**Universidad Nacional del Nordeste**

**Facultad de Ciencias Exactas y Naturales y Agrimensura**

**Licenciatura en Sistemas de Información**

Informe Final: Trabajo Integrador

**Asignatura:** Algoritmos y Estructuras de Datos II

**Profesores:** Espindola, Cecilia. Badaracco, Numa. Sembrana, Iván.

**Comisión:** Grupo A (Siesta)

**Integrantes:**

Cardozo, Carola Guillermina

Celi Seminario, Federico Ezequiel

Conti, Tomás Ariel

Escobar, Sergio David

**Fecha de entrega final:** 20/11/2022.

**Parte Inicial:**

1. Nombre del proyecto

El nombre del proyecto es: Space Crash.

2. Participantes

Cardozo, Carola Guillermina

Celi Seminario, Federico Ezequiel

Conti, Tomás Ariel

Escobar, Sergio David

3. Descripción de proyecto (software a desarrollar)

Realizaremos un videojuego hecho en C + + este consiste en una nave que se encuentra en el espacio y debe esquivar obstáculos hasta que se terminen las 3 vidas, si se terminan las vidas, termina el programa imprimiendo un mensaje que diga GAME OVER. El usuario obtendrá puntos cada vez que avance en el juego y siga con vida. Cuenta con un menú y una cuenta regresiva antes de comenzar el juego. El juego se encuentra en un archivo binario el cual tiene una estructura de árbol. Una vez que el usuario seleccione que desea jugar se le preguntará si es mujer o hombre (esto nada más se diseñó para poder realizar con corte de control en el archivo de lectura), esa pregunta fue necesaria para realizar el corte.

4. Justificación del proyecto

La elección de este proyecto es porque consideramos un desafío desarrollarlo y además lo vemos como algo creativo y divertido de realizarlo. También nuestra elección de realizar el proyecto se debe a que vimos la posibilidad de integrar los distintos temas que fuimos desarrollando en el cuatrimestre.

5. Planificación del Proyecto (Relacionar con el ciclo de vida seleccionado).

Etapas en la resolución de problemas: el ciclo de vida del software. Desde el planteo inicial de un problema hasta que se obtiene el correspondiente programa o aplicación y su instalación y funcionamiento en una computadora, se siguen una serie de pasos que en conjunto constituyen lo que en Ingeniería de Software se denomina ciclo de vida del software. Si bien se reconoce que la creación de programas es un proceso esencialmente creativo, los pasos o etapas que generalmente siguen los programadores son:

Análisis:

Estudio detallado del problema con el fin de obtener los requerimientos que darán inicio al proceso de automatización.

El análisis que se realizó para la creación de Space crash fue: se pensó en todas las funciones que necesitamos para que el juego funcione de manera correcta y que en dichas funciones implementemos lo dado en el cuatrimestre. Por lo tanto, las funciones que utilizamos fueron, la implementación de un archivo binario de escritura, decidimos que posea la estructura de un árbol, porque nos facilita la inserción de los usuarios (puntaje) y nos ahorramos tener que ordenarlo, ya que la estructura del árbol lo ordena. En el archivo se van a ejecutar todas las funciones que fueron convertidas en TAD, como: el juego, una presentación del juego, dos menús, uno para ingresar el género y nombre, el segundo es para que el usuario seleccione si desea jugar y un TAD que contiene una cuenta regresiva que se presenta en la pantalla una vez que el usuario selecciona la opción de jugar.

El objetivo esperado del programa es que cada vez que juegue un usuario, se registre y se guarde en el archivo sus datos, además de que mientras el usuario tenga vidas cuente un puntaje, y ese mismo se inserte en el árbol, para que así obtengamos ordenados los puntajes. Una vez que termine de jugar el usuario, se espera que se registre otro jugador y se produzca el mismo procedimiento. Estos registros se podrán leer en el archivo de lectura, ya que, aquí se producirá el corte de control, este será por el género del jugador. Además, en el archivo de lectura se mostrarán los datos de los jugadores y sus puntajes.

Diseño:

El algoritmo que se pensó fue el siguiente: primero se pensó el funcionamiento de la nave, su movilidad, el aspecto de la nave, el espacio y el límite, las vidas y el puntaje, dónde se van a ejecutar todas las funciones que implementemos, además de dónde y cómo se van a guardar los registros de cada jugador, esta consistía en:

1. La nave fue creada con caracteres del código ASCII, para imaginar cómo se vería la nave lo dibujamos a papel entonces, tendríamos una idea de cómo podríamos ubicar los caracteres y así también sabremos cómo se podrían ver en la pantalla.
2. El espacio y el límite lo hicimos con vectores anidados, es decir, en una matriz, definimos un límite de 78 x 33, entonces iniciamos el primer for desde 2 hasta 78 y pintaba con caracteres el límite, el segundo for lo iniciamos desde 4 hasta 33 y pintaba el límite con los caracteres asignados.
3. Para las vidas creamos una variable de tipo entero a la cual le asignamos un valor de 3, y este funcionaba de tal manera que si a la nave le chocan 3 obstáculos la vida se restaba de uno en uno.
4. La movilidad de la nave la realizamos con funciones específicas de librerías, incorporando la función gotoxy, entonces primero asignamos las teclas con las que el usuario se podía mover y por las cuales el programa lo entendería, entonces asignamos las flechas de las izquierda, derecha, arriba y abajo, el programa lo entendería como el eje cartesiano, es decir, que si el usuario seleccionaba la tecla izquierda se movería al lado del eje x que se encuentra en negativo, y si iba para la derecha se iría por el eje de las x que es positivo, lo mismo pasaría cuando va hacia arriba o hacia abajo por el eje de la y.
5. Todas las funciones que se encuentran en los TAD se van a ejecutar en el archivo binario de escritura que posee la estructura de un árbol. Decidimos realizar con la estructura de un árbol ya que nos facilitaba el ordenamiento de los datos, en este caso el puntaje de cada jugador.
6. El corte de control lo realizamos en un archivo binario de lectura donde el corte se produce por un campo tipo entero de género, para luego poder sacar porcentaje del total de jugadores dependiendo del género que se indicó, además en el archivo de lectura se van a mostrar los puntajes con el nombre de cada jugador.

