Instituto Tecnológico de Costa Rica

Campus Tecnológico Central de Cartago

Escuela de Ingeniería en Computación

Taller de Programación

REVISIÓN TÉCNICA DE VEHÍCULOS (ReTeVe)

Francisco Kuo Liu - 2023394698

Anthony Barrantes Jiménez - 2023152240

Semestre I 2023

2023/06/19

Tabla de Contenidos

Enunciado del Proyecto

Desarrollar un programa para administrar una parte de un proceso de revisión técnica de vehículos llevada a cabo en una estación especializada. La revisión consiste en examinar vehículos para obtener un resultado de la revisión y si el vehículo es apto para circular se emite un certificado de tránsito. De no ser apto para circular este certificado no se emite, en su lugar el vehículo necesitará reinspección o podría ser que la revisión determine que el vehículo debe sacarse de circulación.

El programa tendrá funciones para el registro de las citas de revisión de los vehículos, el ingreso de vehículos a la estación, un tablero que muestra en qué parte de la revisión está el vehículo dentro de la estación, el obtener los resultados de la revisión, una lista de posibles fallas y datos de configuración del sistema.

Para efectos de esta especificación, cuando se considere necesario se hará referencia a los valores mostrados actualmente en la sección de configuración, por ejemplo, cuando se hable de cantidad de líneas de trabajo en la estación será 6 que es su valor actual. Tenemos entonces que en la estación hay 6 líneas de trabajo: una línea de trabajo es una cola (primero que entra, primero que sale) donde están los vehículos a revisar.

En cada línea de trabajo el vehículo debe pasar secuencialmente por 5 puestos de revisión:

- Puesto 1: revisiones generales (luces, llantas, vidrios, etc.)
- Puesto 2: banco de amortiguadores
- Puesto 3: sistema de frenado
- Puesto 4: detector de holguras
- Puesto 5: emisión de contaminantes

En cada puesto puede haber sólo un vehículo, y cuando sale del puesto puede estar sin fallas o con fallas leves o graves.

Al final de la línea de trabajo, después del puesto 5, un vehículo aprueba o no aprueba la revisión técnica. “Cola de espera”: hay una por cada línea de trabajo, antes del puesto 1 los vehículos están en esta cola esperando la entrada a ese puesto de revisión. “Cola de revisión”: hay una por cada línea de trabajo, cuando los vehículos entran a la revisión, a partir del puesto 1, pasan de la “cola de espera” a la “cola de revisión”. Esto causa una salida de la “cola de espera” y una entrada en la “cola de revisión”. Toda la información del sistema debe ser persistente, es decir, se debe preservar la información actualizada de tal manera que pueda estar disponible en cada corrida del programa, para ello puede usar archivos. El programa tendrá un menú principal implementado con botones (widget Button) desde el cual se accederán las funciones del programa. Cada opción debe ofrecer la posibilidad de regresar al menú anterior preservando la información registrada. Usted define la interfaz gráfica siempre y cuando cumpla con los requerimientos funcionales del programa indicados seguidamente.

Temas Investigados

Software de control de versiones y de trabajo colaborativo

- **¿Qué es?**

Un software de control de versiones también conocido como VCS (Version control software) es una herramienta que monitoriza cambios en un sistema de archivos. Este ofrece herramientas de colaboración para compartir e integrar los cambios con otros usuarios del VCS.

- **Importancia de ese software en el desarrollo de software**

Este software es de gran importancia para el desarrollo de software ya que usualmente involucran más de un desarrollador. Mejora el flujo de trabajo en un trabajo colaborativo, permiten la comunicación entre los integrantes, permiten ver el desarrollo del software de una manera transparente y pueden monitorear el proyecto de forma eficiente. Usando este software se puede experimentar y probar varias cosas a través de ramas que son diferentes líneas de desarrollo. Permite aislar diferentes funciones de código para agregar cosas o debuggear. Luego de probarlas se pueden combinar con la rama principal, facilitando la integración continua dentro del proyecto.

