

**Algoritmos y estructura de datos**

Trabajo final

TÍTULO:

Explorer

**Integrantes:**

Angulo Valle, William U201817762

Izarra, Carlos U 20181C45

Flores Palermo, Oscar Daniel U201716498

**Docente: Luis Martin Canaval Sanchez**

Lima, Noviembre de 2019

Indice

1. Introducción
2. Objetivo del Estudiante
3. Capítulo 1: Presentación
4. Capítulo 2: Marco conceptual
5. Capítulo 3: Diseño del proyecto
6. Capítulo 4: Diseño del producto
7. Capítulo 5: Diseño del proceso de desarrollo de la solución
8. Conclusiones
9. Bibliografía
10. Anexos

Introducción

En el siguiente trabajo se realizó una réplica de un buscador dentro de los archivos de la memoria de la computadora, el cual es una aplicación que busca en una carpeta y sus subcarpetas y encuentra una lista de archivos o carpetas que concuerdan con parámetros de búsqueda. Mediante este informe se dará a conocer los métodos y procedimientos con los que se pudo obtener el resultado.

Objetivo del Estudiante

Implementar un programa, con sus conocimientos aprendidos en clases, que ayude a facilitar la agilización de la búsqueda de archivo.

El programa sea fácil de utilizar y se las personas lo puedan usar para sus diferentes trabajos o búsquedas.

Capítulo 1: Presentación

Explorer es un explorador de archivos en el cual puede buscar de manera eficiente y rápida la localización de archivos, también puede filtrarlos por búsqueda de una palaba letra extensión y peso. Al hacer el escaneo permite encontrar más rápido la localización ciertos archivos dando la dirección de donde está guardado, además puede agregar y eliminar carpetas.

Capítulo 2: Marco conceptual

-Lambdas: Lo usamos para las hacer los filtrados de búsqueda de un parámetro de letras o números.

-Arboles: Se usaron para poder guardar los datos de los archivos por nombre, extensión, fecha y peso del archivo, lo balanceamos para que la búsqueda sea más rápida.

-Listas: Se implementaron una lista doblemente enlazada con puntero al inicio y al final para poder recorrer el árbol y ordenarlo según el criterio que el usuario desee.

-Recursividad: Nos ayudó a encontrar los elementos en los arboles balanceados sea mas sencillos de ubicar y con una velocidad rápida

-Vectores: Se usó para guardar los datos de los arboles como auxiliares, al eliminar elementos de los árboles o hacer algún filtrado para mostrar los elementos.

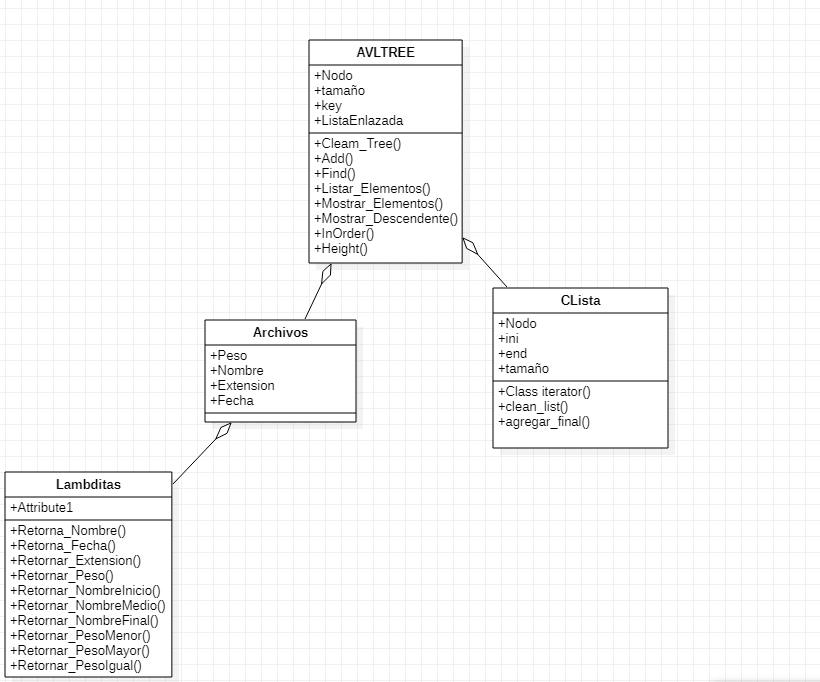
Capítulo 3: Diseño del proyecto

Este proyecto está diseñado para lograr simplificar la búsqueda de los archivos necesarios de manera rápida y sencilla, también vimos como intentar una interfaz sencilla para la comprensión rápida del usuario con el proyecto.