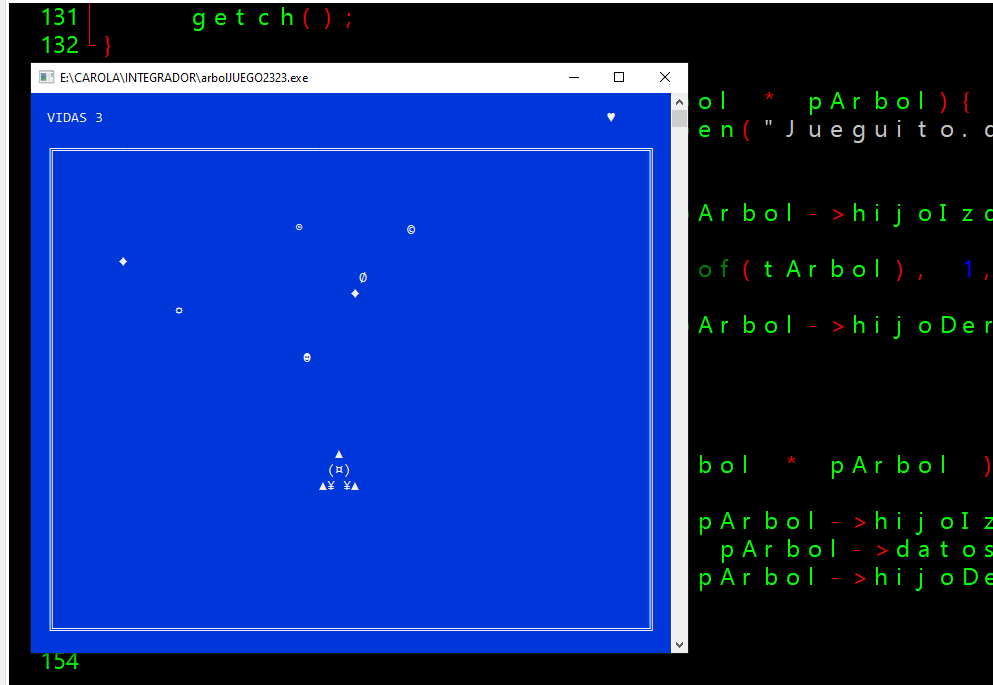
Codificación:

Transcripción del algoritmo a instrucciones para la computadora escritas respetando la sintaxis de un lenguaje de programación, obteniéndose así el programa o código fuente. La transcripción del algoritmo pensado lo hicimos en c + +.

Compilación, ejecución, verificación y depuración:

Este fue uno de los pasos que más trabajo nos llevó realizar debido a que cuando juntamos todos los TAD se presentaron varios problemas con las librerías, entonces, lo que hicimos fue quitar las librerías de todos los TAD y tan solo dejamos uno solo para que se evite el error. Además, tuvimos muchos problemas con la función gotoxy ya que la mayoría de los TAD la utilizaba, y también se produjo una redefinición de variables porque muchas variables tenían el mismo nombre, entonces tuvimos que redefinir muchas variables para eliminar los errores.

Imagen de como se ve el juego al compilar:

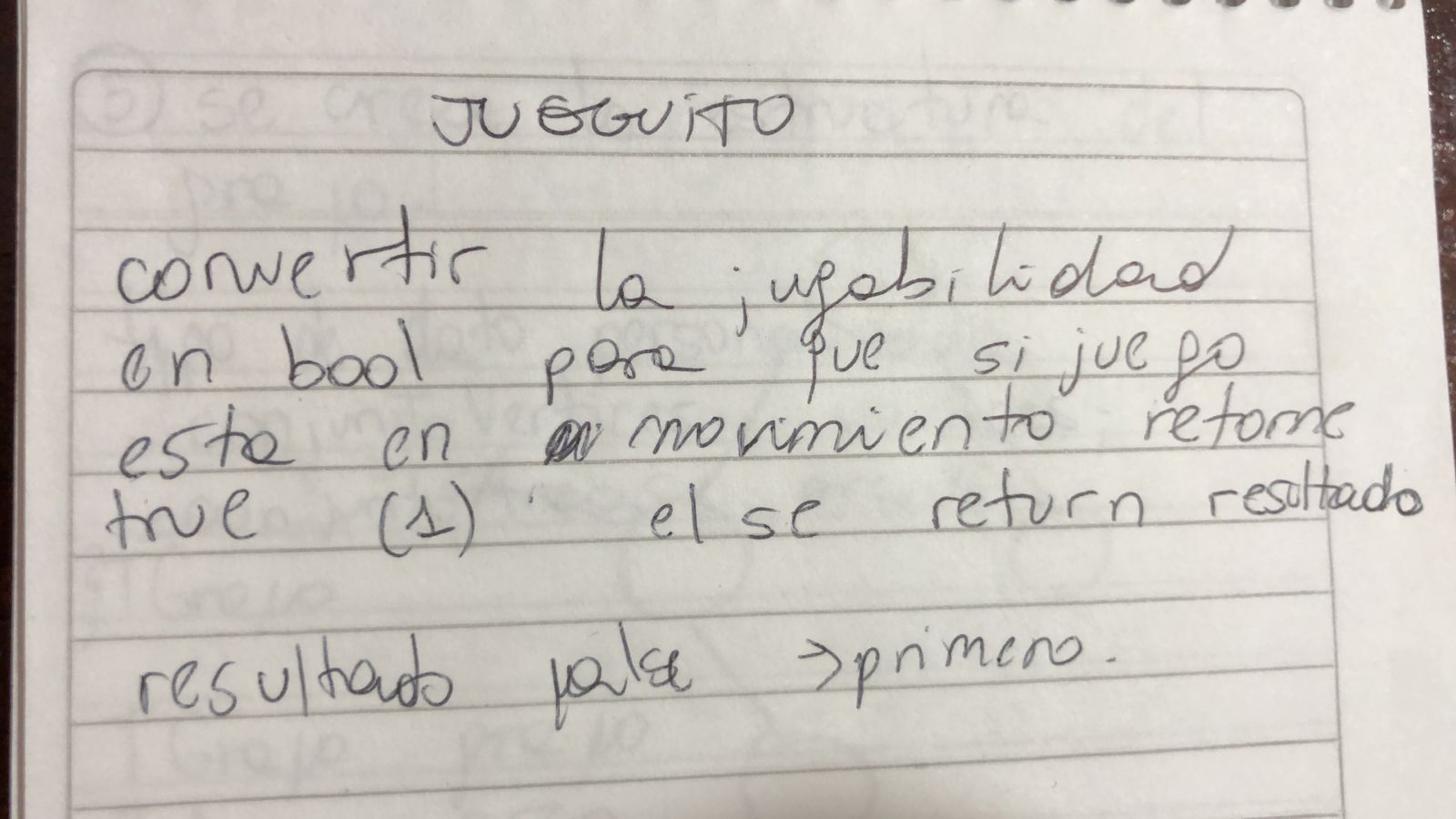


Mantenimiento (o Evolución):

Actualización del programa en función de los requerimientos de los usuarios, o para mejorar el rendimiento. Esta es la etapa más larga del ciclo de vida de desarrollo de software, y puede durar muchos años. En la actualidad se considera que el mantenimiento es realmente un desarrollo evolutivo.

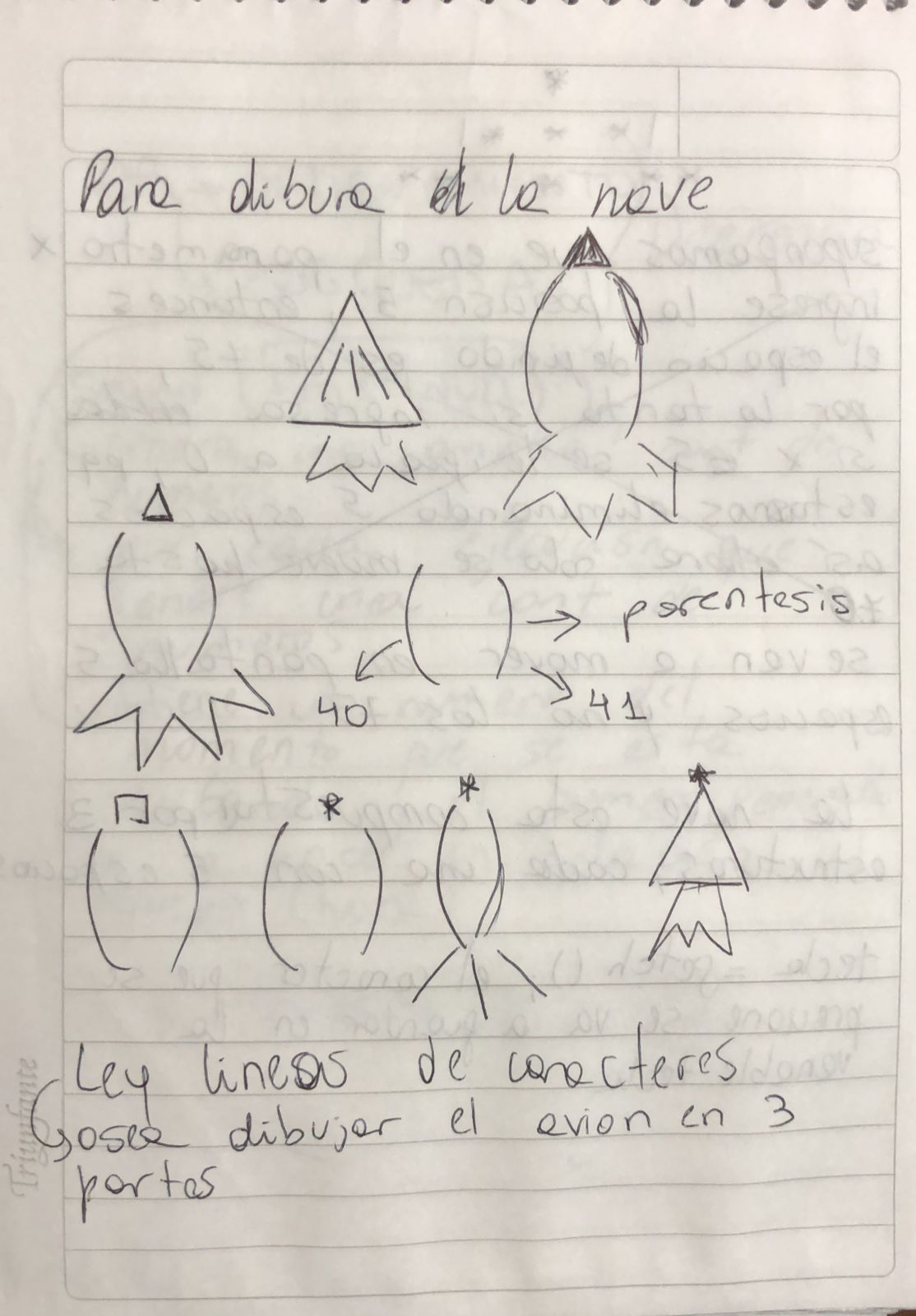
Documentación:

Esta fue una de las posibilidades que se pensó en el algoritmo de la jugabilidad en relación al puntaje, se convirtió la función de la jugabilidad en BOOL para que cuando vida sea mayor a 0 retorne 1 si no retorne 0, para así, en el archivo exista una condición que si jugabilidad==1, entonces cuente el puntaje del jugador, y si jugabilidad==0, que deje de contar.

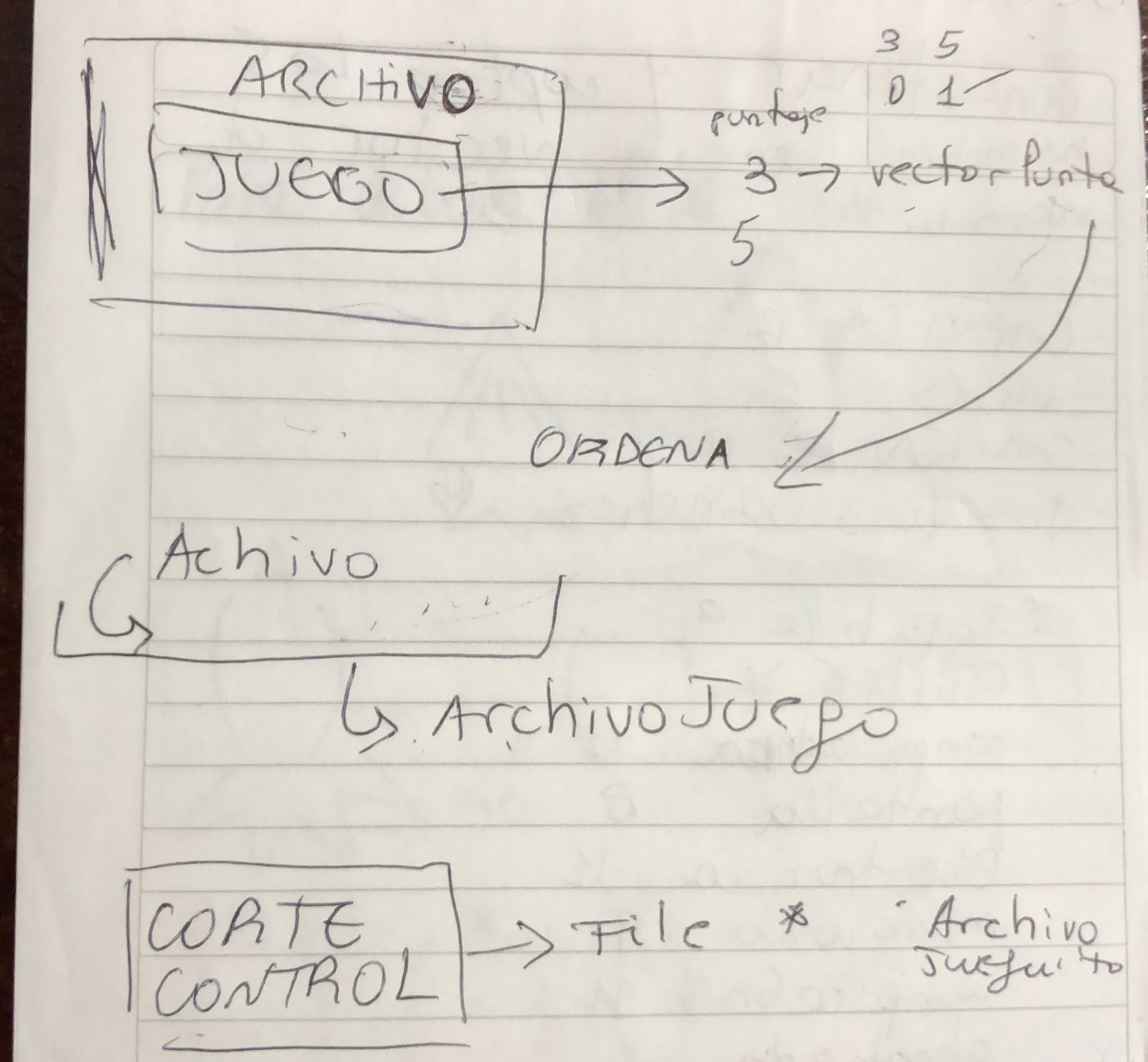


Luego se cambió la función por un INT para que mientras el usuario esté jugando cuente una puntuación, entonces, retorne un entero a la función principal, que es en el archivo de escritura entonces se le pasa el valor de la puntuación a la variable puntaje de datos del jugador, así le asignamos el resultado y lo pasamos al árbol y luego al registro.

Esta es una de las tantas estructuras que se pensó para pintar la nave con caracteres, lo dibujamos a papel para imaginarnos.



Este fue el primer algoritmo que se pensó, consistía en cargar los datos en un vector que iban almacenando en un archivo binario, de ahí ordenarlos aplicando la función de ordenamiento, entonces de ahí pasarlo a una lista y de la lista al archivo de lectura.



Manuales para jugar:

El juego Space Crash consiste en una nave que se encuentra en el espacio, la misma, debe esquivar obstáculos (caracteres) que simulan ser asteroides, el usuario se debe mover presionando las teclas de las flechas: derecha, izquierda, arriba y abajo, con ellas se moverá la nave y logrará esquivar los obstáculos obteniendo un puntaje mayor.

