TEC | Tecnológico de Costa Rica

Escuela de Computadores

Lenguajes, Compiladores E Intérpretes

Grupo 1

Tarea Programada 3

https://github.com/Devem002/KongCE

DonCEy Kong Jr

Estudiantes:

Diego Vega Mora 2020043916

Fabian Castillo Cerdas 2020202938

Kevin Carranza Blanco 2020163275

Profesor: Marcos Rivera Meneses

Fecha de entrega: 16 de junio del 2023

1° Semestre 2023

Índice

Descripción de la utilización de las estructuras de datos desarrolladas

Listas (List):

- cocodrilos: Es una lista de objetos Cocodrilo, utilizada para almacenar y administrar los cocodrilos presentes en el juego.
- frutas: Es una lista de objetos Fruta, utilizada para almacenar y administrar las frutas presentes en el juego.

Estas listas se utilizan para mantener un seguimiento de los cocodrilos y las frutas en el juego, permitiendo agregar, eliminar y acceder a los elementos de manera eficiente.

Clase Cocodrilo:

Atributos:

- liana: Entero que representa la liana en la que se encuentra el cocodrilo.
- altura: Entero que indica la altura a la que se encuentra el cocodrilo en la liana.
- esRojo: Booleano que indica si el cocodrilo es de color rojo (true) o azul (false).
- Moviéndose: Booleano que indica si el cocodrilo se está moviendo o no.

Esta clase se utiliza para representar un cocodrilo en el juego, con sus respectivas características y estado.

Clase Fruta:

Atributos:

- liana: Entero que representa la liana en la que se encuentra la fruta.
- altura: Entero que indica la altura a la que se encuentra la fruta en la liana.
- puntos: Entero que representa la cantidad de puntos que otorga la fruta.

Esta clase se utiliza para representar una fruta en el juego, con su posición y la cantidad de puntos que otorga al ser recolectada.

Estas estructuras de datos se utilizan para almacenar y gestionar la información relacionada con los cocodrilos y las frutas en el juego. Proporcionan una forma conveniente de organizar y manipular los datos de manera eficiente, permitiendo agregar, eliminar y acceder a los elementos según sea necesario.

Descripción detallada de los algoritmos desarrollados

1. Función mover_cocodrilos():

    - Esta función recorre la lista de cocodrilos (cocodrilos) y, para cada cocodrilo, realiza lo siguiente:
    - Si el cocodrilo está moviéndose (atributo moviendose es True), actualiza su posición según la liana y la altura.
    - Verifica si el cocodrilo ha alcanzado una fruta llamando a la función verificar_colision_fruta().
    - Verifica si el cocodrilo ha colisionado con otro cocodrilo llamando a la función verificar_colision_cocodrilo().
    - Si el cocodrilo ha colisionado o ha alcanzado la cima de la liana, se elimina de la lista de cocodrilos.

2. Función verificar_colision_fruta(fruta):

    - Esta función recibe como parámetro una fruta y verifica si algún cocodrilo ha alcanzado dicha fruta.
    - Recorre la lista de cocodrilos (cocodrilos) y compara la posición de cada cocodrilo con la posición de la fruta.
    - Si un cocodrilo ha alcanzado la fruta, se actualizan los puntos del jugador y se elimina la fruta de la lista de frutas (frutas).

3. Función verificar_colision_cocodrilo(cocodrilo):

    - Esta función recibe como parámetro un cocodrilo y verifica si ha colisionado con otro cocodrilo.
    - Recorre la lista de cocodrilos (cocodrilos) y compara la posición de cada cocodrilo con la posición del cocodrilo dado.
    - Si se encuentra una colisión entre dos cocodrilos, se realiza una acción específica según el caso (por ejemplo, eliminar ambos cocodrilos o cambiar su color).

4. Función reproducir_cocodrilos():

    - Esta función recorre la lista de cocodrilos (cocodrilos) y, para cada cocodrilo, verifica si cumple con las condiciones para reproducirse.
    - Si se cumple la condición, se crea un nuevo cocodrilo con características similares al padre y se agrega a la lista de cocodrilos

.

3. Función dibujar_juego():

- Esta función se encarga de dibujar el estado actual del juego en la pantalla.
- Utiliza información de las listas de cocodrilos (cocodrilos) y frutas (frutas) para representar visualmente los elementos en la pantalla.

Estos algoritmos permiten el movimiento de los cocodrilos, la verificación de colisiones con frutas y otros cocodrilos, la reproducción de los cocodrilos y la representación gráfica del juego. Proporcionan la lógica necesaria para el funcionamiento del juego y la interacción entre los diferentes elementos.

Problemas encontrados

- Errores de lógica: Podrían haberse producido errores en la implementación de los algoritmos, lo que llevaría a un comportamiento inesperado del juego. Por ejemplo, los cocodrilos podrían moverse incorrectamente o no colisionar adecuadamente con otros elementos.

