DOCUMENTACION EXTERNA

Daniel Brenes Martínez

INSTITUTO TECNOLOGICO DE COSTA RICA

Introducción.

En el siguiente trabajo se presenta un juego que se asemeja a Space invaders, el mismo implementa listas para guardar datos, utilizando de esta manera lo aprendido en clase. Se implementa el paradigma orientado a objetos en el lenguaje Java para desarrollar el juego, así como el uso de patrones de diseño, para hacer el código más sencillo de entender. Para controlar el juego se hace uso del teclado o una aplicación sencilla en el teléfono.

Se presentan las historias de usuario y features respectivas, los diagramas, de componentes, arquitectura, secuencia y clases. De esta manera se tiene la documentación del código, junto con el java doc.

Breve descripción del problema.

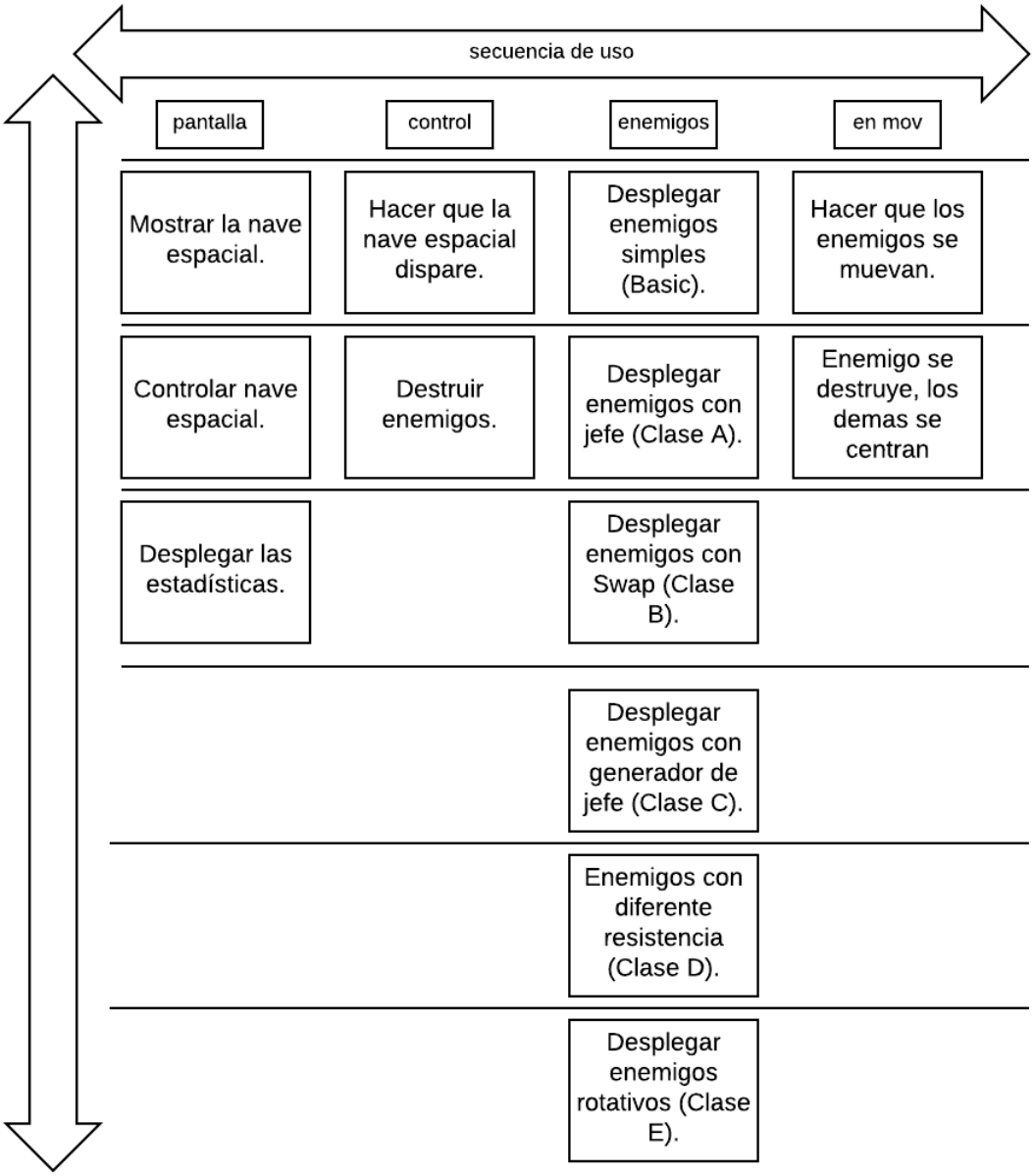
El juego debe implementar la misma lógica que el original space invaders, pero este debe tener otras características conforme a los enemigos, ya que dependiendo del nivel estos difieren a la hora de desplegarse en pantalla, se debe crear una nave espacial que dispare misiles y que estos destruyan los enemigos.

