Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2022.

PROYECTO 1 – IMPLEMENTACIÓN DE MÉTODOS TDA PARA LA SOLUCIÓN DE MODIFICACIONES DE PATRONES DE PISOS PARA LA EMPRESA "PISOS ARSTESANALES, S.A"

202006692 – Josué Santa Cruz Ramírez

Resumen

El siguiente proyecto explica cómo se implementaron los conceptos de TDA, listas enlazadas y memoria dinámica para la solución del programa que nos pide la empresa "Pisos Artesanales, S.A.". Se desarrolló un robot que fuera capaz de crear pisos con diferentes patrones, y a la vez modificándolos sin la necesidad de crear unos nuevos. Y hablando sobre la modificación, el robot tiene que tener la capacidad de realizar una operación de volteo de azulejos y de intercambio de azulejos, cada uno con su diferente costo. El programa es apto para leer archivos con estructura XML, lo cual es más fácil para que el programa pueda leer los patrones, costos, etc.

Para la solución de este proyecto se utilizó datos de tipo (TDA), a la vez los conceptos de listas enlazadas simples como lo son los nodos e incluso la librería Graphviz, que permite que los patrones se visualicen de una mejor manera.

Palabras clave

- Listas Enlazadas
- Matrices
- XML
- Patrones
- Nodos

Abstract

The following project explains how the concepts of TDA, linked lists and dynamic memory were implemented for the solution of the program requested by the company "Pisos Artesanales, S.A.". A robot was developed that was capable of creating floors with different patterns, and at the same time modifying them without the need to create new ones. And speaking of modifying, the robot must have the ability to perform a tile flip and tile swap operation, each with its own different cost. The program is able to read files with XML structure, which is easier for the program to read patterns, costs, etc.

For the solution of this project, type data (TDA) was used, as well as the concepts of simple linked lists such as nodes and even the Graphviz library, which allows patterns to be visualized in a better way.

Keywords

- *Linked Lists*
- *Matrix*
- *XML*
- *Patterns*
- *Nodes*

Introducción

En la programación, una estructura de datos es una forma especial de organizar datos de un ordenador para que más adelante se pueda usar de manera eficiente. Existen muchos tipos de estructuras de datos adecuados para diferentes tipos de aplicaciones, algunas de las cuales están altamente especializadas para tareas específicas.

Generalmente, estos juegan un rol importante al momento de diseñar algoritmos de forma eficiente. En algunos lenguajes de programación destacan las estructuras de datos en lugar de algoritmos como el factor clave de organización. Son útiles porque permiten tener una batería de herramientas para solucionar distintos problemas. Además, nos permiten hacer un software más eficiente optimizando recursos, algo muy útil para los entornos que trabajan con Big Data y el internet de las cosas, que va avanzado exponencialmente.

Existen diversos tipos de estructura de datos, pero este proyecto se enfoca en la implementación de listas para la solución de esta.

Desarrollo del tema

Para entrar más a detalle sobre que son las listas enlazadas, primero se tiene que definir que es una lista. Una lista es una estructura dinámica muy importante en la programación de datos que contiene una colección de elementos ordenados con la característica que todos los elementos son homogéneos (del mismo tipo).

Las listas enlazadas contienen las mismas características que poseen las listas, con la única diferencia de que su estructura es dinámica. Esto quiere decir que el número de nodos de la lista no es fijo y puede ampliarse y reducirse según sea necesario. Cualquier programa que necesite manejar un número desconocido de elementos deberá de implementar una lista enlazada para ir almacenando los valores, como por ejemplo una cola de impresión de una empresa.

Una desventaja de esto en comparación de una matriz es que no permite el acceso directo a un elemento individual, si no lo que se debe hacer es comenzar por la cabecera o inicio de la cola y seguir las referencias hasta que se llegue a él.

Existen dos tipos de listas enlazadas, las simples y doblemente enlazadas, que a continuación se presenta la diferencia.

a. Lista enlazada simple:

Las listas enlazadas son estructuras de datos que enlazan los elementos a través de un puntero. Son muy similares a los arreglos, con la única diferencia que el acceso a un dato no se hace mediante un índice.

Aquí todos los elementos apuntan al nodo que le sigue, excepto del último dato que no contiene nada para indicar el final de la lista. Al primer elemento de la lista se le suele llamar como cabecera o en inglés "*header*".

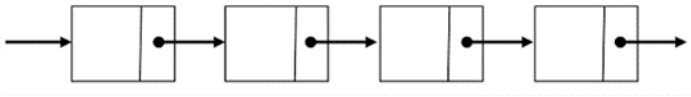

Figura 1. Ejemplo de una lista enlazada simple

Fuente: López Takeyas, B. 2022, Instituto Tecnológico Nuevo Laredo

b. Lista doblemente enlazada:

Este tipo de lista enlazada consiste es un conjunto de nodos enlazados secuencialmente. Cada no tiene dos enlaces, que son referencias al nodo anterior y al

siguiente. Este doble enlace permite recorrer la lista en cualquier dirección, a diferencia de la enlazada simple que solo podía ir en una. Aquí son más simples las operaciones porque no hay necesidad de guardar el puntero.