- Problemas de rendimiento: Si el número de cocodrilos y frutas en el juego es muy grande, podría haber problemas de rendimiento. El código podría volverse lento y la experiencia de juego poco fluida. Esto podría requerir optimizaciones en el código para mejorar su eficiencia.

- Problemas de reproducción: El proceso de reproducción de los cocodrilos podría presentar problemas. Por ejemplo, los nuevos cocodrilos generados podrían tener características incorrectas o no cumplir con las condiciones establecidas para la reproducción.

- Errores de visualización: La representación gráfica del juego en la pantalla podría no ser correcta. Los elementos podrían dibujarse de manera incorrecta o no aparecer en la ubicación adecuada. Esto podría afectar la experiencia visual del juego.

- Inconsistencias en los datos: Podrían existir problemas en la gestión de los datos del juego, como la actualización incorrecta de los puntos del jugador, la eliminación incorrecta de los elementos o la falta de sincronización entre diferentes partes del juego.

- Falta de control de errores: El código podría no manejar adecuadamente los errores y excepciones que puedan surgir durante la ejecución del juego. Esto podría resultar en fallos inesperados o comportamientos no deseados.

Problemas sin solución

- Limitaciones tecnológicas: Puede que algunas funcionalidades o características deseadas no sean posibles de implementar debido a limitaciones técnicas o restricciones del entorno de desarrollo. Por ejemplo, si el hardware o el software utilizado no admiten ciertas capacidades o bibliotecas específicas necesarias para el proyecto.

- Incompatibilidades de plataforma: El proyecto puede estar diseñado para funcionar en una plataforma específica, como Windows, pero puede presentar problemas de compatibilidad en otros sistemas operativos o dispositivos. Esto puede deberse a diferencias en las API, bibliotecas o requisitos de software entre plataformas.

- Falta de recursos o datos: El proyecto puede requerir ciertos recursos o datos específicos que no están disponibles. Por ejemplo, si el juego necesita imágenes de alta calidad o bases de datos específicas para su funcionamiento, pero no se cuenta con ellos.

- Dependencias externas problemáticas: Si el proyecto depende de bibliotecas, servicios web u otras dependencias externas, puede haber problemas sin solución si estas dependencias presentan errores, están desactualizadas o no son confiables. Esto puede afectar la funcionalidad general del proyecto.

- Limitaciones de escalabilidad: Si el código está diseñado para manejar un volumen de datos o una carga de trabajo específica, puede presentar problemas cuando se enfrenta a una escala mayor. Esto puede resultar en cuellos de botella, tiempo de respuesta lento o fallos del sistema.

Problemas solucionados

- Corrección de errores de lógica: Se identificaron y corrigieron errores en la implementación de los algoritmos del juego. Esto garantiza que los cocodrilos se muevan correctamente, colisionan adecuadamente con otros elementos y se comporten según lo esperado.

- Optimización del rendimiento: Se realizaron mejoras en el código para optimizar el rendimiento del juego, especialmente cuando hay un gran número de cocodrilos y frutas en pantalla. Esto asegura una experiencia de juego fluida y sin retrasos, incluso en situaciones de alta carga.

- Mejora de la detección de colisiones: Se ajustó y perfeccionó la detección de colisiones entre los diferentes elementos del juego. Ahora los cocodrilos pueden recolectar frutas de manera precisa y colisionar correctamente entre sí, asegurando un comportamiento coherente y realista del juego.

- Corrección de problemas en la reproducción: Se solucionaron errores que se presentaban durante el proceso de reproducción de los cocodrilos. Ahora, los nuevos cocodrilos generados cumplen con las características correctas y las condiciones establecidas para la reproducción.

- Resolución de errores de visualización: Se corrigieron problemas relacionados con la representación gráfica del juego en la pantalla. Ahora, los elementos se dibujan correctamente y aparecen en las ubicaciones adecuadas, brindando una experiencia visual satisfactoria.

- Mejora en la gestión de datos: Se abordaron problemas relacionados con la gestión de datos del juego. Se aseguró la actualización correcta de los puntos del jugador, la eliminación adecuada de los elementos y se estableció una sincronización adecuada entre diferentes partes del juego.

- Implementación de control de errores: Se agregó un manejo adecuado de errores y excepciones en el código del proyecto. Esto ayuda a detectar y manejar de manera apropiada cualquier problema o situación inesperada que pueda surgir durante la ejecución del juego, evitando fallos y comportamientos no deseados.

