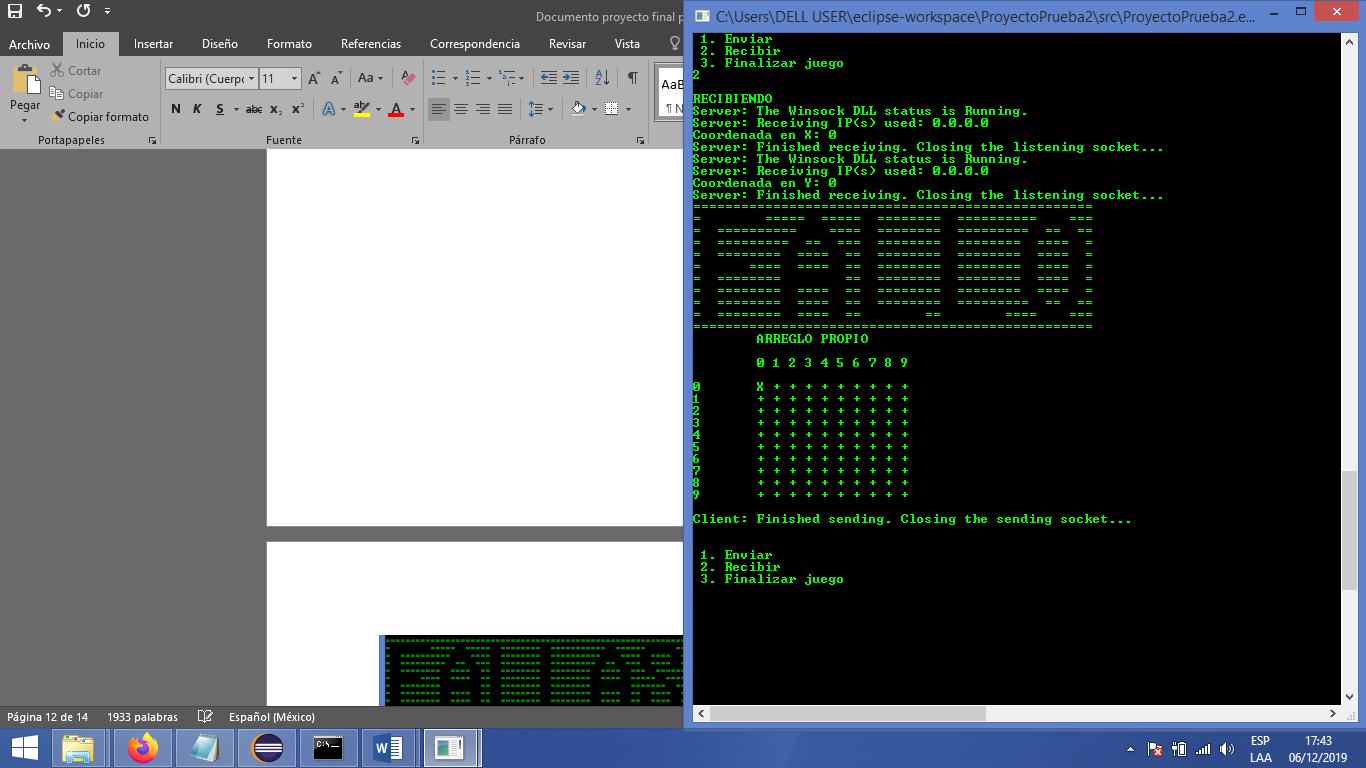
Cuando recibes coordenadas



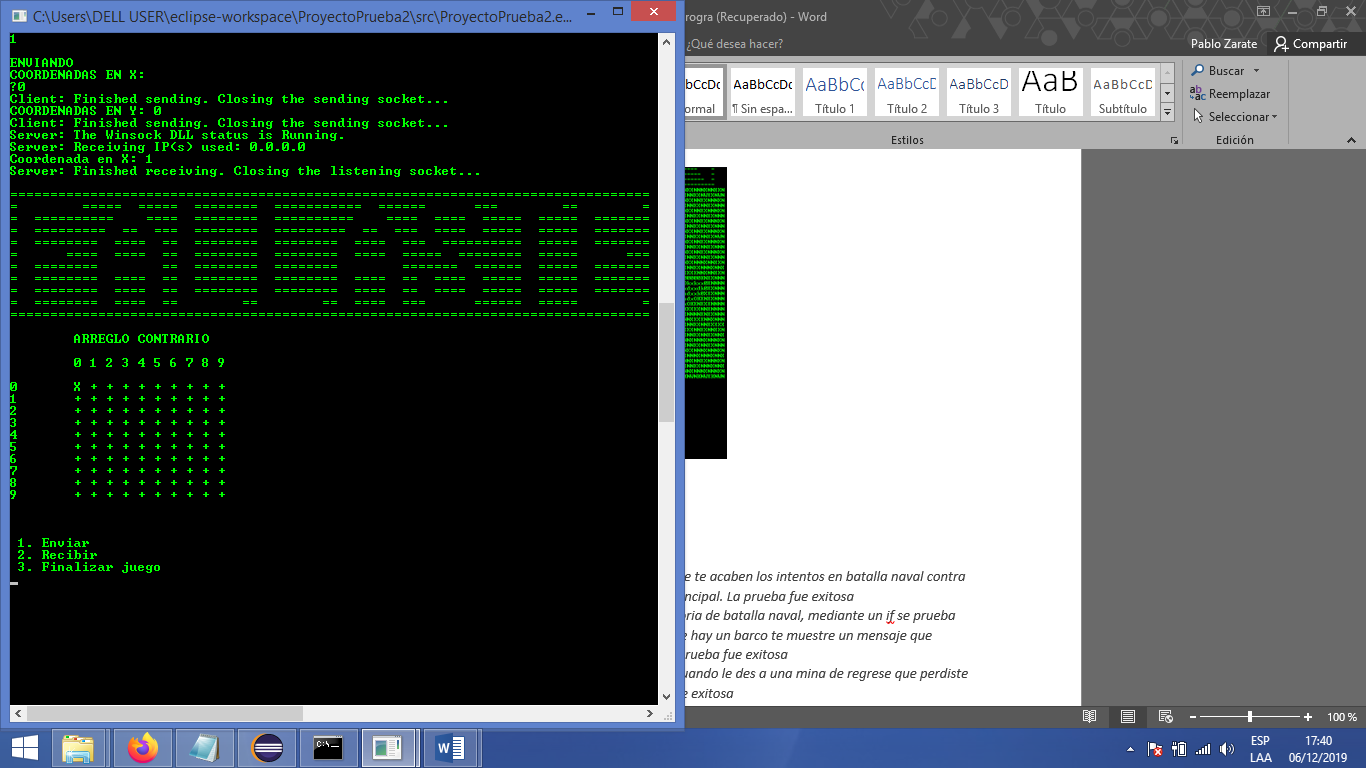
Cuando envías coordenadas



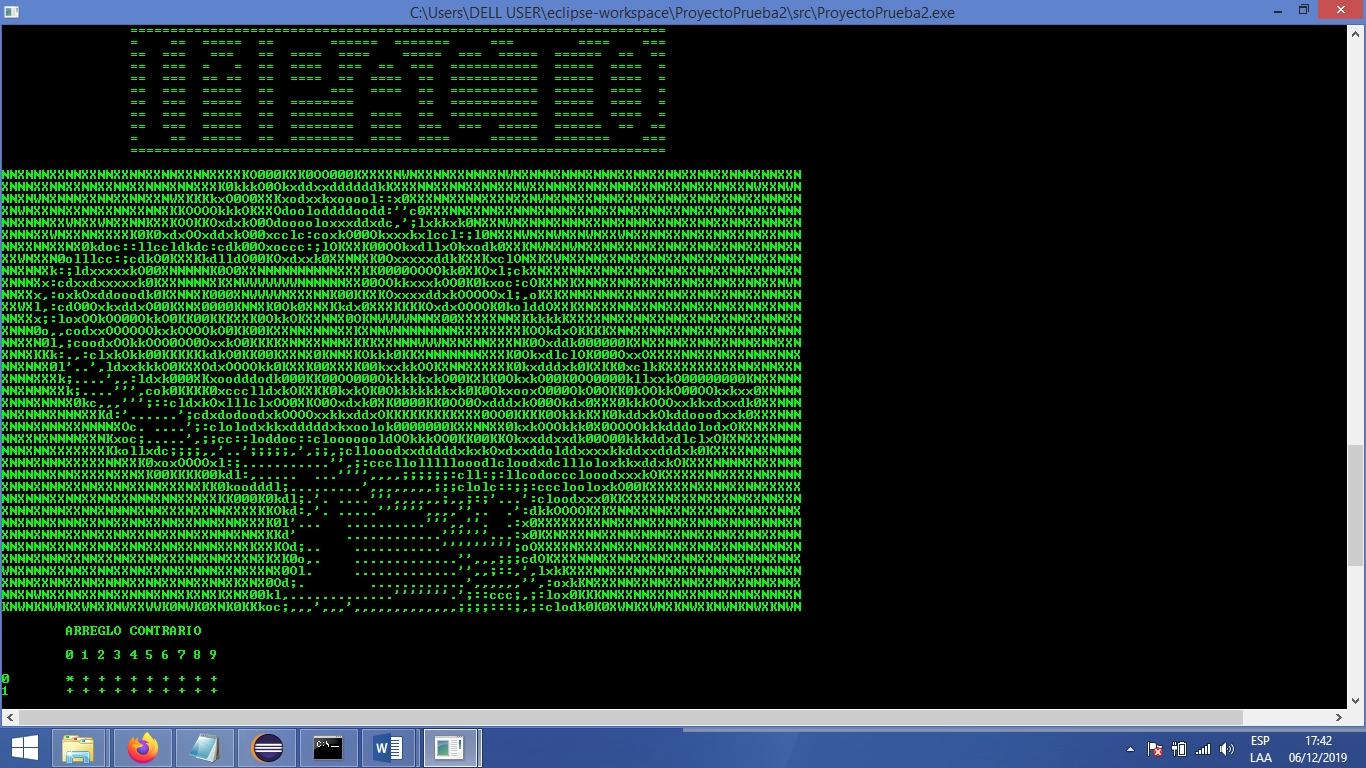
Cuando el cliente impacta uno de nuestros barcos



Cuando fallan el tiro



Cuando tu tiro falla



Cuando tu tiro acierta

Pruebas Realizadas

1. La primera prueba fue probar que cuando se te acaben los intentos en batalla naval contra la maquina pierdas y te regrese al menú principal. La prueba fue exitosa
2. La segunda prueba fue la condición de victoria de batalla naval, mediante un if se prueba que cuando le des a todas las casillas donde hay un barco te muestre un mensaje que ganaste y te regrese al menú principal. La prueba fue exitosa
3. La tercera prueba fue en buscaminas que cuando le des a una mina de regrese que perdiste y te mande al menú principal. La prueba fue exitosa