Planificación y administración del proyecto.

- Lista de features e historias de usuario.
    - Features:
        - Nave espacial: El jugador controla la nave espacial y la mueve de izquierda a derecha en la parte inferior de la pantalla, utiliza misiles para destruir a los enemigos.
        - Enemigos: Aparecen en hileras en la pantalla, inicialmente todos los enemigos son simples, conforme se avanza en el juego, salen enemigos con resistencias diferentes.
        - Jefes: Los Jefes tienen las mismas características que los enemigos, pero son más resistentes a los disparos, por lo tanto, más difíciles de eliminar.
        - Estadísticas en pantalla: Se muestran las estadísticas del juego en la pantalla y estos van cambiando conforme se avanza el juego.
        - Controlador del juego: El jugador puede controlar el juego con el teclado o con una aplicación de teléfono sencilla.
    - Historias de usuario:
        - Mostrar la nave espacial.
        - Hacer que la nave espacial dispare.
        - Desplegar enemigos simples (Basic).
        - Hacer que los enemigos se muevan.
        - Controlar nave espacial.
        - Destruir enemigos.

- Desplegar enemigos con jefe (Clase A).
- Hacer que cuando un enemigo se destruye los que queden se muevan hacia el centro.
- Desplegar las estadísticas.
- Desplegar enemigos con Swap (Clase B).
- Desplegar enemigos con generador de jefe (Clase C).
- Desplegar enemigos con diferente resistencia (Clase D).
- Desplegar enemigos rotativos (Clase E).

- Distribución de historias de usuario.

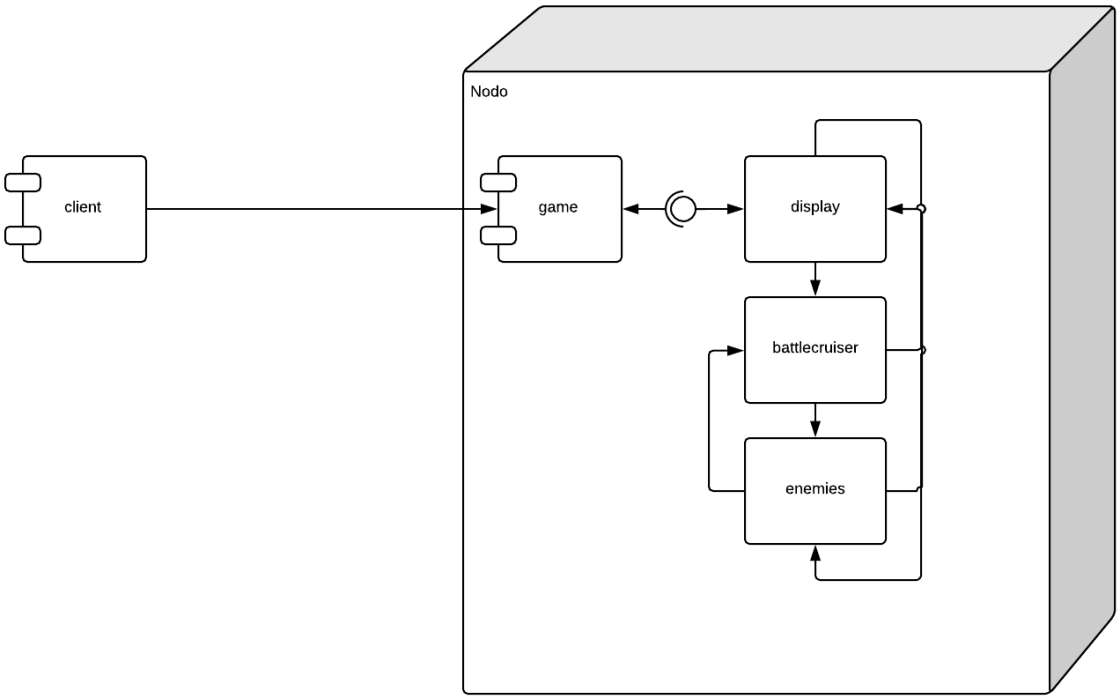


- Minimal System Span.
    - La nave espacial debe estar siempre visible.
    - La nave debe de disparar siempre.
    - Los enemigos deben ser visibles en la pantalla.
    - Los enemigos deben moverse.
- Plan de Iteraciones.
    - Capa de presentación: Se crea una interfaz que muestre una pantalla inicial, antes de empezar a jugar.
        - Mostrar la nave espacial: Se muestra la imagen de la nave espacial, e la pantalla.
        - Desplegar enemigos Simples (Basic): Se despliegan los enemigos simples en pantalla.
        - Desplegar enemigos con jefe (Clase A): Se muestran en pantalla los enemigos, pero esta vez se incluye un jefe.
        - Desplegar las estadísticas: Se muestran las estadísticas en pantalla, las cuales cambian conforme se juega.
    - Capa de lógica de juego: Se trabaja el comportamiento más detallado de los enemigos y el jugador.
        - Hacer que la nave espacial dispare: Se muestra en pantalla un misil que tenga una trayectoria recta en la pantalla.
        - Hacer que los enemigos se muevan: Los enemigos se mueven de izquierda a derecha y abajo en la pantalla.
        - Controlar la nave espacial: Mediante el teclado se puede controlar el movimiento de la nave espacial.
        - Destruir los enemigos: Al colisionar el misil con los enemigos, estos deben destruirse.
        - Hacer que cuando un enemigo se destruye los que queden se muevan hacia el centro.
        - Desplegar enemigos con Swap(Clase B): El jefe se cambia rápidamente con el resto de los miembros de la hilera.
        - Desplegar enemigos con generador de jefe (Clase C): Al destruirse el jefe, otra toma su lugar aleatoriamente.
        - Desplegar enemigos con diferente resistencia (Clase D): Los miembros de la hilera de enemigos tienen resistencias diferentes y estos se ordenan de mayor a menor resistencia.
        - Desplegar enemigos rotativos (Clase E): La hilera de enemigos rota conforme a las manecillas del reloj y el jefe está en el centro de la hilera.
- Descomposición de user stories en tares.
    - Mostrar la nave espacial: Implementar una interfaz donde se muestre la nave espacial en el centro de la parte inferior de la pantalla de juego.

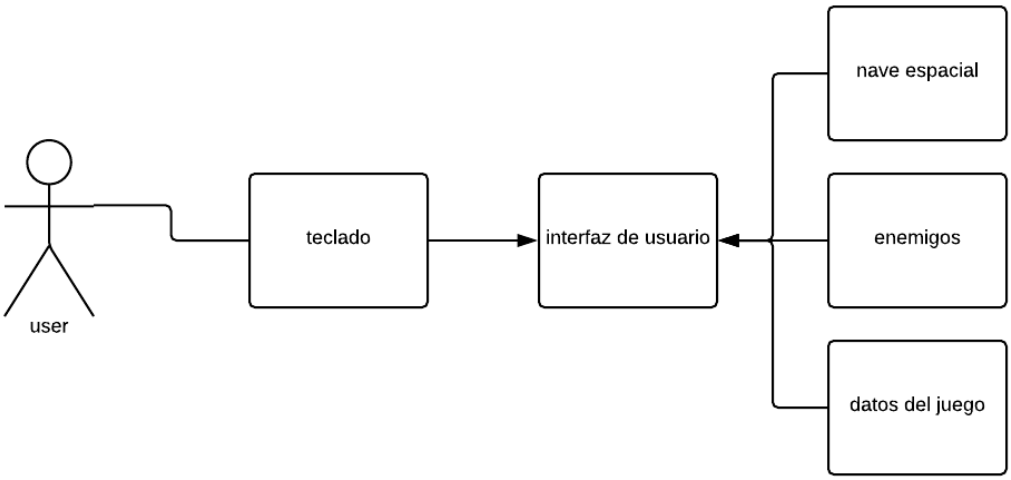
- Hacer que la nave espacial dispare: Implementar una interfaz donde la nave espacial muestre un proyectil en la pantalla.
- Desplegar enemigos simples (Basic): Implementar una interfaz donde se desplieguen los enemigos en la pantalla de juego.
- Hacer que los enemigos se muevan: Crear un método que haga que los enemigos se muevan de izquierda a derecha y abajo.
- Controlar nave espacial: Crear un método para mover las coordenadas en el eje x de la pantalla con el teclado.
- Destruir enemigos: Al colisionar el misil con los enemigos, estos deben desaparecer de la pantalla.
- Desplegar enemigos con jefe (Clase A): Crear un tipo jefe y desplegarlo en pantalla con los enemigos.
- Hacer que cuando un enemigo se destruye los que queden se muevan hacia el centro.
- Desplegar las estadísticas: Crear una interfaz que despliegue los datos del juego en la pantalla y que estos cambien conforme se avanza el juego.
- Desplegar enemigos con Swap (Clase B): Crear una interfaz y método en la lista, donde el jefe cambia de posición con los enemigos.
- Desplegar enemigos con generador de jefe (Clase C): Crear una interfaz donde cada vez que se destruya un jefe, otro se genere en una posición aleatoria.
- Desplegar enemigos con diferente resistencia (Clase D): Crear resistencias en los enemigos, hacer un método bubble sort para que ordene los enemigos conforme a sus diferentes resistencias.
- Desplegar enemigos rotativos (Clase E): Crear un método e interfaz, donde los enemigos roten conforme a las manecillas del reloj.

