Instituto Tecnológico de Costa Rica

Análisis y diseño de Algoritmos

Guest Who!!

Integrantes:

Daniel Alvaro Sáenz Cordero

Brayan De Jesus Barquero Madrigal

Profesor:

Walter Jeancarlo Alvarez Grijalba

13 de junio de 2023

Introducción

El siguiente proyecto, "Guest Who!!"tiene como objetivo ser un programa desarollado en el lenguaje c++, capaz de simular, aunque de una manera sustancialmente diferente el juego de "Guest Who!!". En el cual 2 jugadores tienen un tablero lleno con fotos de diversas personas o personajes. Cada jugador debe de escoger a un personaje con el cual jugará la partida. Para el caso de este proyecto se escoge unos de los personajes preestablecidos; Mario, Luigi, Peach y Bowser. El programa debe realizar una serie de preguntas con el fin de determinar el personaje seleccionado. Sin embargo, este programa posee requerimientos y restricciones, por lo que deberá cumplir con ciertas condiciones, de las cuales las más importantes o destacables son las siguientes:

- Lectura de archivos: El programa deberá ser capaz de leer dos archivos de tipo .txt con la informacion necesaria para que inicie el juego.
- Implementacion de arbol:se debe utilizar un árbol binario para almacenar las preguntas del archivo "questions.txt", dicho árbol debe construirse en base al archivo "index.txt".
- Desarollo del programa:El programa deberá realizar la lectura de los archivos y posteriormente generar el árbol de almacenamiento de las preguntas, un vez hecho esto, se necesita mostrar en consola un mensaje de bienvenida y a continuación imprimir la pregunta del nodo raíz y la opción sí o no, una vez el usuario responda a las opciones presentadas, el proceso anterior debe repetirse con los nodos internos hasta que se encuentre con la respuesta en los nodos hoja.
- Re-start:Se debe agregar una pregunta final, que consulte si el personaje encontrado es el correcto, sino debe volver a iniciar la búsqueda en el árbol binario

Ambiente de desarrollo

Durante el desarrollo del proyecto se utilizaron varios ambientes y herramientas para desarrollar el proyecto, por lo tanto, a continuación, se indicarán las herramientas utilizadas y su propósito.

Se utilizaron tanto los sistemas operativos de linux como de Windows.

El código para el juego fue desarrollado en el lenguaje de programación C++.

El editor de código utilizado principalmente es Visual studio code.

Se utilizaron los navegadores de Chrome y Opera GX para la investigación de los materiales necesarios para el desarrollo del mismo.

Se utilizó Overleaf como programa de escritura con el fin de utilizar latex para el documento escrito.

Se utilizaron también los blocs de notas incorporados en windows para generar los archivos txt.

Se utilizo también la herramienta de chatgpt con el fin de depurar código.

Estructuras de datos usadas

Primeramente, definimos un header file con la clase Node y binary tree. Externamos las definiciones de los datos de tipo string p1, p2, p3, p4, r1, r2, r3 y demás con el fin de utilizarlos en otros lugares del código con mayor facilidad. Iremos explicando las clases definidas en el header file junto con sus métodos localizados en el cpp. La clase Node esta definida como clase amiga de binary tree, lo cual nos permite acceder a las definiciones o miembros privados de esa clase. Luego definimos como private un dato de tipo char, creamos un puntero al nodo hijo izquierdo y otro al derecho. Para los atributos públicos, tenemos el constructor de la clase nodo, que recibe un valor y lo asigna al dato del nodo seguido de otro constructor de la clase Nodo que recibe punteros a los nodos hijos izquierdo y derecho, así como un valor para asignar al dato del nodo. Seguido de una función miembro publica que devuelve el valor almacenado en el nodo, seguidamente tenemos 2 funciones que devuelven los punteros a loas nodos hijos izquierdo y derecho. 3 funciones de tipo void que permiten cambiar el valor de un dato de un nodo, e intercambiar los nodos hijos izquierdo y derecho. La clase Binary Tree define como private el puntero al nodo raíz. Luego en public; tenemos un constructor que crea un árbol binario vacío y otro que crea un árbol binario con un nodo como raiz, método de tipo vacío llamado “Proot” que asigna un nodo como raíz del árbol binario, un puntero al nodo raíz del árbol (seguidamente uno similar), un verificador que comprueba si el árbol este vacío mediante un booleano, dos métodos que devuelven los punteros al hijo izquierdo y derecho del nodo raíz. Seguidamente tenemos un método que crea un nuevo árbol binario con un nodo raíz, hijo izquierdo y derecho ya dados. Finalmente, en public tenemos 3 variables de tipo vacío; una para imprimir el valor de un nodo dado, toma un puntero que apunto a un valor almacenado. La segunda función de tipo vacío simplemente llama a la función preorden pasando como argumento el puntero a la raíz del árbol. Finalmente, un vacío para preorden que realiza un recorrido preorden en el árbol binario, visita primero la raíz, luego el subárbol izquierdo y finalmente el derecho, este método es recursivo. Estas son básicamente las clases y métodos definidos

