Guion proyecto progra 3

Introducción (máximo 20 segundos):

-Muy buenos días a todos, en esta oportunidad el grupo 4 del curso de programación 3 presentará el proyecto final MAZE, en este video tocaremos los siguientes puntos (eso se muestra con edición para ahorrar tiempo), ahora le doy la palabra a (integrante que sigue) que nos describirá las características de funcionamiento del proyecto.

Descripción de las características de funcionamiento del proyecto (máximo 50 segundos):

-Muchas gracias (integrante que hizo la introducción), el proyecto consiste en un juego de laberintos, en el cual el jugador tiene que encontrar la ruta de un punto A a un punto B. El programa genera el laberinto perfecto mediante el uso de algoritmos que explicaremos posteriormente. Luego se colocan los puntos de entrada y partida y empieza el juego, como datos adicionales el programa es capaz de emplear un bot en el juego y está la opción de jugar contra reloj. En datos más técnicos, este programa fue desarrollado en c++ y usando la librería raylib en la parte grafica. Ahora le cedo la palabra a (integrante que hace la parte siguiente).

Describir los algoritmos más importantes utilizados en el desarrollo (máximo 1 minuto):

El proyecto posee 4 algoritmos de búsqueda y 2 algoritmos de generación del laberinto. El Depth First Search es un algoritmo el cual sigue un camino hasta llegar a su final y cuando no le quedan más caminos retrocede al ultimo nodo que posea caminos sin recorrer, así recursivamente o iterativamente hasta llegar a su destino. El algoritmo Breadth First Search se diferencia del anterior porque este primero revisa todos los nodos adyacentes al punto inicial antes de pasar al siguiente y repetir el proceso hasta llegar al destino. Gready Best First Search es un algoritmo que calcula el costo de cada nodo y en base a este costo elige la mejor opción para llegar al objetivo, empezando en el inicio calcula el costo de los adyacentes y se mueve hasta llegar al objetivo, estos cálculos se realizan mediante una función heurística. Por último tenemos el algoritmo A\* el cual funciona muy similar al anterior, este calcula los costos de llegar del punto A al B, calculando el costo total y pasando a cada nodo para calcular el costo más bajo y colocarlo en la pila de prioridad recursivamente hasta llegar al objetivo y retorna el camino menos costoso gracias a la pila de prioridad.

Para la generación del laberinto se utiliza el algoritmo Wilsons que funciona de la siguente manera:

Primero iniciamos el laberinto como un rectángulo cerrado, posteriormente se elige aleatoriamente el punto de partida y se elige un punto al azar no visitado, se inicia la caminata aleatoria hasta llegar al punto ya visitado, posteriormente se retrocede por el camino recorrido hasta llegar al punto de partida o a otro punto visitado anteriormente (de esta manera aseguramos que sea un laberinto perfecto) y se repite hasta haber visitado todas las celdas del laberinto.

Estructuras de datos:

Principalmente el proyecto cuenta con 2 grandes grupos de estructuras de datos, los propios de la librería estándar “std::” y los que empleamos desde raylib para la parte grafica. Por el lado de la librería estándar tenemos los que vimos en clase desde queue, vector, set, map, arrays, etc. Todo esto con el fin de optimizar el uso de contenedores a las necesidades especificas de cada tarea en el código fuente. Y por la parte de raylib tenemos principalmente rectangles, textures, color y Font, estos los explicaremos porque no son de conocimiento general en la clase. El rectangle crea una estructura que se representa como rectángulo, con alto, ancho y altura, para poder posicionar en la pantalla. El texture sirve para poder cargar texturas 2d en nuestro juego y pasar de una visualización tipo consola a una interfaz grafica. Color sirve para asignar colores usando la notación rbg como sus atributos y por ultimo Font es utilizado para las fuentes de texto del juego.

Organización del código fuente:

El código esta organizado de la siguiente forma:

En la carpeta “src” está todo el código realizado en c++, dentro de esta carpeta tenemos subcarpetas que contienen el código fuente de cada parte según el nombre, para no ser redundante explicaremos brevemente una: La carpeta Bot contiene el Bot.cpp y el Bot.hpp que es el código del bot como lo indica el nombre, estos se incluyen en game que es el archivo donde se encuentran las funciones que permiten el funcionamiento del juego, luego se encuentra el main que corre el game que tiene incluidas todos los archivos de las subcarpetas de “src”. Las carpetas examples y assets contienen muestras de pngs empleados para la parte grafica del juego y el ultimo archivo relevante es el README.md que se encarga de informar a los usuarios acerca del programa.

Mostrar funcionamiento (simplemente grabamos el juego funcionando)

Conclusiones

Como podemos observar el programa cumple con todas las especificaciones mencionadas en el video, empleando los recursos de c++, sus librerías y los conocimientos adquiridos a lo largo del curso para generar este proyecto. Sumado a las diciplinas no propias de programación como es el trabajo en equipo y la gestión de equipos para el desarrollo del trabajo. Dando para nosotros un resultado satisfactorio.

Recomendaciones: