



Área de ingeniería en computadores

Lenguajes Compiladores e Interpretes

Tarea programada 3

Manual de usuario

Profesor

Marco Rivera Meneses

Estudiantes:

Gabriel Cerdas Chinchilla

Kendall Adolfo Martínez Carvajal

Daniel Ureña López

I Semestre

Cartago, 10 de abril de 2022

Contenido

Manual de uso.....	3
Introducción	3
Requerimientos mínimos	3
Guía de inicialización.....	3
Estructuras de datos desarrolladas.....	9
Listas.....	9
Struct.....	9
Algoritmos desarrollados	9
Solución general del trabajo	9
Problemas Encontrados	11
Plan de Actividades	11
Conclusiones y recomendaciones	12
Bibliografía	12
Bitácoras.....	12
Kendall Martínez Carvajal	12
Daniel Ureña López	15
Gabriel Cerdas Chinchilla	16

Manual de uso

Introducción

El siguiente manual tiene como objetivo explicar e instruir al usuario para que realice de forma exitosa todos los pasos para poder ejecutar el juego Pole_Position, lo que incluye una breve explicación del sistema operativo requerido para su correcto uso y una guía de inicialización para el servidor al cual se conectarán los clientes (el juego).

Requerimientos mínimos

- ❖ **Hardware:** Para poder utilizar el programa se recomienda una computadora de gama baja-media en adelante, con el fin de que el sistema cargue de forma rápida y no haya problemas con el retraso. En la prueba del juego fuera del ambiente de desarrollo se confirmó que una computadora de 4 gb de Ram y un Intel i3 de tercera generación, con sistema operativo Windows 10 es capaz de correr el juego de forma satisfactoria.
- ❖ **Software:** Para poder ejecutar el juego solo se necesita descargar el código del juego y ejecutar el archivo que se le indicará a continuación en la guía de inicialización. Para el servidor se recomienda tener un editor de código que pueda ejecutar la carpeta del proyecto llamada "Server" (altamente recomendamos usar IntelliJ IDEA" así como le pedimos por favor tenga instalado el "JDK 18" que puede ser descargado desde la pagina misma de "Java".

Guía de inicialización

Una vez haya descargado el código fuente del juego e ingresado a la carpeta principal necesitará dirigirse a la siguiente carpeta:





 ClionGame	10/4/2022 00:36	Carpeta de archivos	
 .gitignore	9/4/2022 17:49	Documento de te...	1 KB
 LICENSE	6/4/2022 17:29	Archivo	35 KB
 README.md	6/4/2022 17:29	Sega Mega Drive	1 KB

Figura 1. Apertura de la carpeta principal del juego.

Fuente. Confección propia.

Una vez haya ingresado en esa carpeta procedemos a inicializar el servidor, para ello vamos a darle click izquierdo en la carpeta de "Sever" y lo abrimos con el editor de código "IntelliJ IDEA", esto porque sin iniciar el servidor el juego no va a iniciar y causará un problema.

.idea	10/4/2022 11:20	Carpeta de archivos	
cmake-build-debug	10/4/2022 00:35	Carpeta de archivos	
res	8/4/2022 22:43	Carpeta de archivos	
Server	9/4/2022 17:49	Carpeta de archivos	
CMakeLists.txt	9/4/2022 17:49	Documento de te...	2 KB
escribir.c	9/4/2022 17:49	Archivo de origen C	2 KB
main.c	10/4/2022 00:36	Archivo de origen C	48 KB
out.txt	8/4/2022 14:53	Documento de te...	2 KB

Figura 2. Ubicación de la carpeta del servidor.

Fuente. Confección propia.

Una vez dentro del editor seleccione la clase llamada PPServer la cual se encuentra dentro de la carpeta “src” y “TEC_PPServer”.

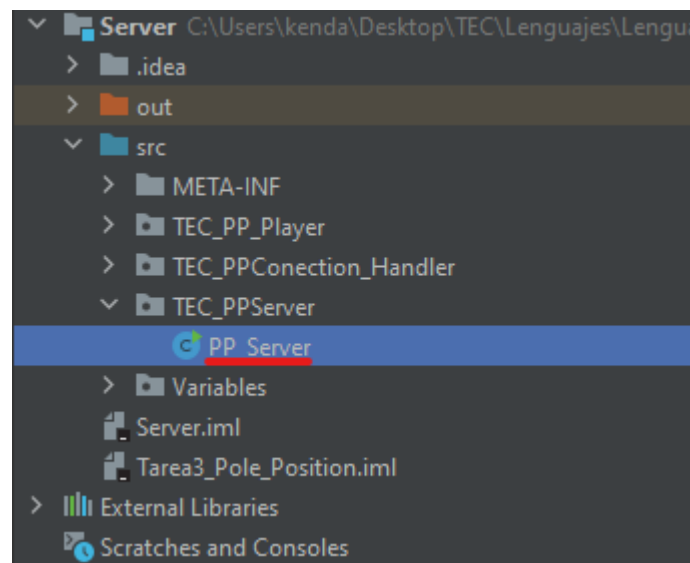


Figura 3. Dirección de carpetas de la clase PP_Server.

Fuente. Confección propia.