en el el header file que tiene relación con el cpp. Sin embargo, hay algunos métodos que se utilizar en el main que están determinados solo en el cpp. Respecto a las fuciones referentes a la lectura de archivos tenemos una función de tipo vacío, “reader questions” que abre el archivo .txt, valida que este abierto y lee las preguntas que se localizan ahí, las asigna a una variable y cierra el archivo, en caso de no encontrar un archivo con estas características notifica al usuario. La función de tipo vacío “reader index” hace lo mismo con un archivo .txt llamado index, pero con la peculiaridad de que ignora las preguntas que se encuentran en este para guardar como dato las respuestas. Finalmente, en el cpp tenemos una función de tipo booleano que comprueba que la respuesta dada al programa por medio de la consola cumpla con los requerimientos estipulados. En caso de que la respuesta sea dada con otro formato no se tomara en cuenta y repetirá la pregunta. La última función del cpp es una función de tipo char “findCharacter” que recibe una instancia del árbol binario, un mapa de preguntas (generado en el archivo main). Realiza un recorrido descendente por el árbol binario, haciendo preguntas las preguntas alojadas en cada nodo interno, determinando según la respuesta, si continua su camino por el subárbol izquierdo o derecho. Llega hasta el nodo hoja y retorna el valor de este. En el archivo main se declaran algunas variables con el fin de usarlas en cualquier sitio, pero cabe recalcar el uso de la biblioteca con la clase map, con el fin de acceder a los datos de manera más eficiente mediante su uso de clave-valor. Directamente con la función main, llamamos las funciones de lectura de archivos desde un inicio, declaramos 5 objetos de la clase BinaryTree y 4 punteros de la clase Node. Luego definimos un booleano para poder establecer un ciclo que nos permitirá cumplir un re-start en caso de que así lo dispongamos. El ciclo while establece las relaciones entre los nodos con el fin de formar el árbol mediante las llamadas al método respectivo a “newTree”. Se establece uno de los objetos (a) como la raíz del árbol mediante el método explicado anteriormente “Proot”. Luego creamos una especie de diccionarios que realmente lo que hacen es asignar las preguntas y respuestas de los archivos .txt a los diferentes nodos establecidos, esto debido a la complejidad de establecer la relación de nodos utilizando únicamente los .txt. Declaramos una variable char “charaterFounded”, y le asiganos

el valor obtenido de la función “FindCharacter”. Establecemos una condicional que, en caso de encontrar el personaje, imprime el personaje encontrado. Creamos un string decisión, un booleano continue igual a “true” y solicitamos al usuario que nos responda si ese es su personaje. Creamos un ciclo while que se mantiene siempre y cuando continuar sea igual a “true”. Si la respuesta del usuario es si, la variable “result” (encargada de determinar el primer ciclo while) se asigna como false, y continue también, por lo que ambos ciclos se terminan y el programa finaliza. En caso de respuesta negativa “result” se mantiene en true, por lo que se repite el primer ciclo while, pero continue cambia a false, rompiendo el segundo ciclo, esto con el fin de detener el segundo ciclo while.