Diseño.

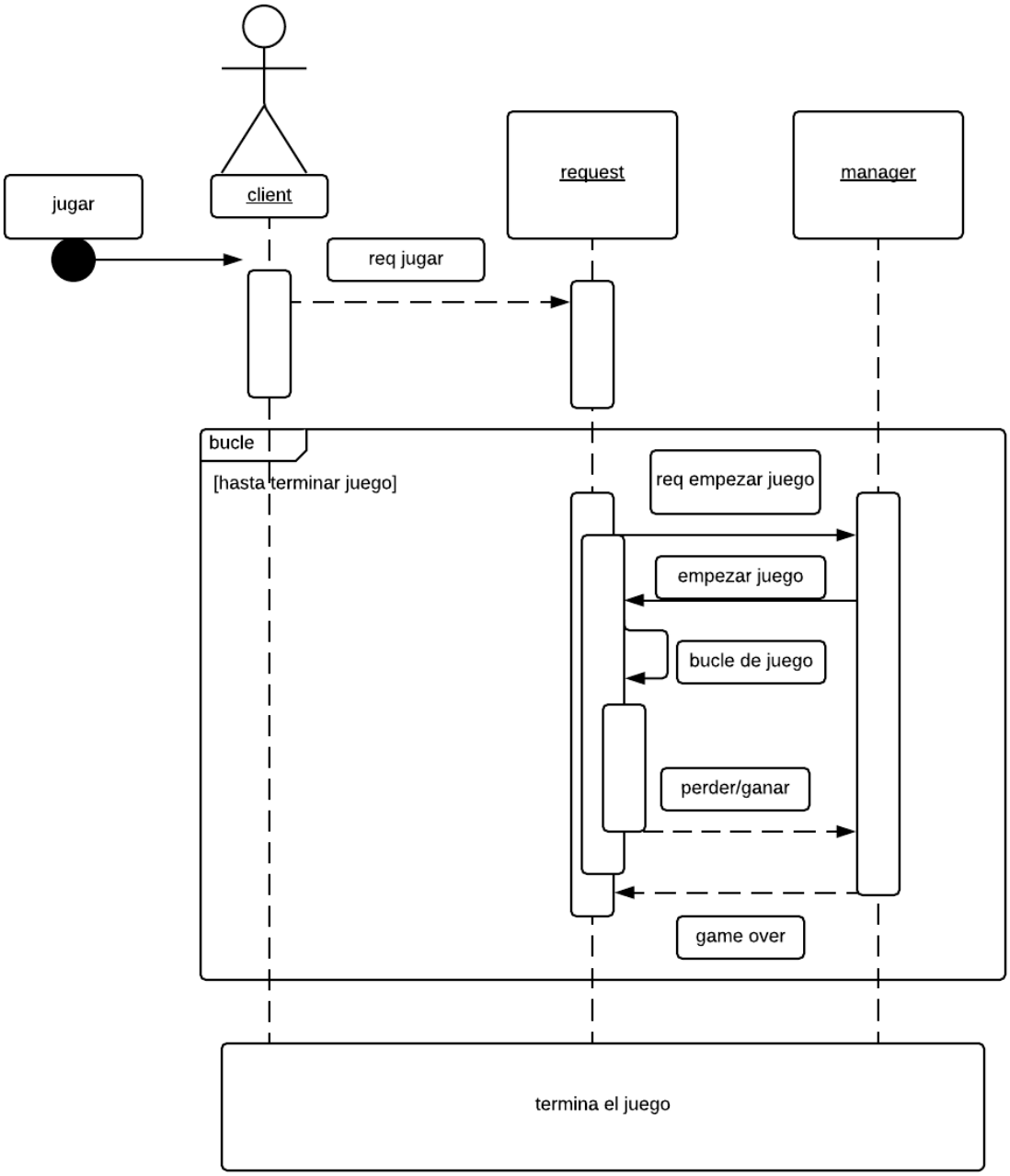
- Diagrama de componentes.