Una vez aquí siga todos los pasos de su IDE para poder compilar el código, en IntelliJ IDEA basta con hacer lo siguiente:

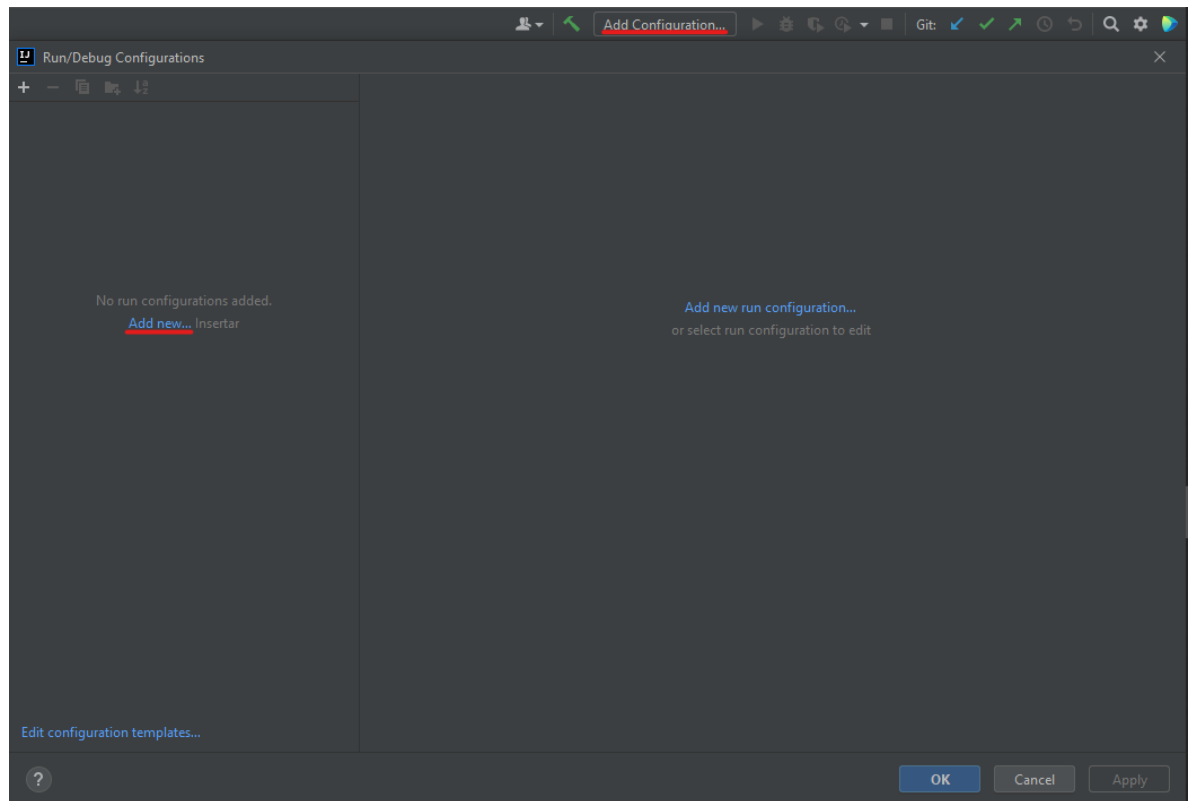


Figura 4. Configuración para correr el servidor en IntelliJ IDEA.

Fuente. Confección propia.

Lo siguiente es añadir configuración de tipo aplicación y seleccionamos como main la dirección `TEC_PPServer.PP_Server` como se ve en la siguiente imagen.

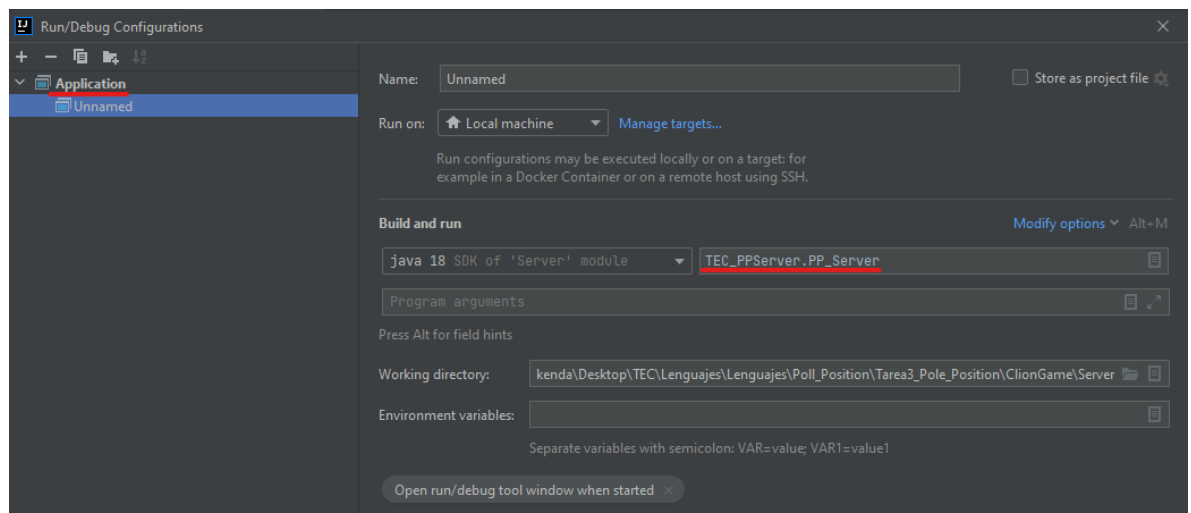


Figura 5. Colocación de las configuraciones del compilador.