Pasos para ejecutar el proyecto

Los pasos necesarios para ejecutar el proyecto de manera correcta, así como los necesarios para jugar sin complicaciones son los siguientes:

- Primeramente asegúrese de que su computador posee un sistema operativo compatible con c++, se recomienda usar linux o windows.
- Luego, descargue el archivo comprimido respectivo al proyecto.
- Seguidamente descomprima el contenido en una carpeta vacía, esto con el fin de evitar cualquier error a la hora de ejecutar los archivos.
- En un compilador o idle abrá los archivos arbol.cpp, arbol.h y arbol.cpp.
- Ahora, ejecute el archivo en la terminal respectiva a su sistema operativo
- Este abrirá una terminal que le mostrará una serie de preguntas, de una en una. A la derecha de la pregunta se le solicitará responder a la pregunta con un si o un no. El programa acepta tanto mayúsculas como solo la primera letra del si o no.
- Continue respondiendo las preguntas necesarias.
- Al final el programa desplegara el personaje encontrado a base de sus respuestas.

Actividades realizadas

Brayan Barquero Madrigal:

- Planteo de objetivos e investigación. 3 horas. Martes 30 de mayo.
- Investigación. 2 horas. miércoles 30 de mayo.
- Definición de clases y atributos. 2 horas. Jueves 1 junio.
- Lectura de archivo de texto. 3 horas. Viernes 9 de junio.
- Implementación del uso de preguntas en un árbol. 6 horas.
- Traducción del código, comentarios, arreglos necesarios: 5 horas

  Total de horas: 20 horas.

Daniel Sáenz Cordero:

- Determinación de objetivos, consultas, investigación, creación de seudocódigos. 3 horas. Martes 30 de mayo.
- Reutilización de códigos: 3 horas. Viernes 2 de junio.
- Intento de implementación de Código de árbol binario. 3 horas. Domingo 4 de junio.
- Investigación sobre árboles. 2 horas. Domingo 2 de abril.
- Flujo del programa, correcta disposición del makefile, cpp, h. 5 horas. Viernes 9 de junio.
- Re-Start de juego. 3 horas. sábado 10 de junio.
- Investigación de uso sobre latex, redacción y finalización del documento. 4 horas. Domingo 11 de junio.

- descripción de las estructuras utilizadas. 4 horas. Lunes 12 de junio.

  Total de horas: 27 horas.

Total de horas entre los 2 estudiantes: 47 horas.

Comentarios finales

El proyecto funciona correctamente en su totalidad. Las preguntas y las respuestas se crean con el uso de los archivos ,txt, la función extra de re-start funciona correctamente. En un inicio los objetos del árbol binario se creaban y destruían en cada ciclo while. Pero después de algunos arreglos esto quedo arreglado con el fin de optimizar un poco más el programa.

Conclusiones y recomendaciones

Los árboles binarios y demás nos permiten utilizar una gran variedad de datos, en este caso utilizamos un árbol binario adaptado para el uso de datos cualitativos. Las estructuras de datos son sumamente importantes para el manejo de archivos, ya que nos permiten organizar de manera correcta la información. Así manejar diferente tipos de datos de manera fluida, en este caso fuimos capaces de utilizar un árbol binario, en conjunto con herramientas como; ciclos while, condicionales, nodos, etc... con el fin de encontrar un personaje determinado, de manera correcta. Es de especial importancia definir correctamente que clase de datos determinamos en un proceso, así de como los determinamos, ya que esto puede evitar el consumo excesivo de recursos. Así como errores que puedan en ciclar el programa. Gracias a este trabajo se pudo entender y mejorar el manejo de árboles, latex y demás estructuras de datos.

Bibliografía

C++ Variables de tipo map — Aprende Programación Competitiva. (s. f.). https://aprende.olimpiada-informatica.org/index.php/cpp-map

GitHub: Let's build from here. (s. f.). GitHub. https://github.com

Joyanes, L., Zahonero, I., Sanchez, L. (2007). ESTRUCTURA DE DATOS EN C++ (1.a ed.). McGraw-Hill.

OpenAI. (2023). ChatGPT (Modelo de lenguaje grande). Recuperado de https://openai.com/chatgpt