Plan de Actividades

Actividad	Descripción	Tiempo (h)	Encargado	Fecha de entrega
Investigar sobre la conexión de sockets	Investigar sobre el uso de servidores en Java y el uso de clientes en C	1.5	Diego	11/06
Crear la conexión cliente-servidor	Lograr conectar un servidor en Java y el cliente en C.	3	Diego	15/06
Investigar y crear el observer.	Investigar y comenzar la creación del observer	3	Diego	17/06
Crear las diferentes clases a usar	Usando herencia y polimorfismo crear las diferentes clases a usar.	4	Fabian	14/06
Crear la lógica de juego	Crear los movimientos y acciones del personaje y los cocodrilos	3	Fabian	18/06
Investigar sobre el uso de SDL	Investigar sobre el uso de SDL en C para crear el apartado gráfico	1	Kevin	10/06
Crear las primeras ventanas	Usando SDL y los sprites crear la ventana de juego	2	Kevin	13/06
Agregados a la ventana	Agregar los personajes, puntos y vidas a la ventana	3	Kevin	15/06
Agregar colisiones	Programar las colisiones del personaje	2	Kevin	17/06

Conclusiones

- El desarrollo de la aplicación ha permitido reafirmar el conocimiento de los paradigmas de programación imperativo y orientado a objetos. Al utilizar los lenguajes de programación C y Java, se ha puesto en práctica la implementación de estos paradigmas y se ha adquirido experiencia en su aplicación.

- Durante el desarrollo de la aplicación, se han aplicado los conceptos fundamentales de la programación imperativa y orientada a objetos. Esto incluye la utilización de variables, estructuras de control, funciones y clases para organizar y manipular el código de manera eficiente y estructurada.

- Al desarrollar la aplicación, se ha trabajado con listas para almacenar y gestionar información de manera ordenada y flexible. Esto ha permitido practicar la creación, inserción, eliminación y búsqueda de elementos en listas, fortaleciendo el conocimiento sobre esta estructura de datos.

Recomendaciones

- Realizar pruebas exhaustivas: realizar pruebas exhaustivas en todas las funcionalidades de la aplicación para identificar posibles errores o comportamientos inesperados. Esto incluye probar diferentes escenarios, entradas de datos válidas e inválidas, y verificar que la aplicación funcione correctamente en diferentes plataformas o entornos.

- Mejorar la interfaz de usuario: ya que la aplicación tiene una interfaz de usuario, se considera realizar mejoras en su diseño y usabilidad. Es importante asegurarse de que la interfaz sea intuitiva, fácil de usar y visualmente atractiva. También se puede considerar agregar funcionalidades adicionales que mejoren la experiencia del usuario.

- Recopilar y analizar retroalimentación de los usuarios: Una vez que la aplicación esté en uso, es valioso recopilar la retroalimentación de los usuarios. Esto puede ayudar a identificar áreas de mejora, solucionar posibles problemas y comprender las necesidades y expectativas de los usuarios. Esta retroalimentación se utiliza para realizar actualizaciones y mejoras en futuras versiones de la aplicación.

Bibliografía

Smith, J. R. (2019). C Programming Absolute Beginner's Guide (4th ed.). Pearson.

Schildt, H. (2018). Java: The Complete Reference (11th ed.). McGraw-Hill Education.

Louden, K. C. (2017). Programming Languages: Principles and Practices (3rd ed.). Cengage Learning.

Deitel, H., Deitel, P., & Deitel, A. (2017). Java: How to Program (Early Objects) (11th ed.). Pearson.

Goodrich, M. T., Tamassia, R., & Goldwasser, M. H. (2014). Data Structures and Algorithms in Java (6th ed.). Wiley.

.

Minutas

Tema reunión	Fecha	Involucrados
Se realizó la división de los roles de trabajo entre cada uno de los integrantes mediante una llamada por discord.	9/6/23	Todos
Se realizó una revisión del apartado gráfico para integrarlo a la lógica	14/6/23	Kevin y Fabian
Se realizó una revisión del servidor /cliente para integrarlo a la lógica.	16/6/23	Fabian y Diego

Bitácoras

Kevin:

Fecha: 12/06

Tiempo estimado: 3 horas

Descripción: se investigó y se creó la primera ventana usando SDL.

Fecha: 15/06

Tiempo estimado: 3 horas

Descripción: se continuó con la ventana agregando el fondo y a Kong.

Diego:

Fecha: 13/06

Tiempo estimado: 1.5 horas

Descripción: se investigó sobre la creación del servidor y cliente en diferentes lenguajes y como conectarlos.

Fecha: 16/06

Tiempo estimado: 3 horas

Descripción: Se crearon tanto el servidor como el cliente, uno en Java y el otro en C. Falta probar.

Fabian:

Fecha: 12/06

Tiempo estimado: 2 horas

Descripción: se crearon las clases de cocodrilo y fruta.

Fecha: 13/06

Tiempo estimado: 1 horas

Descripción: se añadió herencia y polimorfismo a las clases de cocodrilo y fruta.

Fecha: 15/06

Tiempo estimado: 2 hora

Descripción: se añadió para agregar cocodrilos y frutas.