Fuente. Confección propia.

Una vez configurado a la derecha tendremos habilitado los siguientes botones, ahora funcionaría con solo darle al botón de play.



Figura 6. Botón de inicio para correr el programa servidor.

Fuente. Confección propia.

Finalmente, ahora volvemos a la carpeta que vimos en la figura 2 y accedemos a la siguiente carpeta de “cmake-build-debug”.

.idea	10/4/2022 11:20	Carpeta de archivos	
<u>cmake-build-debug</u>	10/4/2022 00:35	Carpeta de archivos	
res	8/4/2022 22:43	Carpeta de archivos	
Server	9/4/2022 17:49	Carpeta de archivos	
CMakeLists.txt	9/4/2022 17:49	Documento de te...	2 KB
escribir.c	9/4/2022 17:49	Archivo de origen C	2 KB
main.c	10/4/2022 00:36	Archivo de origen C	48 KB
out.txt	8/4/2022 14:53	Documento de te...	2 KB

Figura 7. Dirección de la carpeta cmake-build-debug.

Fuente. Confección propia.

Después de abrir esa carpeta bastará con ejecutar el “ClionGame.exe” para poder acceder al juego.

.cmake	9/4/2022 00:14	Carpeta de archivos	
_deps	9/4/2022 20:57	Carpeta de archivos	
CMakeFiles	10/4/2022 11:20	Carpeta de archivos	
Testing	9/4/2022 00:14	Carpeta de archivos	
<u>ClionGame.exe</u>	9/4/2022 20:58	Aplicación	2,623 KB

Figura 8. Ubicación de ClionGame.exe.

Fuente. Confección propia.

Al abrirlo, te encontraras con la siguiente ventana:

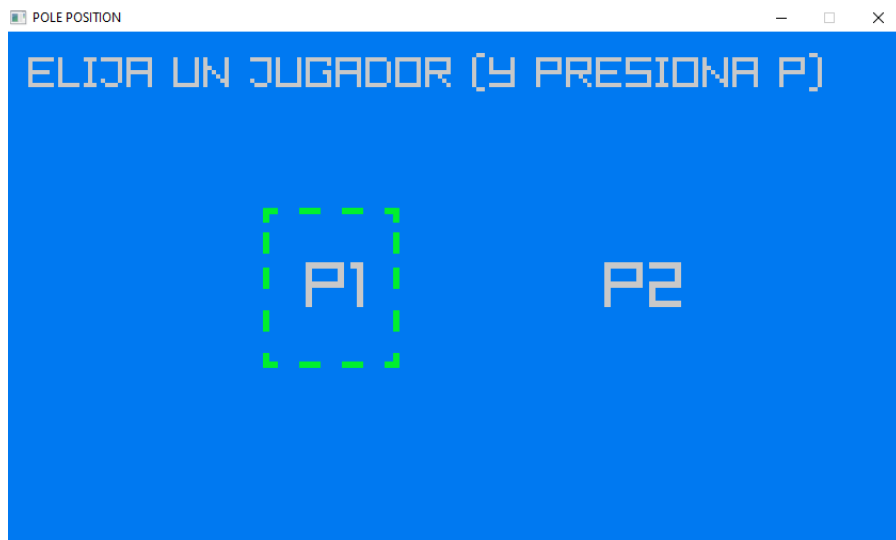


Figura 9. Pantalla de elección de jugador.

Fuente. Confección propia.

Mediante el uso de las teclas direccionales (flecha izquierda o derecha), podrás elegir cual jugador puedes ser, jugador 1 o jugador 2. Una vez ubiques el cuadro verde de selección sobre el jugador que quieres ser, presiona la tecla “P” para elegirlo de manera definitiva. Seguido de esto, observarás una ventana como esta:

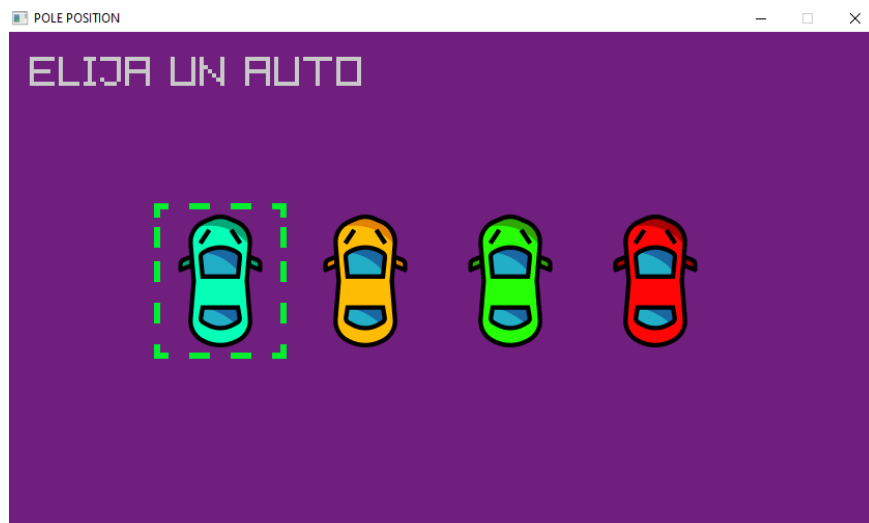


Figura 10. Pantalla de elección de auto.

