Descripción de las Estructuras de Datos Creadas

Para la implementación de la solución se crearon listas, colas y pilas a continuación se describirán a fondo cada una de ellas. Se crearon listas para las balas las torres y el puntaje del jugador, estas listas están compuestas de link o nodos que tiene una referencia a el siguiente nodo, y además un dato dentro de ellos que varía dependiendo de la lista, así se conectan formando una lista enlazada, las listas tienen funciones para remover el primer elemento, cuando se usa esta función se cambia las referencias de los nodos para que ningún nodo referencie a el que se desea eliminar y el recogedor de basura de Java lo elimina, además posee una función que la deja insertar un elemento en la primera posición de la lista, además tiene funciones para obtener el primer elemento o un elemento en cualquier posición de la lista, por último las listas tienen una variable size que devuelve el tamaño de la lista. Las colas que fueron creadas fueron usadas para los enemigos y para los jefes, siguen el sistema FIFO (First In First Out), lo cual significa que cuando un enemigo entraba a la cola era el primero que va a salir, la cola contaba con funciones que la dejan encolar nuevas elementos o también puede desencolar el elemento del tope además posee una función que retornaba el objeto en el frente de la cola, y además lo remueve de la cola para poder acceder a los demás elementos de la cola, la cola estaba compuesta de nodos que contienen una referencia a qué nodo está sobre ellos y además poseen un tipo de dato que en este caso son enemigos. Por último esta l estructura de las Pilas que funcionan por la lógica LIFO (Last In First Out) lo que esto significa es que el último elemento que entre va a ser el primero que sale el inverso de las colas, las pilas que fueron desarrolladas se usaron para controlar los powerups, las pilas creadas están compuestas por nodos que tiene un dato, en este caso un dato tipo Power Ups, y una referencia a el siguiente elemento en la pila, las funciones de la pila lo dejan adherir elementos a la pila, donde el último que es insertado se establece como el primero que debería salir, además puede remover elementos de la pila y moverlos de arriba o abajo si fuera necesario. El desarrollo de estas Estructuras de Datos fue crítico para poder implementar la solución al problema dado.

Descripción de las Bibliotecas Utilizadas:

JFrame: Una biblioteca que es un extendimiento de Swing(librería gráfica utilizada en Java), que se usa para poder generar ventanas sobre las cuales se pueden adherir diferentes formas o paneles.

JPanel: Biblioteca que da acceso a contenedores que pueden ser insertados en Jframes que pueden contener una variedad de elementos.

ActionListener: Biblioteca que permite responder a diferentes ActionEventes de maneras diferentes.

ActionEvent: Biblioteca que deja saber que una acción ocurrió sobre algún componente del programa esto se pasa a el Actionlistener

KeyEvent: Biblioteca que deja escuchar cuando una tecla ha sido presionada en el teclado el evento generado por esto se pasa a la clase KeyListener.

KeyListener: Biblioteca que permite realizar acciones que viene después de un evento recibido del KeyEvent y hacer lo que se quiera con esa información.

Java.io: Biblioteca que permite la escritura de diferentes cosas a archivos de texto o incluso la creación del archivo de texto.

Random: Biblioteca que da acceso a la clase random que de integers aleatorios en un rango definido.

Problemas Encontrados y Soluciones:

El problema más grande que fue encontrado y el único en el que se buscó una solución y no se logró encontrar fue la del movimiento en diagonal, este problema se da debido a las clases KeyEvent y KeyListener, ya que ellas solo pueden recibir una tecla a la vez no es posible poder tocar ambas teclas de arriba y derecha al mismo tiempo por lo que no se puede mover en diagonal. Para poder solucionar esto se encontro una solucion crear un array de 2 posiciones que guarde las dos teclas más recientes y revise si el movimiento debe ser diagonal o de algún otro tipo, esto implicaba recursividad, la razón por la que no fue implementada es porque se encontró errores recorriendo el array de manera correcta y por razones de tiempo y ya que no es una parte tan crítica se dejó de lado. Bibliografía solicitada:<http://stackoverflow.com/questions/22580933/diagonal-movement-of-a-sprite>

<http://stackoverflow.com/questions/18112124/how-to-move-a-visible-image-diagonally>

<http://www.java-gaming.org/index.php/topic,11102>.