ACLARACIONES: El juego puede presentar algunas complicaciones con los gráficos, es decir, que puede suceder que la nave se salga del plano del juego, si eso sucede presione la flecha para ir hacia arriba tantas veces sea necesaria y volverá al plano del límite establecido.

**Parte Final:**

6. Detalle de cada etapa

a. Descripción de las actividades

Etapa 1: Introducción al proyecto.

En esta etapa nos pusimos de acuerdo sobre el proyecto a realizar, pensar las diferentes estrategias y que tema dado se podría utilizar para la realización de las diferentes partes del juego.

Etapa 2: Estructuras compuestas: listas (implementación estática mediante arrays)

No se pensó utilizar ningún tipo de implementación estática, debido a que no era conveniente para lo que queríamos realizar en nuestro proyecto.

Etapa 3: Estructuras compuestas enlazadas (implementación dinámica mediante listas enlazadas)

Estructuras compuestas (listas) al principio se utilizó, se armó un algoritmo para la implementación de la lista, esta consistía en un archivo que poseía la estructura de una lista enlazada, en este se iban a ejecutar todas las funciones del juego, además queríamos guardar los datos del usuario en una lista, para esto, decidimos que los datos de cada usuario se guardan primero en un vector para luego ordenarlos y pasarlo a la lista, porque si los datos se guardan ordenados tan solo quedaba pasarlo a la lista para luego poder realizar el la lectura del archivo y así también ya no tendríamos dificultad al realizar el corte de control ya que los datos iban a estar ordenados.

Se pensó en la posibilidad de utilizar las listas enlazadas con punteros pero no lo implementamos a este tema porque nos fue más conveniente utilizar árboles. Porque el árbol ya guarda los datos de forma ordenada a diferencia de las listas enlazadas.

Etapa 4: Corte de control.

El corte de control se va a realizar en un archivo de lectura el cual posee la estructura de un árbol. Para aplicar corte de control decidimos agregar un campo en el registro de datos, este campo será para indicar el género del usuario, registrará datos del tipo entero "opción: 1) si es mujer" o 2) si es hombre.

Luego el corte de control tomará ese campo (género) como clave para recorrer y así, contar cuantos jugadores de cada género jugaron y realizar un porcentaje con la cantidad de jugadores totales y la cantidad de jugadores dependiendo el género, además en el archivo de lectura se recorrerá el árbol y se mostrarán cada uno de los nodos.

Etapa 5: Implementación de Tipo Abstracto de Datos (TAD).

En el proyecto decidimos implementar TAD primero para poder distribuirnos las tareas y segundo porque mejoraba la modularización de cada una de las funciones, luego las incluimos en el archivo de escritura que es donde se ejecutan todas las funciones del juego. Los TAD que implementamos son: un TAD que posee una cuenta regresiva, la jugabilidad y la nave, un menú con el que el usuario indica si desea jugar una ronda, y un último menú que presenta el juego.

Etapa 6: Implementación de búsqueda.

La implementación de búsqueda no fue necesaria en nuestro proyecto, ya que no necesitamos localizar ningún dato, además que para la implementación de búsqueda necesitábamos vectores, en nuestro proyecto no utilizamos vectores para almacenar datos, solo usamos vectores para pintar los límites de la pantalla con caracteres.

Etapa 7: Implementación de ordenamiento.

Podría considerarse como ordenamiento a la estructura misma del árbol que al insertar un elemento al árbol ya lo ordena, por ello fue que decidimos implementarla ya que, por dicha estructura una vez que se van almacenando los datos ya los guarda ordenados, a la derecha guarda los datos mayores al nodo padre y a la izquierda guarda los datos menores al nodo padre.

Etapa 8: Recursividad:

Implementamos recursividad en un TAD donde decidimos realizar una cuenta regresiva antes que comience el juego, es decir, luego de seleccionar la opción de iniciar juego del menú, se ejecuta la recursividad hasta que el caso base sea igual a 0 y ahí se inicia el juego. La cuenta regresiva la representamos mediante caracteres de manera que se vea 5, 4, 3, 2, 1. Cada número desde el 5 al 1 fue diseñado en diferentes funciones, por medio del carácter asterisco \* y usando printf fuimos armando los números en grande. En una función llamada "cuentaRegresiva" se pasa como parámetro el valor 5 y con un switch y teniendo en cuenta el valor va comparando los diferentes casos, en cada iteración dónde la función se llama así misma retorna su parámetro valor -1, entonces:

* Si caso 5 es igual al parámetro valor 5, llama a la función dónde está diseñado el número 5 con caracteres.
* Si caso 4 es igual al parámetro valor 4 (al llamarse a sí mismo al parámetro que era 5 se le resto 1), llama a la función dónde está diseñado el número 4 con caracteres.
* Si caso 3 es igual al parámetro valor 3 (al llamarse a sí mismo al parámetro que era 4 se le resto 1), llama a la función dónde está diseñado el número 3 con caracteres.
* Si caso 2 es igual al parámetro valor 2 (al llamarse a sí mismo al parámetro que era 3 se le resto 1), llama a la función dónde está diseñado el número 2 con caracteres.
* Si caso 1 es igual al parámetro valor 1 (al llamarse a sí mismo al parámetro que era 2 se le resto 1), llama a la función dónde está diseñado el número 1 con caracteres.
* Si caso 0 es igual al parámetro valor 0 (al llamarse a sí mismo al parámetro que era 1 se le resto 1), se cumple el caso base y entonces termina la recursividad.

Etapa 9: Implementación de árboles.

