UNIVERSIDAD INTERNACIONAL

DE LAS AMÉRICAS

ESCUELA DE INGENIERÍA INFORMÁTICA

PROYECTO DEL CURSO

ESTRUCTURAS DE DATOS Y ALGORITMOS

JENNIFER LÓPEZ CAMPOS

RAFAEL JIMENEZ ARCE

ANTHONY ALVAREZ DELGADO

PROFESOR

DANIEL ALVAREZ DELGADO

SAN JOSÉ, COSTA RICA

Programación Recursiva

La recursividad en el código se utilizó en la clase npc_mage del paquete modelo con el método setDialogue este se llama a sí mismo de manera repetida hasta que se cumple una condición de salida. Para que se cumpla, se llama al método getDialogue para obtener un diálogo correspondiente a un index determinado y se almacena en el arreglo dialogues. Luego se llama nuevamente a setDialogue con un índice incrementado en uno, lo que hará que el método vuelva a llamarse a sí mismo hasta que se alcance la condición de salida.

Modelo Vista Controlador:

El código se distribuyó en en tres archivos cumpliendo así con el MVC, en la carpeta model, se programó todo lo relacionado al manejo de datos dentro del juego. En la carpeta "view" se controla la parte gráfica con la que va a interactuar el usuario, en las cuales se puede mencionar métodos como "add window" donde se crea la ventana donde el "gamepanel" estará y es donde el jugador podrá observar el mapa.

Finalmente, la carpeta "controller" funciona como intermediario entre model y view, tal como en el método setObject que coloca los elementos necesarios dentro del mapa: monedas, espadas, escudos, poción roja, corazones, NPC´s y enemigos. Como ejemplo se tiene el método damageReaction que funciona para hablar con los NPC's y que muestre los diálogos en pantalla cada vez que se acerca a uno de ellos.

Estructura de datos

Se utilizó una lista enlazada con la clase LinkedList en el paquete controller, esta tiene dos nodos: head y tail. El primero de ellos apunta al primer nodo de la lista y tail apunta al último nodo de la lista. Además, la clase LinkedList tiene dos métodos: addNode() y getNode(). El método addNode() agrega un nuevo nodo a la lista, y el método getNode() devuelve el nodo en una posición específica de la lista. La implementación de la lista enlazada se realiza mediante el uso de la clase Nodo, que tiene un valor entero y una referencia al siguiente nodo en la lista.

TAD:

Al analizar los requerimientos del juego, se llegó a la conclusión que se debería utilizar los tipos de datos abstractos en los "tile", y "direction" con los cuales al hacer sus respectivas

operaciones, se logra calcular la dirección en el mapa con respecto a la pantalla en la cual el jugador se va a mover.

Programación Orientada a Objetos:

A lo largo del programa podemos encontrar POO aplicada en diferentes clases, por ejemplo en la clase “MonsterGreenSlime” se hereda de la clase “Entity” los atributos que le permiten al NPC ser un enemigo para el jugador. Tambien se puede apreciar el paradigma abstracción Entity donde se le atribuyen características específicas al jugador y a los enemigos.

En la clase Player, se aplica el paradigma Herencia en el cual se heredan los atributos que permiten ciertas funcionalidades al jugador por ejemplo: ubicar su posición tomando como coordenadas X y Y.

En cuanto a encapsulamiento, en la clase GamePanel se aplicó este mecanismo al ocultar las OperacionesTAD para que el usuario no tenga acceso al funcionamiento interno de la dirección de la pantalla.

Interpretación de Darksiders: Warmastered Edition:

Este programa es un juego basado en el universo de DarkSiders, que representa una pantalla del juego original, además de la representación de una pantalla del juego original, la reinterpretación se basa en la idea de cómo se hubiera visto el juego a finales de los años 70 comienzos de los años 80 donde los juegos tipo arcade estaban en su auge. El objetivo del juego es limpiar el desierto de los temibles slimes amarillos que aparecen después de matar al jefe.

Algoritmo de Ordenamiento

El algoritmo de ordenamiento se realizó en la clase npc_mage del paquete modelo con el método “ShuflleArray”. En particular, este método implementa un algoritmo aleatorio para cambiar la posición de los elementos de un array.

Algoritmo de Búsqueda:

Para efectos del programa, se utilizó búsqueda secuencial en el que se busca un nodo en el índice generado aleatoriamente con las siguientes clases: se crea una lista, se genera una lista

aleatoria de números entre el rango del 1 al 100, se agregan los nodos con los números aleatorios a la lista enlazada, se genera un número aleatorio el cual ayuda como índice, se busca el nodo en el índice generado aleatoriamente y finalmente se verifica si se encontró o no el nodo en el índice el cual va a determinar la dirección del NPC basada en el valor del nodo.

Implementación de Interfaces:

El juego comienza con una pantalla de inicio, al presionar "Enter" el usuario ingresa al juego. Dentro del juego, el usuario puede pausar el juego presionando "P", acceder al menú de personajes presionando "C" o salir del juego presionando "Escape". Controles del juego: El usuario controla al personaje con las teclas "W", "S", "A" y "D". El personaje dispara con la tecla "F" y ataca con la tecla "Enter".

Problemas conocidos: Si el usuario presiona "Enter" muy rápidamente después de iniciar el juego, es posible que el cambio de música entre la pantalla de inicio y el juego no se reconozca. Este problema se puede resolver simplemente reiniciando el juego.