4. La cuarta prueba fue la condición de victoria en buscaminas, mediante un if que resta las casillas totales menos las minas que hay y si se cumple con un contador te muestra que ganaste y te lleve al menú principal. La prueba fue exitosa.
5. La quinta prueba es la batalla naval multijugador, verificamos que los programas reciban coordenadas y las comparen en sus arreglos locales, en base a esto envíen un número que indica si fue un ataque exitoso o no, posteriormente el contrincante recibe esto y le imprime en su pantalla el resultado. La prueba fue exitosa

# Conclusiones

Pablo Zárate: Yo creo que este proyecto fue un reto ya que nos hizo aplicar todo lo visto durante el curso, así como tener que buscar información en distintas fuentes, tener que buscar los errores, etc.

Me di cuenta de que ejercitas mucho tu lógica y tienes que desarrollar un pensamiento estructurado para no perderte entre tus pensamientos y tantas líneas de código, sin duda lo que me llevo de esta materia es la forma de pensar y de atacar problemas.

Luis David Bucio: Creo que este proyecto fue interesante y bastante útil ya que aplicamos lo visto en este curso y en anteriores y además de tener que buscar cosas que no se alcanzaron a ver en el curso por falta de tiempo.

Creo que lo más difícil de este proyecto fueron las cosas nuevas que tuvimos que investigar por nuestra parte, en este caso la parte del servidor y cliente, y creo que fue lo que mas nos costo y lo que mas tiempo nos llevó. Nos ayudo mucho este proyecto ya que creo que pudimos darnos cuenta de que gracias a este curso si fortalecimos diferentes habilidades de código y diferentes pensamientos para resolver un problema.