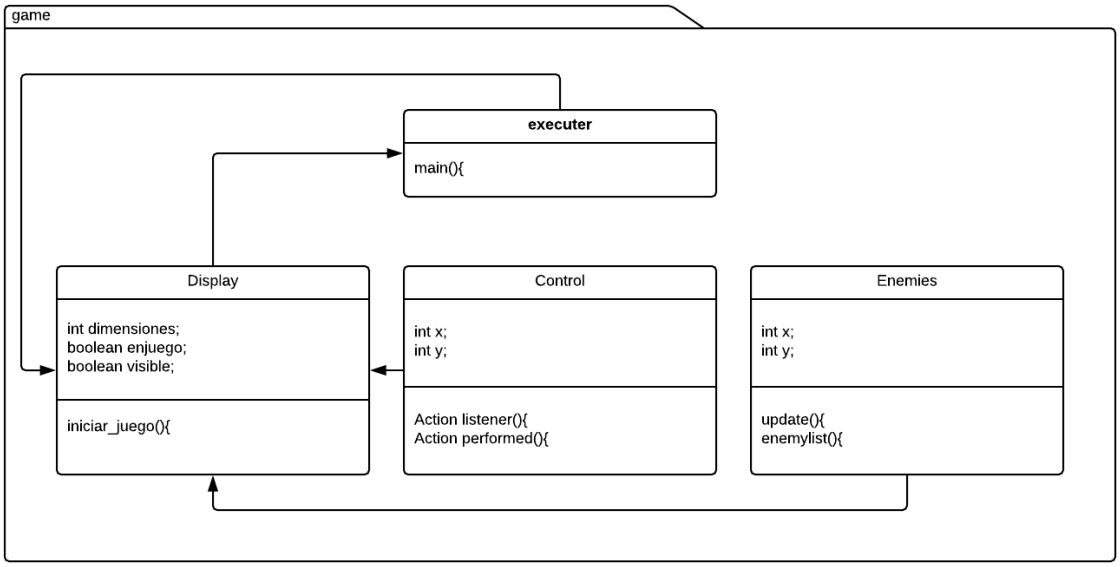
- Diagrama de Arquitectura.