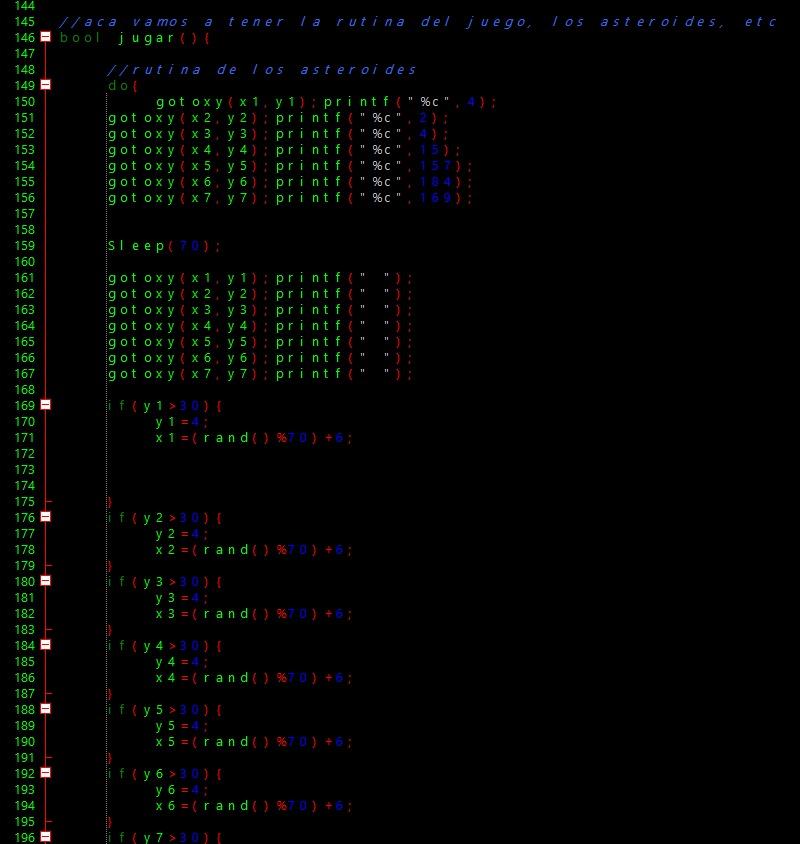
Decidimos implementar árboles de la siguiente manera: creamos un archivo binario que posee la estructura de un árbol, en ella se va a guardar como registro los datos de cada jugador, la elección de una estructura tipo árbol se dio porque el árbol utiliza un método de inserción de elementos que los ordena de mayor a menor, es decir, de lado izquierdo del nodo se guardan los datos menores al nodo padre, y del lado derecho se guardan los datos mayores al nodo padre, como nuestro objetivo era utilizar el árbol y ordenar los puntajes cada que ingrese un nuevo jugador con un puntaje diferente, se almacenará en el árbol de forma ordenada, permitiendo que cuando pasemos los datos al archivo de lectura ya se encuentren ordenados, además necesitábamos para poder realizar el corte ya que los datos deben estar ordenados.

Etapa 10: Implementación de grafos.

No implementamos en el juego grafos exclusivamente porque al utilizar árboles binarios para guardar los datos del usuario en el archivo binario de escritura ya estamos haciendo un tipo de grafo. En [teoría de grafos](https://es.wikipedia.org/wiki/Teor%C3%ADa_de_grafos), se usa la siguiente definición: «Un árbol binario es un [grafo conexo](https://es.wikipedia.org/wiki/Grafo_conexo), [acíclico](https://es.wikipedia.org/wiki/Grafo_ac%C3%ADclico) y [no dirigido](https://es.wikipedia.org/wiki/Grafo_no_dirigido) tal que el [grado](https://es.wikipedia.org/wiki/Grado_(teor%C3%ADa_de_grafos)) de cada [vértice](https://es.wikipedia.org/wiki/V%C3%A9rtice_(teor%C3%ADa_de_grafos)) no es mayor a 2». De esta forma solo existe un camino entre un par de nodos.

b. Código (en lenguaje “C”) de los módulos realizados:

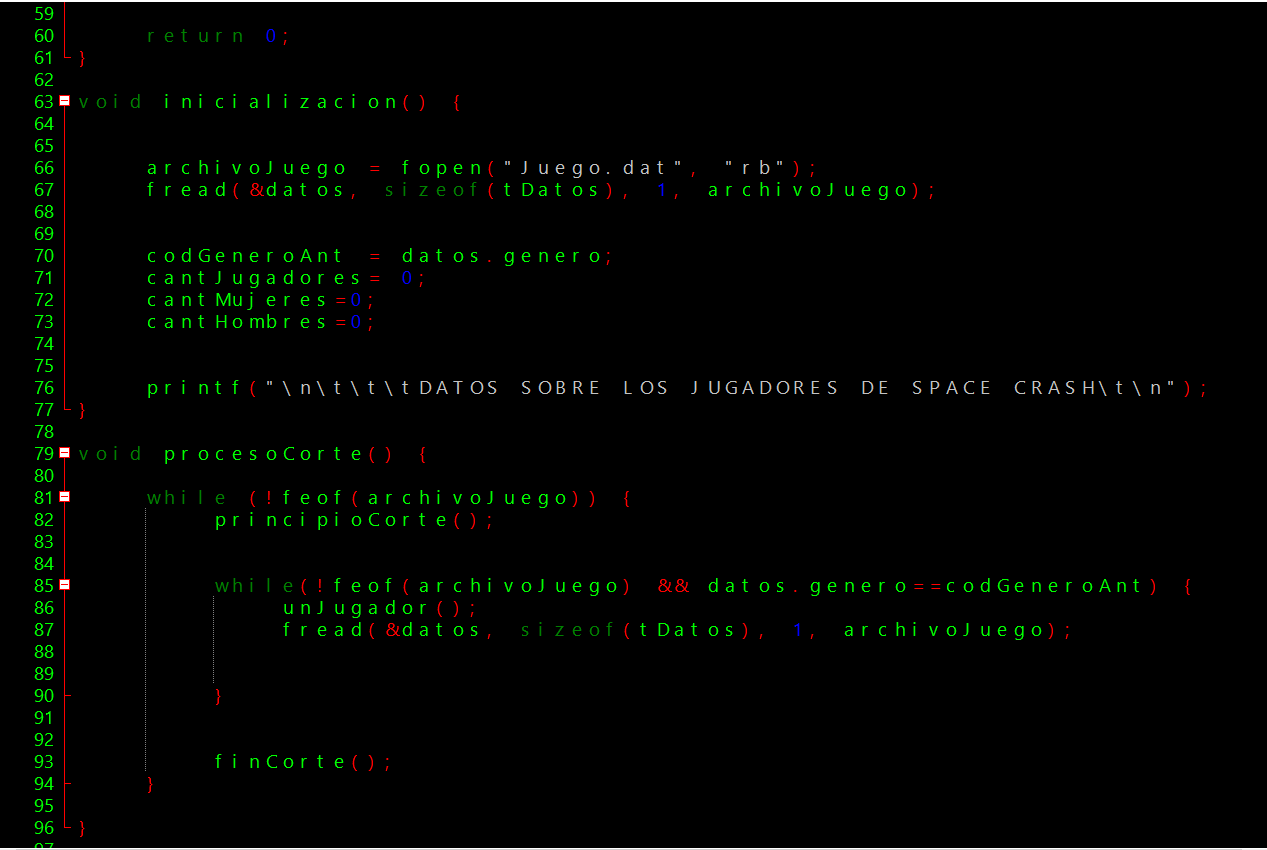
* A continuación, está es una de las partes de la jugabilidad, del código del juego y el de la nave.