- **Mencionar algunos softwares de ese tipo**

Algunos softwares de control de versiones son: Git, CVS, Apache Subversion, Mercurial y Monotone. Aunque todos cumplen con una misma función, es importante probar lo que ofrece el mercado para elegir el que mejor se adapte al flujo de trabajo requerido.

- **Explicar las características del software de este tipo usado en el proyecto**

Los VCS crean una instancia remota del repositorio para compartir los cambios entre desarrolladores. Este se puede alojar de forma externa lo cual significa que se guarda una copia del código fuente por si alguna casualidad ocurre. Mientras el VCS monitorea el sistema de archivos del código fuente, guarda un historial de los cambios y estados del código fuente. Con esta función es posible deshacer y revertir a una versión anterior por si ocurre un error. También es responsable en la resolución de conflictos entre varios desarrolladores ya que puede ser que tengan que trabajar en el código fuente al mismo tiempo.

Algoritmos para implementar el ABB

Árboles binarios de búsqueda son un tipo de datos abstracto que se conforman por un grupo de elementos llamados nodos. Estos se organizan de forma jerárquica, inicia en un nodo

raíz, tiene un nodo a la izquierda y otro a la derecha. A la izquierda van los que son menor al nodo raíz y a la derecha van los que son mayor.

Un algoritmo para implementar el ABB es el de inserción. Si el árbol está vacío, se crea un nuevo nodo y se establece como raíz. Para insertar un valor, se compara con el valor del nodo actual. Si el valor es menor que el del nodo actual, el algoritmo se desplaza al hijo izquierdo. Si el valor es mayor que el del nodo actual, el algoritmo se desplaza al hijo derecho. Este proceso continúa hasta que se encuentra un lugar vacío apropiado, donde se inserta el nuevo nodo.

El ABB también tiene un algoritmo de borrado. Este comienza en el nodo raíz y busca el nodo que debe borrarse sin perder de vista su nodo padre. Una vez encontrado el nodo, se consideran varios casos. Si el nodo no tiene hijos, simplemente se elimina del árbol. Si el nodo sólo tiene un hijo, éste se vincula al padre del nodo que se elimina. Si el nodo tiene dos hijos, el algoritmo busca el nodo con el siguiente valor mayor, que puede ser el valor mínimo del hijo derecho o el valor máximo del hijo izquierdo. A continuación, el nodo encontrado se sustituye por el nodo que se va a eliminar y, si es necesario, se repite el proceso de eliminación para el nodo encontrado.

Se usan algoritmos transversales para procesar los nodos del ABB en un orden específico. En el recorrido en orden, primero se visita el subárbol izquierdo, luego el nodo actual y, por último, el subárbol derecho. El recorrido pre-orden comienza con el nodo actual, luego visita el subárbol izquierdo y, por último, el subárbol derecho. El recorrido de orden posterior visita el subárbol izquierdo, luego el subárbol derecho y, por último, el nodo actual. El recorrido por niveles visita los nodos nivel por nivel, empezando por la raíz y moviéndose de izquierda a derecha.

Nuevos componentes en GUI usando tkinter

Listbox: Es un widget que despliega una lista de ítems de los cuales el usuario puede seleccionar. En el proyecto solo se puede seleccionar una opción a la vez ya que se usa para el tipo de vehículo.

Radiobutton: Es un widget que ofrece muchas opciones al usuario para seleccionar solamente 1. Cada grupo de radiobuttons tienen que asociarse con la misma variable y cada botón simboliza un único valor.

Combobox: elemento gráfico que cumple la función de los conocidos “dropdown-menus”. Funciona para hacer que el usuario elija dentro de una serie de opciones restringidas.

Diseño de la solución

Estructuras usadas

Clases: Las clases son una estructura de programación que permite definir un conjunto de métodos para un objeto. Permite agrupar datos y funciones a una sola unidad, facilita la organización del código y la modularidad. Además, favorecen la reutilización de código. Después de definir una clase se pueden crear múltiples instancias de esa misma clase, cada una con su funcionalidad y comportamiento. Esto permite reutilizar la funcionalidad de las clases en diferentes partes del proyecto. En el proyecto se usa una clase Nodo que contiene los datos de las citas. Se registran con la función de programar citas. La forma cómo se organizan los datos son a través de ABB. Dentro del ABB los datos se organizan usando fechas. La primera fecha de cita será el nodo raíz y dependiendo de la siguiente fecha puede ir a izquierda o a la derecha. Los datos van dentro de listas para agrupar las citas.