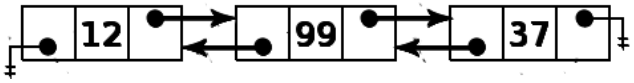

Figura 1I. Ejemplo de una lista doblemente enlazada

Fuente: Anónimo. 2022, Wikipedia

Problema:

La empresa "Pisos Artesanales, S.A." ha construido un azulejo especial con el que puede crear pisos con distintos patrones. Cada piso consiste en una matriz de R filas y C columnas de azulejos cuadrados. Cada azulejo es reversible, un lado es blanco y el otro es negro, para poder crear patrones diversos. Además, la empresa garantiza que, para los pisos ya instalados, podrá cambiar el patrón original por un nuevo patrón que el cliente desee sin necesidad de comprar nuevos azulejos.

La empresa ha comprado un robot especializado capaz de colocar los pisos de dimensiones R x C con cualquier patrón, combinando azulejos del lado blanco con azulejos del lado negro. Además, la empresa garantiza que cada piso colocado podrá cambiar el patrón, a cualquier patrón deseado siguiendo las siguientes reglas: Para cambiar el patrón original del piso colocado por "Pisos Artesanales, S.A.", se debe cumplir que el nuevo patrón corresponda a las dimensiones R x C del piso colocado y el robot será capaz de realizar una de las siguientes operaciones con cada uno de los azulejos que componen el piso:

1. Voltear un azulejo, cambiando el color visible de blanco a negro o viceversa, y
2. Intercambiar dos azulejos adyacentes (horizontal o verticalmente, pero no en diagonal), sin voltear ninguno.

A la empresa "Pisos Artesanales, S.A." le resulta en un costo que el robot realice cada una de las operaciones antes mencionadas, de tal manera que, realizar una operación de volteo de azulejo cuesta F Quetzales, mientras que realizar una operación de intercambio de azulejos cuesta S Quetzales. Debido a que la empresa "Pisos Artesanales, S.A." posee una cuota fija pactada por contrato con los clientes que han colocado los pisos que ofrecen, se le ha solicitado realizar un programa que garantice que, al modificar un patrón en un piso existente, el costo de hacer esta modificación sea el mínimo posible para optimizar el uso del robot especializado adquirido para este fin.

Para poder solucionar el problema se debió de usar el lenguaje de Python. Además, se implementaron dos librerías, "miniDom" y "Graphviz. "nimi Dom" sirvió para que el programa fuera capaz de leer, escribir en archivos de tipo XML y poder pasar la información en listas enlazadas, ya que este es el tema del proyecto. Y la librería Graphviz permitió que se pudieras visualizar los patrones de los pisos de forma gráfica y ordenada. Para el proceso aplicamos todo el conocimiento adquirido sobre la programación orientada a objetos. Se crearon varias clases para poder tener una mejor estructura y adicionalmente carpetas en donde estaban divididos.

Clases:

- **Main**: Clase principal del programa en donde se implementaba toda la interfaz del usuario,

en donde se muestran todos los menús e instrucciones a través de consola

- **Carpeta Listass**: Aquí se almacenan todas las clases de las listas a utilizar
    - Pisos
    - Patrones
    - Cuadritos
- **Carpeta Nodos:** Aquí se almacenan todas las clases de los nodos
    - NodoCuadro
    - NodoPatron
    - NodoPiso
- **Lista:** Clase done se guardan los patrones
- Cuadro
- Patron
- Piso

Métodos:

- **Carga:** Función que da el mensaje de bienvenido y permite la carga del archivo XML al programa.
- **MenuPrincipal:** Esta función se llama si se cargó exitosamente el archivo y además permite las funciones de seleccionar patrón y código del piso que se quiere trabajar o regresar a escoger un nuevo archivo.
- **MenuPisos**: Esta función se llama únicamente si se ha seleccionado un código y patrón del archivo, en donde el usuario tendrá la opción de mostrar el patrón, modificar y regresar al menú anterior.
- **ImprimirLista:** Permite la impresión completa de la lista a través de consola
- **buscarImprimir:** Función que se llama si se escoge la función de seleccionar patrón en MenuPrincipal que busca el patrón y código según haya ingresado el usuario.
- **guardarPatron:** Permite guardar el patrón seleccionado para luego imprimirlo de forma gráfica.

Conclusiones

- Las estructuras de datos son un medio para manejar grandes cantidades de datos de manera eficiente.
- Una lista doblemente enlazada permite recorrer una estructura de datos de forma eficaz ya que se puede en cualquier dirección, con la desventaja del uso de nodos, que pueden confundir si no se saben implementar de forma correcta.
- El uso de listas enlazadas puede optimizar el uso de memoria.
- Es importante conocer e implementar diferentes librerías que Python ofrece ya que permite un mejor manejo y entendimiento a lo que se requiere el programa.

Referencias bibliográficas

Anónimo. (2022). *Lista Doblemente Enlazada*. Wikipedia Sitio web: https://es.wikipedia.org/wiki/Lista_doblemente_enlazada

C. J. Date, (1991). *An introduction to Database Systems.* Addison-Wesley Publishing Company, Inc.

Joyanes Aguilar, Luis. (2008). *Fundamentos de programación*. Madrid: McGraw Hill.