* A continuación, está la parte de la del código de la cuenta regresiva: 
* A continuación, está la parte del código del menú para presentar el juego:

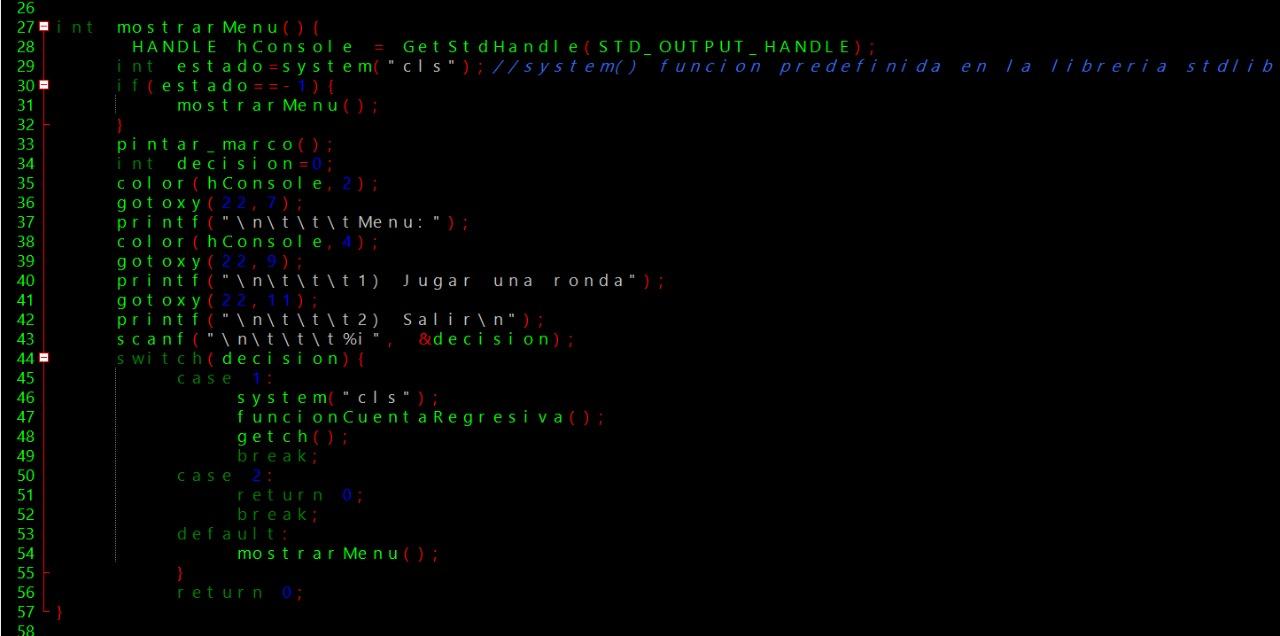


* A continuación está la parte del código del corte de control:



* A continuación, está la parte del código del archivo con estructura de árbol:



* A continuación, está la parte del código menú de iniciar el juego:

c. Herramientas utilizadas

Utilizamos para codificar el proyecto ide Dev C + +. La aplicación en la nube de Google Documents para hacer el informe. Utilizamos Whatsapp para comunicarnos mediante chat y llamada. Y la aplicación de Discord para codificar en grupo.

<https://elcodigoascii.com.ar/>

<https://www.youtube.com/watch?v=qgFOgcsG-M8&list=LL&index=17>

<https://www.youtube.com/watch?v=ptp8VUHBZ6Y&list=PL733F3A4B21EFCF77&index=12>

<https://www.youtube.com/watch?v=73dlfeXSoTE>

<https://www.youtube.com/results?search_query=agregar+colores+en+c%2B%2B>

<https://www.youtube.com/watch?v=N-3mf0VCnhw>

<https://www.youtube.com/watch?v=DSc22NqARk8&t=44s>

<https://www.youtube.com/watch?v=Vrjg8F4U22Y>

<https://www.youtube.com/watch?v=H2BZqEPVKYE>

<https://www.youtube.com/watch?v=WuEQW1j_lAk>

7. Conclusiones

En conclusión, este proyecto de la materia Algoritmos y Estructuras de Datos II de nuestra elaboración del videojuego Space Crash en C++ nos permitió aplicar los distintos temas que fuimos dando en la materia como corte de control, TAD, ordenamiento, árboles y también aprender a trabajar en equipo, sobre todo en este nuevo ámbito que es la programación, arreglar nuestras diferencias, aprender a salir adelante cuando no todo sale como esperamos. Además, nos permitió conocer el lenguaje C que es la base de la mayoría de los lenguajes de programación y C++ que es la evolución de C. Por otra parte, si no hubiéramos desarrollado este proyecto no habríamos adquirido la experiencia y la posibilidad de aplicar lo que damos en la materia en algo concreto y real como lo es el videojuego que hicimos. También consideramos que la realización de este proyecto fue un gran acercamiento a lo que sería trabajar como programadores, desarrollamos una gran habilidad lógica a la hora de realizarlo y también para poder solucionar los errores que se nos presentó en el camino a su finalización.

**Evaluación, revisión y reflexión:**

Momento de Reflexión:

1) ¿Qué aprendieron de nuevo? ¿Qué fue lo que más les gustó de estas actividades?

Al realizar este proyecto lo que aprendimos de nuevo fue como realizar un videojuego en C + + del que antes no teníamos los conocimientos para realizarlo. Además logramos relacionar todos los conceptos que aprendimos durante el 2do cuatrimestre para la realización del proyecto. También aprendimos a trabajar en equipo, consideramos que esta experiencia es nuestro primer acercamiento a cómo trabajar en equipo como programadores, la cual fue muy divertida y creativa.

Lo que más nos gustó realizar:

Carola: Lo que más me gusto realizar de las actividades fue pensar en el algoritmo del juego, ya que, podíamos imaginar muchas posibilidades implementando los temas de la asignatura para crear el juego. También lo que me gusto fue pensar en cómo podría verse la nave ya que usamos diferentes tipos de caracteres para que tome la forma más parecida a una nave. Otra de las cosas que realizamos en grupo que me gusto fue cuando debíamos resolver problemas y pensamos todos juntos probando diferentes soluciones para que funcione el código.

Sergio: Lo que me gustó fue ir pensando que se puede implementar y cómo, analizar los algoritmos con diferentes posibilidades para hacer que la computadora lo ejecute, también trabajar con TAD ya que hace que nos dividamos las partes y avancemos más rápido.

Tomás: Lo que aprendí de nuevo es sobre el tema de tad que permite dividir en archivos las resoluciones de problemas pequeños y unir las mismas para resolver un problema más grande utilizando un archivo de cabecera. También aprendí sobre el tema de la recursividad que permite por ejemplo hacer una cuenta regresiva dentro del juego que hicimos. Lo que más me gustó fue ver cómo al unir diferentes archivos, se logra resolver el problema grande y de esa manera permite dividir las tareas en el grupo resolviendo cada uno una parte del problema, a pesar de que siempre hay dificultades a la hora de que funcione correctamente.

Federico: Lo que aprendí fue que en C + + podemos realizar varias cosas diferentes a las que hacíamos en la práctica como programas de juegos, aprendí a modularizar grandes códigos en varias partes, solucionar problemas pensando en una estrategia, trabajar en equipo. También lo que más me gustó fue el tema de la recursividad ya que aprendí a rehusar la misma función llamándose a sí misma.

2) ¿En qué piensan que tienen que mejorar? ¿Qué pueden hacer para mejorar?

Con el grupo consideramos que debemos mejorar en muchas cosas en general, debido a que este es nuestro primer año, asimismo manejamos bien los conceptos dados en el cuatrimestre, y que para la complejidad de nuestro proyecto lo hicimos muy bien e implementamos los conceptos de manera correcta. Lo que podemos hacer para mejorar es seguir adquiriendo conocimiento y lograr relacionarlo constantemente, además de seguir practicando.

3) ¿Cuál/cuáles de la tarea/s le demandó mayor esfuerzo o desafío para desarrollarla?

La tarea que más nos demandó esfuerzo fue la de implementar TAD en el código debido a que a la hora de juntar ciertos códigos TAD .h en un solo código .c, tuvimos complicaciones al compilar porque algunas librerías y ciertas funciones específicas de las librerías no eran compatibles en archivos con extensión c. Otra de las actividades que más nos dificultó realizar fue la parte del archivo que tiene una estructura de árbol, debido a que a la hora de contar el puntaje de cada jugador fue un poco complejo.

4) ¿Crees que lo aprendido lo pueden aplicar en otro contexto? ¿Cuál por ejemplo? y ¿Cómo?

Si, consideramos que todo lo aprendido podemos implementar en muchos algoritmos, por ejemplo, podríamos crear otros juegos con la misma complejidad del realizado en el proyecto, además todos los temas dados en el cuatrimestre nos brindaron una gran amplitud de posibilidades donde podemos implementar estos conceptos. Aunque en las actividades prácticas de la asignatura nos daban problemas a resolver respecto de cantidades de memoria muy pequeña, aun así estos temas nos han brindado herramientas para imaginar diferentes algoritmos como videojuegos, páginas donde el usuario se puede contactar con un correo y al insertar el correo se guarde con alguna estructura de pila, o también se pensó en algún algoritmo de páginas de productos donde la clasificación del mismo producto pero con diferentes precios se puede hacer con un árbol, ya que los de menor precio se van del lado izquierdo y los de mayor precio del lado derecho, es decir, que si el usuario desea saber un tipo de producto pero los de menor precio, se podría mostrar siguiendo la estructura de un árbol.