Árbol binario de búsqueda: El árbol binario de búsqueda es un tipo de dato abstracto que se conforman por un grupo de elementos llamados nodos. Se utiliza principalmente para buscar y recuperar datos de forma eficaz. Permite encontrar rápidamente una clave concreta realizando comparaciones en cada nodo, lo que reduce el espacio de búsqueda. Muchas operaciones con ABB, como la búsqueda y el recorrido, suelen realizarse mediante algoritmos recursivos. Estos algoritmos aprovechan la naturaleza recursiva del árbol para realizar operaciones de forma eficiente. En algunos casos, las estructuras de datos de pila o la recursividad pueden utilizarse para recorrer o realizar operaciones en un BST. Por ejemplo, pueden emplearse enfoques iterativos basados en pilas o métodos recursivos para implementar recorridos en orden, pre-orden o post-orden. Este proyecto en particular usa el ABB junto con recursión de pilas y clases. Se organiza el árbol con las fechas de las citas como mencionado anteriormente.

Recursión de pila: La recursividad de pila se utiliza para resolver problemas que pueden dividirse en instancias más pequeñas del mismo problema. Permite resolver un problema resolviendo recursivamente subproblemas más pequeños y combinando sus resultados. En este proyecto se relaciona con el árbol binario de búsqueda. Cuando se recorre un árbol de forma recursiva, la pila se puede utilizar para llevar la cuenta de los nodos o vértices que hay que visitar. La organización de la recursividad en la pila se basa en el principio LIFO (Last-In-First-Out) de la pila. La llamada recursiva más reciente y su correspondiente solución parcial se encuentran siempre en la parte superior de la pila, mientras que las llamadas anteriores están más abajo.

Archivos usados

Usamos dos archivos multimedia que corresponden al logo de la ventana y al del menú principal. Estos se mostrarán cuando se ejecuta el programa y otros menús subsecuentes.

Los archivos principales que usamos en el proyecto son los de las funcionalidades. Estas son abb, programar_citas, cancelar_citas, ingreso_citas, tablero_citas, fallas_citas, configuracion. Todos estos archivos son de Python. Ya que estamos trabajando en pareja, usamos un software de control de versiones. Dividimos las funcionalidades del proyecto en diferentes archivos porque hace más fácil la organización y monitorización. En lugar de tener un archivo que contiene todo el código, lo dividimos en múltiples partes y luego se importan al archivo principal. El archivo principal del proyecto se llama main y contiene el menú principal donde van todos los botones de las funcionalidades. Como mencionado anteriormente, los archivos de las funciones se importan a main y a otros módulos según sea necesario; un ejemplo de esto es el archivo abb.py, este script es importado en casi todos los módulos debido a que contiene las clases del Árbol Binario de Búsqueda y de sus nodos.

Todos estos archivos se relacionan entre ellos ya que algunos archivos requieren datos de otros. Un ejemplo es con la configuración y programar_citas, programar_citas hará validaciones de las fechas de acuerdo a la configuración del sistema. También cuando crea una lista de citas automáticas van a ser de acuerdo a la configuración. Este es el caso con todos los archivos de python ya que necesitan datos de la configuración o del árbol binario de búsqueda donde se registran todas las citas.

Toda aplicación necesita guardar datos, ya sean listas de fallas, nodos de un árbol binario, citas registradas o cualquier otra información que deba ser usada por todo el programa y que conviene almacenar, para suplir esta necesidad hacemos uso de los archivos .dat en formato JSON. Estos archivos son manejados dentro de Python por el módulo “json” y nos permiten guardar desde los nodos del árbol binario hasta las listas de fallas leves y graves. Como archivos .dat usamos: citas_abb.dat (almacena los nodos del Árbol Binario de Búsqueda), configuracion.dat (guarda la configuración del programa), estaciones.dat (mantiene un registro actualizado de la presencia en cada cola de espera y de revisión) y fallas.dat (diccionario con todos los tipos de fallas leves y graves que se pueden presentar).