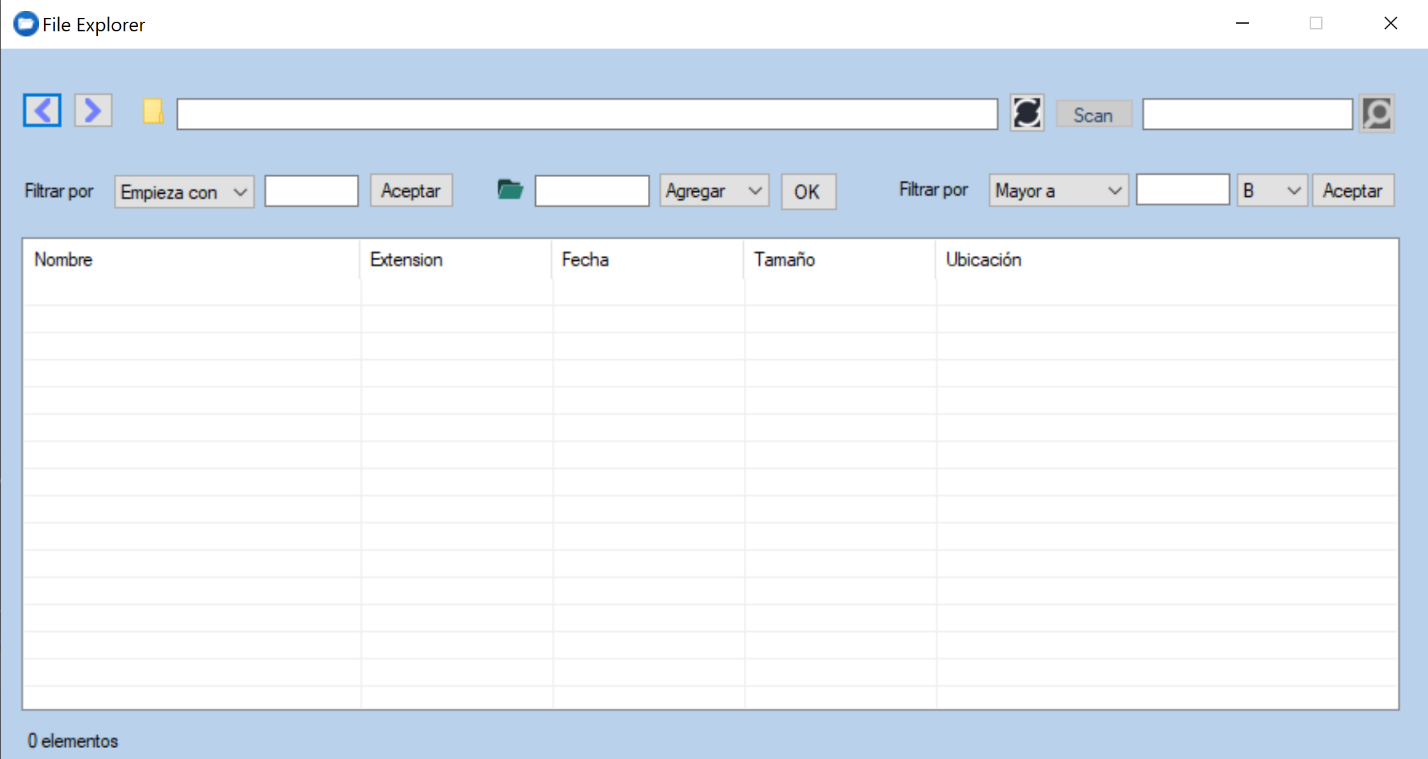
Capítulo 4: Diseño del producto

Un explorador que permite facilitar la tarea de búsqueda de archivos, dando varias opciones de cómo se puede buscar, filtrar los archivos y con una interfaz fácil de entender.

1. Con una buena velocidad de respuesta se puede minimizar el tiempo de búsqueda dando a la larga un ahorro de energía para las personas.
2. Tiene varias opciones de encontrar los documentos o filtrarlos como uno desea, son unos requisitos que se le pide al programa. En cuanto a estética se puede es muy fácil de utilizar y entender por lo que es un requisito no funcional.
3. El diagrama de clase



1. Diseño de la interfaz de usuario:



Capítulo 5: Diseño del proceso de desarrollo de la solución

Para que sea más fácil de usar el programa se decidió usar Windows Forms por su fácil de manejo en las personas no muy acostumbradas a la tecnología.