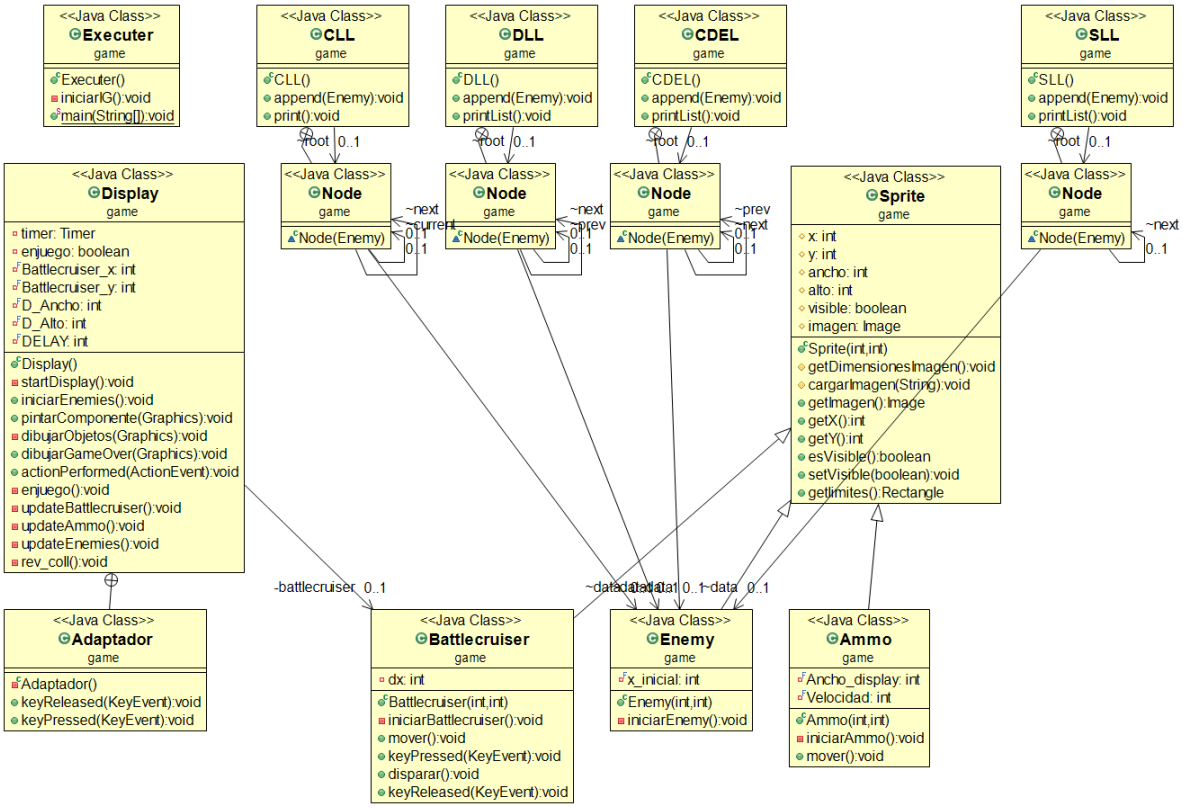
- Diagrama de secuencia



- Diagrama de clases inicial.



- Diagrama de clases final.



Implementación.

- Descripción de las estructuras de datos: Las estructuras de datos utilizadas en este proyecto fueron listas simples, doblemente enlazadas, circulares y circulares

doblemente enlazadas, las listas fueron utilizadas para guardar las instancias creadas sobre las clases de los enemigos, de la misma forma se utilizaron para manejar estos datos a partir de lo que se solicitaba en las instrucciones, por ejemplo los diferentes niveles tenían métodos distintos para manejar los datos (swap, sort, append, insert) los cuales definían la complejidad del nivel. No solo se utilizaron para guardar objetos de la clase enemigos sino también los misiles que usa el jugador para destruir los enemigos, no se utiliza ningún método preconstruido de java para las estructuras de datos, todas son desarrolladas por el programador.

- Descripción de algoritmos desarrollados: Algunos de los algoritmos utilizados son bubblesort, lista simplemente enlazada, lista doblemente enlazada, lista circular, lista circular doblemente enlazada, método eventqueue para ejecutar el juego.
- Problemas encontrados: El problema más complejo fue implementar los diferentes tipos de listas al juego, al entender teóricamente como funciona una lista, llevarlo a la práctica no fue tan difícil, se buscó ayuda de un código en internet para entenderlo mejor y así fue con todas las listas. Otro problema fue aprender en poco tiempo como hacer una interfaz gráfica, se decidió utilizar awt ya que a manera personal y por las experiencias programando en otros lenguajes, me es más sencillo entender la sintaxis de este paquete, a pesar de esto si se invirtieron muchas horas de investigación para aprender a utilizarlo.

Conclusión.

El proyecto fue una buena manera de aprender a manejar estructuras de datos e interfaz gráfica en java, fue una buena manera de aprender a programar en el lenguaje y conocer varios métodos para nuevos proyectos, en general se aprendió bastante no solo del lenguaje y el manejo de estructuras, sino de una manera más estructurada para programar.

Bibliografía.

Jan Bodnar. (2007). Collision Detection. 10/4/2018, de ZetCode Sitio web: http://zetcode.com/tutorials/javagamestutorial/collision/

Se utilizo la página geeks for geeks: https://www.geeksforgeeks.org/ esta no tiene autores ya que es una asociación.

Se utilizaron tutoriales sobre cómo crear listas y sobre cómo utilizar la interfaz gráfica, los links a continuación.

https://www.youtube.com/watch?v=1rquQwfUvHc

https://www.youtube.com/watch?v=coK4jM5wvko&list=PLU8oAlHdN5BktAXdEVCLUYzvDyqRQJ2lk

https://www.youtube.com/watch?v=9q758AJ1nck