Conclusiones

Cada proyecto plantea un nuevo desafío, algunos programas nos piden aprender cómo manejar ciertos tipos de datos, otros nos piden aprender a usar interfaces gráficas, el presente proyecto nos obligó a colaborar. Como programadores, debemos siempre trabajar en aras de un futuro laboral en el campo de la computación, debido a esto es crucial que sepamos hacer uso de herramientas de control de versiones.

Trabajar en un flujo colaborativo no es simplemente crear un repositorio, dos ramas y empezar a desarrollar, para realmente poder establecer un flujo colaborativo adecuado, se debe mantener un canal de comunicación abierto y constante entre todas las partes involucradas, así asegurándonos de que el desarrollo está siguiendo un camino en común. No es eficiente que dos desarrolladores comiencen a escribir código aislado, sin implementaciones comunes y con estructuras y/o recursos drásticamente diferentes.

Implementar una estructura de datos avanzada como lo es un Árbol Binario de Búsqueda puede parecer intimidante en un principio, mas haciendo uso de las herramientas que nos proporciona tanto el curso como el lenguaje de programación, podemos obtener un resultado aceptable. La estructuración de ciertas partes del código en clases simplifica la carga de trabajo y nos permite implementar una estructura compleja en poco tiempo.

La mayor dificultad presente en cada proyecto es empezar el proceso de desarrollo, lograr entender el planteamiento para poder implementarlo en código es una etapa clave del programa y de las que más tiempo toman. El proceso investigativo, de lectura y de aprendizaje previo es crucial para propiciar un trabajo a la altura de los requerimientos.

Una práctica muy importante dentro de un flujo de trabajo colaborativo es el compartir ideas y evacuar dudas entre los participantes. El tener a una persona mano a mano que está en el mismo proceso que nosotros, nos permite tener un mayor control en conjunto sobre el producto final.

Estadística de tiempos

Actividad realizada	Horas
Análisis de problema	6
Diseño de algoritmos	9
Investigación de…	11
Programación	23
Documentación interna	4
Pruebas	7
Elaboración del manual de usuario	3
Elaboración de documentación del proyecto	6
Corregir programa	8
TOTAL	77

Lista de revisión del proyecto

Concepto	Puntos	Puntos obtenidos	Avance 100/%/0	Análisis de resultados
Programación de citas implementando ABB con recursión	25		100%	
Asignación manual de citas	2		100%	
Asignación automática de citas	5		100%	
Cancelar citas	2		100%	
Ingreso de vehículos a la estación	5		100%	
Despliegue del tablero de revisión	10		100%	
Ejecución de los comandos del tablero	12		100%	
Registro de fallas por cita de revisión	5		100%	
Resultado de la revisión en PDF	5		100%	
Certificado de tránsito en PDF	3		100%	
Envío de correos electrónicos (cita, resultado de la revisión)	5		100%	
Lista de fallas (CRUD)	10		100%	
Configuración del sistema	6		100%	
Ayuda: manual de usuario en pantalla	5		100%	
TOTAL	100		100%	
Funcionalidades adicionales				

Bibliografía

Bitbucket. (s.f.). *Software de control de versiones: descripción general | Bitbucket*. https://bitbucket.org/product/es/version-control-software

GeeksforGeeks. (s.f.-a). *Deletion in Binary Search Tree (BST) - GeeksforGeeks*. https://www.geeksforgeeks.org/deletion-in-binary-search-tree/

GeeksforGeeks. (s.f.-b). *Searching in Binary Search Tree (BST) - GeeksforGeeks*. https://www.geeksforgeeks.org/binary-search-tree-set-1-search-and-insertion/

GeeksforGeeks. (s.f.-c). *Tree Traversal Techniques - Data Structure and Algorithm Tutorials - GeeksforGeeks*. https://www.geeksforgeeks.org/tree-traversals-inorder-preorder-and-postorder/

SEIDOR. (s.f.). *5 softwares de control de versiones*. Drauta SEIDOR. https://www.drauta.com/5-softwares-de-control-de-versiones