Fuente. Confección propia.

De la misma forma que la pantalla anterior, mediante el uso de las teclas direccionales (flecha izquierda o derecha), podrás elegir cual auto podrás usar (celeste, amarillo, verde, o rojo). Una vez ubiques el cuadro verde de selección sobre el auto que quieres, presiona la tecla “ENTER” para elegirlo de manera definitiva. Seguido de esto, observarás una ventana como esta:

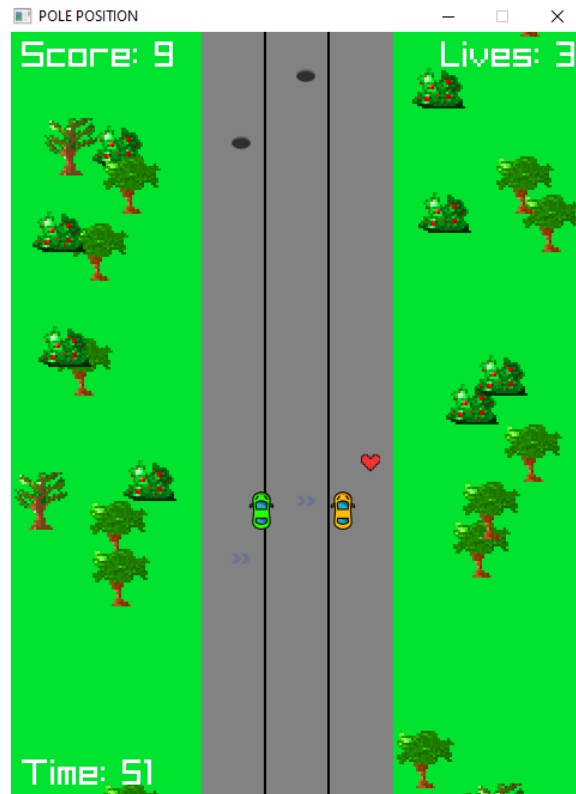


Figura 10. Pantalla de juego.

Fuente. Confección propia.

Esta es la pantalla principal del juego. Aquí estarás compitiendo contra tu rival, puedes adelantarlos mediante el uso de los turbos, los cuales aparecen en la calle desde la parte superior de la pantalla, y puedes identificarlos por las flechas azules que aparecen en la calle, solo tienes que conducir encima de ellos. Para manejar tu vehículo, tienes que usar las teclas direccionales (flechas izquierda y derecha) de tu teclado. Ten cuidado de los huecos, ya que, si conduces por encima de ellos, se reducirá tu velocidad por cierto tiempo y pierdes una vida. Empiezas con 3 vidas, puedes recuperarlas una a una si conduces por encima de ellas cuando aparezcan en la calle, pero solo puedes tener un máximo de 5 vidas. En total es un solo minuto de juego, y vas ganando un punto con cada segundo que pase, y cuando se acabe ese tiempo tienes que asegurarte de que estar al frente a tu rival, ya que también ganas puntos con la posición en la que te encuentres. Si tienes mas puntos que tu rival, ganas!

Estructuras de datos desarrolladas

Listas

Se utilizan para poder almacenar los objetos, ya sean poderes o datos correspondientes a los atributos, por lo que las listas se utilizan en el trabajo para almacenar y manipular datos de diferentes partes del código.

Struct

En el trabajo se utilizan diferentes tipos de estructuras, una de las principales corresponde a la del cuadrado, esta contiene la información del objeto el cual al ser trabajada más adelante permite definir las hit boxes de las diferentes entidades del juego.

Algoritmos desarrollados

Solución general del trabajo

Para la solución del trabajo se realizó la conexión del cliente con el servidor utilizando sockets, para poder comunicar datos entre ellos se utilizó un txt, el cual funciona para escribir la respuesta que contiene los datos necesarios para iniciar el juego.

Una vez el cliente ya cuenta con los datos necesarios para iniciar el juego el juego es iniciado y comienza la carrera donde los huecos, corazones y turbos se general aleatoriamente en todo el mapa conforme este se desarrolla.