Motivación

Nuestra motivación fue hacer un trabajo que pueda facilitar la búsqueda de los archivos y el desafío que presentaba al implementar en el programa los árboles, listas, vectores, recursividad.

Problemas:

En varios puntos tuvimos problemas con la implementación del código:

-Como en las listas en un punto tuvimos una deficiencia para agregar en la parte final de la lista.

-También en las fechas no sabíamos cómo hacerlo y no queríamos poner letras y solo números.

-En los árboles en la deficiencia del tiempo que se demoraba mucho en escanear y buscar.

Solución:

Las soluciones que terminamos implementando en el código:

-Crear un auxiliar dándole un elemento y poniendo que cuando sea de tamaño 0 sea el inicial y el final a la vez y luego que siga para cualquier tamaño que agregue como el siguiente del último.

-Le pedimos ayuda al profesor para poder averiguar las fechas de su última edición del archivo, y tuvimos que convertirlo en número las palabras.

-Por la sugerencia del profesor decidimos separa el escanear y buscar en dos botones y que harían su función y mejoramos el método de buscar en el árbol.

Conclusión:

El proyecto nos ayudó a aprender más sobre nuestra carrera, además esto nos ayudara a aprender cómo hacer este tipo de proyectos, pero en mayor escala en un futuro y poder contribuir al desarrollo de la tecnología venidera. EL uso del indexado nos ayuda a tener un mayor orden en el trabajo ya que es así como podemos realizar una búsqueda, además del ordenamiento adecuado

Objetivos:

Nuestro objetivo es crear una librería que pueda ordenar “DataFrames” y crear nuevos a partir de este de manera eficiente y eficaz con algoritmos de ordenamiento.

\* Examinar los distintos tipos de estructuras de datos para poder recopilar su información para poder construir una mejor estructura.

\* Aplicar lo aprendido en clases, para darle un correcto funcionamiento a nuestro programa.

\* Disminuir la redundancia e inconsistencia de los datos.

\* Evitar dificultad en el acceso a los datos.

\* Crear e innovar en la creación del programa identificando funciones que pueden presentar un mayor rendimiento.

Alcance de proyecto:

Mediante este proyecto, queremos emplear nuestros conocimientos en el curso, para poder implementar un programa que nos facilite el manejo de “dataframes”. Gracias a este programa podremos importar, indexar y filtrar datos. Además, emplearemos algunos algoritmos

de ordenamiento aprendidos en clase para ordenar los datos en las columnas. Para avanzar este proyecto, lo dividiremos en 3 hitos, en los cuales iremos avanzando diariamente, guardando y clonando el proyecto actual, gracias a GitHub. Debemos organizar y dividir tareas entre todos los miembros del grupo para lograr terminar el proyecto a la fecha indicada y con todos los requerimientos.

Diagrama de clases:

Plan de trabajo detallado:

El proyecto estará disponible para todos los integrantes del grupo; sin embargo, constará de 3 copias, y uno principal, gracias a que GitHub nos permite crear nuevas ramas podemos cambiar el código continuamente y mostrarlo a los demás integrantes del grupo. Además de esto, GitHub nos permite utilizar herramientas como los milestones e issues los cuales nos permitirán asignarnos tareas y completarlas progresivamente. Los milestones son tareas en paquetes, de esta manera estarán elaborados los entregables e hitos, mientras que las tareas que están dentro de los entregables e hitos se encontrarán dentro de los issues. De esta manera nuestro plan de trabajo será eficiente y podremos ver el avance de cada integrante.

Bibliografía:

Windows Forms (2017). Documentation. Recuperado de: https://docs.microsoft.com/en-us/dotnet/framework/winforms/

CPLUSPLUS. Vector. Recuperado de: http://www.cplusplus.com/reference/vector/vector/?kw=vector

CPLUSPLUS. std::string. Recuperado de:http://www.cplusplus.com/reference/string/string/?kw=string

CPPREFERENCE. Filesystem library. Recuperado de: https://en.cppreference.com/w/cpp/experimental/fs

Anexos:

YouTube Programación ATS: https://www.youtube.com/watch?v=ld4nzao5XAc