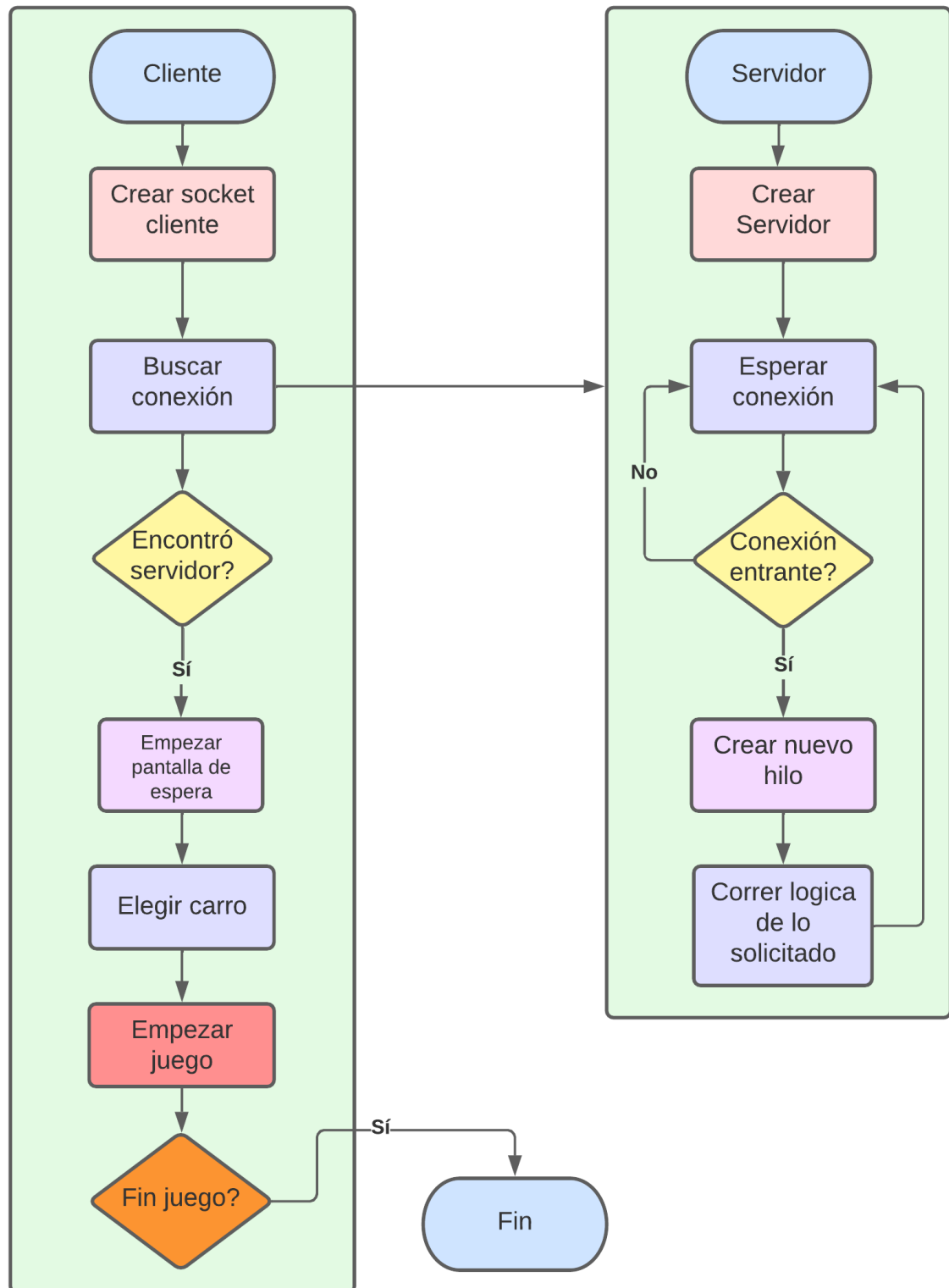


Figura 11. Diagrama de flujo solución general.

Fuente. Confección propia.

Problemas Encontrados

Problema	Intentos de solución	Solución	Recomendaciones	Conclusión	¿Corregido?	Bibliografía consultada
El servidor logra recibir mensajes del cliente, pero no logra responderlos.	23	No	Cambiar de enfoque y centrarse en la lectura y escritura de un archivo donde se puedan obtener los mensajes del cliente y servidor.	La conexión entre sockets funciona pero poder transmitir información varía mucho entre versiones del código y compiladores.	No	Profesor y Geeks for Geeks (2022)
Conectar dos clientes al mismo tiempo	4	No	Utilizar un lenguaje más adecuado para crear un juego y su cliente para un servidor.	La manipulación de variables es muy compleja si el servidor tiene muchas limitantes para enviar información y recibirla.	No	Kochański, P. (2022)
Lectura de archivos .txt	7	Sí	El lector de texto puede comportarse de forma incorrecta cuando otra función interfiere en la misma.	Revisar el código y comprender cada uno de los pasos que realiza puede llevar a encontrar bugs de una librería oficial del lenguaje.	Sí	("Lectura y Escritura de Ficheros en Java - ChuWiki", 2022)
Escritura de archivos .txt	10	Sí	El escritor de texto puede borrar el contenido del mismo causando que sea mejor definir el escritor justo antes de escribir en el archivo.	Revisar el código y comprender cada uno de los pasos que realiza puede llevar a encontrar bugs de una librería oficial del lenguaje.	Sí	("Lectura y Escritura de Ficheros en Java - ChuWiki", 2022)
Instalación de la librería gráfica	6	Sí	Algunos editores de código pueden utilizar otros métodos de compilación para añadir librerías.	Utilizar un editor de código que añada dependencias al trabajo puede favorecer mucho la instalación de un ambiente.	Sí	("GitHub - raysan5/raylib: A simple and easy-to-use library to enjoy videogames programming", 2022)
Funciones repetidas dentro de las librerías	7	Sí	Asegurarse de antemano que no hay funciones repetidas entre librerías.	Si este problema sucede lo ideal es matar las funciones de una de las librerías, de esta forma quitas el conflicto aunque puede llegar a ser peligroso.	Sí	Fork, G. (2022)
El cliente a veces causa que una tecla se quede pegada	1	No	Tener mucho cuidado al ejecutar el .exe	A veces las librerías pueden causar que no se cierren todos los punteros provocando problemas en la computadora	No	-

Figura 12. Problemas encontrados.

Fuente. Confección propia.

Plan de Actividades

Tipo de trabajo	Responsable	Tarea a realizar	Tiempo dedicado	Fecha
Programación	Gabriel Cerdas	Diseño del escenario.	0.5 h	30/3/2022
	Daniel Ureña	Creación de pantalla de selección de automóvil.	2 h	8/4/2022
	Gabriel Cerdas	Movimiento del vehículo en el juego.	1 h	31/3/2022
		Movimiento de ítems en la pantalla del juego.	3 h	31/3/2022
		Ítems: Mecanismo de huecos.	1 h	31/3/2022
		Ítems: Mecanismo de vidas.	1 h	31/3/2022
		Ítems: Mecanismo de turbos.	1 h	31/3/2022
		Movimiento del vehículo sobre tierra.	2 h	31/3/2022
	Daniel Ureña	Visualización de los jugadores enemigos.	5 h	8/4/2022
	Gabriel Cerdas	Diseño de proyectiles de jugadores (UI)	0.5 h	3/4/2022
	Kendali Martínez	Diseño de proyectiles de jugadores (Mecanismo)	1.5 h	31/3/2022
		Escritura de archivos .txt para enviar al servidor.	1 h	8/4/2022
		Retransmisión de archivos .txt a los demás clientes.	6 h	8/4/2022
		Trigger de aparición de ítems desde servidor.	-	-
		Diseño y creación de un cliente en C.	2.5 h	6/4/2022
Documentación	Daniel Ureña	Diseño y creación del servidor en Java.	3 h	2/4/2022
		Describir algoritmos implementados.	1 h	9/4/2022
		Actualizar y revisar las bitácoras en el documento entregable.	2 h	9/4/2022
		Describir estructuras de datos utilizados en el proyecto.	1 h	9/4/2022
		Describir la explicación del algoritmo general de la solución.	0.5 h	9/4/2022
		Redacción de problemas encontrados durante la solución.	1 h	9/4/2022
		Dar formato a la documentación técnica.	0.5 h	3/4/2022
		Redactar la sección de conclusiones del trabajo.	1 h	9/4/2022
		Redactar la sección de problemas sin solución del trabajo.	0.5 h	9/4/2022
		Escribir la sección de recomendaciones.	0.7 h	9/4/2022
	Kendali Martínez	Añadir bibliografía consultada durante el proyecto.	0.6 h	9/4/2022
		Crear y actualizar el plan de actividades.	2 h	9/4/2022
		Escribir manual de usuario.	2 h	10/4/2022

Figura 13. Plan de actividades

Fuente. Confección propia

Conclusiones y recomendaciones

Durante la realización del proyecto se encontraron diferentes obstáculos, uno de ellos fue la mala explicación sobre cómo utilizar los métodos para leer y sobre escribir un archivo utilizando las librerías de Java, después de realizar un debug y estudiarlo a fondo nos dimos cuenta de que al llamar a la función "PrintWriter" los ".txt" se reiniciaban por lo que este problema se soluciona al mover la declaración justo antes de donde deseamos empezar a escribirlo.

Otra complicación y recomendación es la utilización de un observador para poder obtener la información siempre que el cliente haga un intento por llamar a la conexión con el servidor ya que transmitir información entre "C" y "Java" es más complicado de lo que parece y nunca se encontró forma de enviar información de forma eficiente entre el cliente y el servidor de no ser por la observación de cambios en un archivo.

Bibliografía

- Fork, G. (2022). Building conflicts with windows.h for some users · Issue #5 · greenfork/nimraylib_now. Retrieved 10 April 2022, from https://github.com/greenfork/nimraylib_now/issues/5
- Kochański, P. (2022). socket programming multiple client to one server. Retrieved 10 April 2022, from <https://stackoverflow.com/questions/10131377/socket-programming-multiple-client-to-one-server>
- tbag/migration.sh at v1.0.0 · osom8979/tbag. (2022). Retrieved 10 April 2022, from <https://github.com/osom8979/tbag/blob/v1.0.0/dep/raylib/migration.sh>
- TCP Server-Client implementation in C - GeeksforGeeks. (2022). Retrieved 10 April 2022, from <https://www.geeksforgeeks.org/tcp-server-client-implementation-in-c/>
- Lectura y Escritura de Ficheros en Java - ChuWiki. (2022). Retrieved 10 April 2022, from http://chuwiki.chuidiang.org/index.php?title=Lectura_y_Escritura_de_Ficheros_en_Java
- GitHub - raysan5/raylib: A simple and easy-to-use library to enjoy videogames programming. (2022). Retrieved 10 April 2022, from <https://github.com/raysan5/raylib>

Bitácoras

Kendall Martínez Carvajal

29/03/22

16:00 Comencé con la investigación sobre como comunicar por sockets dos lenguajes diferentes y como hacer que se transfiera información entre ellos.

17:00 Di por concluida la investigación y comencé a preparar el ambiente de desarrollo.

19:00 Terminé de alistar todo lo necesario y ajusté el plan de actividades del proyecto.

30/03/22

16:00 Comencé a escribir el código básico en Java para realizar una prueba de conexión entre Java y C.

16:30 Terminé de programar el servidor y busqué como establecer el cliente de pruebas en C.

17:45 Terminé de escribir el código y precedí con las pruebas.

18:15 Terminé con las pruebas y confirmé la conexión de sockets.

31/03/22

15:00 Empecé a trabajar con el servidor para que pueda aceptar varios clientes y responderle a cada uno por aparte.

16:00 Terminé la modificación al servidor y confirmé la actividad esperada. Me comuniqué con ellos para definir el siguiente paso de trabajo.

16:40 Definimos que el cliente envía y debe recibir información, por lo que procederé a trabajar con esto.

18:40 Después de una investigación concluí que métodos utilizar y como proceder, di por concluida la sesión de hoy.

02/04/22

14:00 Trabajé en la implementación de lo investigado.

16:00 La modificación del cliente y el servidor fue finalizada y se procedió con las pruebas pertinentes.

18:00 La conexión y recepción de un mensaje por parte de C a Java se lleva de forma efectiva, pero al parecer el código en C nunca recibe el mensaje del servidor... Se procede a indagar sobre el tema.

22:00 Después de varios intentos y pruebas decidí dejar de intentar por el día de hoy. No se ha podido encontrar una solución al problema.

03/04/22

14:00 Continué con la búsqueda de una solución.

22:00 A pesar de probar con todo tipo de posibles soluciones no se dio con una solución y se dio por terminada la sesión de hoy.

04/04/22

15:00 Continúe buscando y probando soluciones para el problema de recibir datos desde el servidor.

23:00 No se ha encontrado solución y ahora el servidor se queda en espera y solo devuelve cuando uno de los procesos acaba. Discutiéndolo con mis compañeros acordamos consultar con el profesor.

05/04/22

16:10 Se consultó con el profesor por una posible solución y agendamos una consulta para el día siguiente a las 16:00.

16:15 Se intentó un poco más con el problema.

20:00 Concluimos con las pruebas y no se logró nada.

06/04/22

16:26 Nos conectamos a la consulta.

16:52 El profesor me pidió que revisara si los datos llegan bien al servidor y si la conexión se realiza correctamente, así como que intente utilizar los métodos flush para llevar a cabo él envió de la información.

22:10 Se decidió cambiar el entorno de desarrollo de UNIX a Windows y al adaptar el programa a Windows se volvió a caer en el mismo problema, aun tomando en cuenta las recomendaciones del profesor.

22:13 Se consulto con el profesor para agendar otra consulta.

07/04/22

21:42 Nos reunimos con el profesor y no se encontró el problema, el profesor nos facilitó archivos de ejemplo y nos dio páginas de consulta para comparar nuestro código actual.

23:40 Después de probar varias cosas nos retiramos y lo dejamos por el día.

08/04/22/

10:30 Se ideó un nuevo plan de acción utilizando la escritura de un "txt" como medio de comunicación. Se procedió a ajustar el código.

13:00 Se modificó el servidor y comprobó la funcionalidad del sistema, ahora procedemos a configurar un método para observar el archivo y sobre escribirlo de forma que logre dar la información al cliente.

16:00 Se terminó con la implementación del servidor y la asignación de valores por medio de polimorfismo y POO; procedo a reunirme con mis compañeros para unir los trabajos.

16:45 Se encontró una incompatibilidad con la librería gráfica del proyecto y la librería del socket. Se procedió a encontrar una solución.

09/04/22

01:00 Se dio por concluida la sesión hasta la mañana para seguir buscando una solución al problema.

09:33 Le pregunté al profesor por ayuda.

10:30 Me conecté a trabajar con Gabriel y buscamos una solución al problema.

11:00 Después de investigar encontramos que lo mejor es aplicarle un cambio de nombres a la librería grafica...Daniel se unió a nosotros para resolver el problema.

12:45 Se intentó realizar el cambio de nombres en la librería para los métodos, pero no hubo éxito. Gabriel se retiró para almorzar.

13:45 Procedí a remover los métodos conflictivos desde la librería de "winsock2.h".

13:50 Removí todos los métodos que causaban conflicto y el problema se solucionó. Por lo que decidimos descansar un poco.

15:00 Regresamos a trabajar para confirmar que el cambio de la carpeta "include" de "winsock2.h" resolvió el problema en las computadoras de todos.

15:20 Confirmamos la funcionalidad del trabajo y procedimos con la integración final.

16:20 Se actualizó el plan de actividades.

16:45 Se actualizó la sección de recomendaciones y conclusiones.

17:45 Se añadió la bibliografía consultada.

[Daniel Ureña López](#)

29/03/22

14:00 Se procedió con la investigación de librerías graficas en C que pudieran ser capaces de llevar a cabo el juego de PolePosition.

16:00 Se finaliza la reunión con los compañeros Gabriel Cerdas y Kendall Martínez, se realiza una evaluación sobre las librerías graphics.h y raylib.h, y se decide utilizar Raylib debido a la mayor versatilidad que esta ofrece, además de poder usarse con los editores vs code y Clion, a diferencia de graphics que necesitaba del editor TurboC.

30/03/22

15:00 Procedí a instalar la librería Raylib, y preparar el editor vs code para que trabajara lenguaje c y además también pudiera trabajar con Raylib.

20:00 Se encontraron problemas al querer usar Raylib con vs code, el siguiente día continuaré con la búsqueda de una solución al respecto

31/03/22

14:00 Se procedió con la investigación de librerías graficas en C que pudieran ser capaces de llevar a cabo el juego de PolePosition.

03/04/22

13:00 Continúe trabajando en la confección de la documentación técnica, se avanzó en el documento hasta las 17:00.

06/04/22

13:00 Continúe trabajando en la confección de la documentación técnica, se avanzó en el documento hasta las 17:00.

16:30 El grupo y yo nos conectamos a consulta con el profesor para evacuar ciertas dudas respecto al servidor y en la forma que este opera.

07/04/22

11:30 Se hace otra reunión con los compañeros para continuar trabajando en el juego, se adapta el programa IntelliJ idea para que pueda correr el servidor y así probar su efectividad con un cliente C.

18:00 Se continúa con la documentación técnica del proyecto, se realizan tablas en Excel que incluyan información para agregarlas a la documentación técnica.

21:00 Se logra adaptar el editor CLion para que trabaje con la librería Raylib. Se finalizó de trabajar a las 1:30 horas del día siguiente.

09/04/22

11:00 Se continúa trabajando en el juego, se crea la pantalla de elección de auto, se agregaron muchas características menores como la detección de disparos y se arreglaron ciertos bugs, se finalizó de trabajar a las 0:30 horas del día siguiente.

10/04/22

11:00 Se continúa trabajando en ajustar detalles finales para el juego, como también en finalizar la documentación del juego, se finalizó a las 3:30 horas.

Gabriel Cerdas Chinchilla

29/03/22

14:00 Se comenzó la investigación de librerías gráficas en conjunto con Daniel Ureña, se hace una extensa búsqueda con resultados poco exitosos.

15:00 Se da por finalizada la búsqueda, de esa forma barajando dos opciones de librerías gráficas, una llamada Graphics y la otra Raylib.

16:00 Finalmente se decide utilizar Raylib de esta forma comenzando los preparativos para su uso.

30/03/22

12:00 Comencé a escribir el código base en el cual se implementan escenarios iniciales para el juego.

13:30 Terminé de programar las primeras acciones de jugabilidad y sprites.

31/03/22

15:00 Este día empecé con las colisiones del carro con diferentes sprites del juego.

18:40 Terminé de realizar las respectivas colisiones, además agregué imágenes que representaban los diferentes sprites (Huecos, vidas y turbos).

03/04/22

14:00 Empecé con cambio de velocidades en el juego, además de eso agregué distintos marcadores por la interfaz.

18:00 Finalicé el juego en una etapa de desarrollo beta donde todo lo que hacía falta era con respecto a incluir más jugadores, además se realizó el disparo de balas por parte del vehículo.

04/04/22

15:00 Ayudé a Kendall Martínez a continuar buscando y probando soluciones para el problema de recibir datos desde el servidor.

23:00 No se ha encontrado solución y ahora el servidor se queda en espera y solo devuelve cuando uno de los procesos acaba. Discutiéndolo con mis compañeros acordamos consultar con el profesor.

05/04/22

16:10 Se consultó con el profesor por una posible solución y agendamos una consulta para el día siguiente a las 16:00.

16:15 Se intentó un poco más con el problema.

20:00 Concluimos con las pruebas y no se logró nada.

06/04/22

16:26 Nos conectamos a la consulta.

16:52 El profesor me pidió que revisara si los datos llegan bien al servidor y si la conexión se realiza correctamente, así como que intente utilizar los métodos flush para llevar a cabo él envío de la información.

22:10 Se decidió cambiar el entorno de desarrollo de unix a Windows y al adaptar el programa a Windows se volvió a caer en el mismo problema, aun tomando en cuenta las recomendaciones del profesor.

22:13 Se consulto con el profesor para agendar otra consulta.

23:50 Se logró usar la librería de Raylib en Clion, cosa que era imposible para los miembros del grupo.

07/04/22

21:42 Nos reunimos con el profesor y no se encontró el problema, el profesor nos facilitó archivos de ejemplo y nos dio páginas de consulta para comparar nuestro código actual.

23:40 Después de probar varias cosas nos retiramos y lo dejamos por el día.

09/04/22

11:00 Se empieza con la recolección de datos y la escritura de los mismos para dárselos al juego desde el servidor.

23:30 Finalmente se logra hacer la respectiva recolección de datos y se empiezan a interactuar con el juego.

10/04/22

11:00 Se ajustan detalles finales del juego.

14:30 Se da por concluida la participación en el trabajo.