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 ROBOT DE COLOCACIÓN DE AZULEJOS

202000510 – Javier Alexander Girón Donis

Resumen

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.

Este algoritmo tiene que ser capaz de tomar los patrones ingresados y así el robot debe de elegir el mejor cambio para conseguir el patrón deseado, ocupando la menor cantidad de movimientos para eso. La forma en que este se moverá únicamente es: arriba, abajo, izquierda, derecha y volteo. Por último, la elaboración del algoritmo se hará en el lenguaje de programación Python

Palabras clave

Nodo: es cada uno de los elementos de una lista enlazada, un árbol o un grafo en una estructura de datos.

XML: Es un metalenguaje que fue diseñado para estructurar, almacenar e intercambiar datos entre aplicaciones.

POO: Programación Orientada Objetos

Dijkstra: algoritmo para la determinación del camino más corto.

TDA: Tipo de dato abstracto

Abstract

The company "Pisos Artesanales, S.A." has built a special tile with which you can create floors with different patterns. Each floor consists of an array of R rows and C columns of square tiles. Each tile is reversible, one side is white and the other is black, to create different patterns.

This algorithm has to be able to take the entered patterns and thus the robot must choose the best change to achieve the desired pattern, occupying the least amount of movements for that. The way it will only move is: up, down, left, right and flip. Finally, the elaboration of the algorithm will be done in the Python programming language

Keywords

Node: is each of the elements of a linked list, a tree, or a graph in a data structure.

XML: It is a metalanguage that was designed to structure, store and exchange data between applications.

OOP: Object Oriented Programming

Dijkstra*: algorithm for the determination of the shortest path.*

ADT/TDA: Abstract data type

Introducción

Este proyecto trata de la elaboración de un algoritmo para optimizar los cambios que se deseen realizar a los patrones ingresados, dichos patrones se ingresaran por medio de un archivo en formato XML, al comparar dichos patrones con el deseado, el algoritmo empleará la mejor manera de conseguir el resultado esperado.

Python

Para entender la forma en la cual Python utiliza la memoria dentro de un desarrollo se debe conocer y aclarar todo referente al mismo. Es un lenguaje de programación orientado a objetos, con sintaxis sencilla cuya principal filosofía es que sea legible por cualquier persona con conocimientos básicos de programación.

Dentro de este se pueden realizar variadas Aplicaciones Web, Automatización y secuencias de comandos, Ciencia de datos y aprendizaje automático. Lo que lo convierte en un lenguaje multiparadigma. Esto significa que combina propiedades de diferentes paradigmas de programación, lo que permite que sea muy flexible y fácil de aprender de manera independiente de los conocimientos del interesado.

Al ser un lenguaje Open Source / Codigo Abierto, es decir que no necesitamos el pago de licencias para su uso, hace que se creen múltiples librerías y aplicaciones constantemente además de un gran respaldo de parte de la comunidad para la resolución de dudas.

Memoria Dinámica

La memoria dinámica que se almacena en el *heap* es aquella que se utiliza para almacenar datos que se crean en el medio de la ejecución de un programa. En general, este tipo de datos puede llegar a ser casi la totalidad de los datos de un programa.

TDA (Tipo Dato Abstracto)

Un tipo abstracto de datos es un tipo definido por el usuario que:

- Tiene un conjunto de valores y un conjunto de operaciones. - Cumple con los principios de abstracción, ocultación de la información y se puede manejar sin conocer la representación interna.

• Es decir, los TDA ponen a disposición del programador un conjunto de objetos junto con sus operaciones básicas que son independientes de la implementación elegida

Archivos XML

XML es el acrónimo de Extensible Markup Language, es decir, es un lenguaje de marcado que define un conjunto de reglas para la codificación de documentos.

El lenguaje de marcado es un conjunto de códigos que se pueden aplicar en el análisis de datos o la lectura de textos creados por computadoras o personas. El lenguaje XML proporciona una plataforma para definir elementos para crear un formato y generar un lenguaje personalizado.

Un archivo XML se divide en dos partes: *prolog* y *body*.

La parte *prolog* consiste en metadatos administrativos, como declaración XML, instrucción de procesamiento opcional, declaración de tipo de documento y comentarios.

La parte del *body* se compone de dos partes: estructural y de contenido (presente en los textos simples).

Listas Enlazadas

Este tipo de estructura es de forma lineal y permite almacenar elementos comúnmente llamados nodos, en donde cada uno de ellos puede almacenar distintos datos y un apuntador hacia otros nodos.

Son estructuras dinámicas que pueden almacenar datos que cambian con frecuencia. Las listas enlazadas se propagan y se retraen para que sean más sencillas al añadir o eliminar información y permiten almacenar información en diferentes posiciones de memoria

Figura 1. Lista Simple

Fuente: elaboración propia

Lista Doble Enlazada

Las listas doblemente enlazadas son estructuras de datos semejantes a las listas enlazadas simples, pero algo más complejas. La asignación de memoria se realiza en el momento de la ejecución.

En cambio, en relación con la lista enlazada simple el enlace entre los elementos se hace gracias a dos punteros (uno que apunta hacia el elemento anterior y otro que apunta hacia el elemento siguiente):

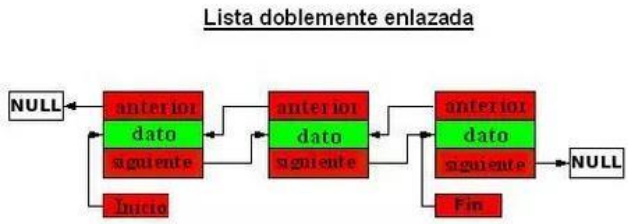

Figura 2. Lista Doble

Fuente: elaboración propia

El puntero anterior del primer elemento debe apuntar hacia NULL (el inicio de la lista).

El puntero siguiente del último elemento debe apuntar hacia NULL (el fin de la lista).

Para acceder a un elemento, la lista puede ser recorrida en ambos sentidos: comenzando por el inicio, el puntero siguiente permite el desplazamiento hacia el próximo elemento; comenzando por el final, el puntero anterior permite el desplazamiento hacia el elemento anterior.

Resumiendo: el desplazamiento se hace en ambas direcciones, del primer al último elemento y/o del último al primer elemento.

Algoritmo Dijkstra

También llamado algoritmo de caminos mínimos es un algoritmo para la determinación del camino más corto dado un vértice origen al resto de vértices en un grafo con pesos en cada arista. Su nombre se refiere a Edsger Dijkstra, quien lo describió por primera vez en 1959. **Aplicaciones**

En múltiples aplicaciones donde se aplican los grafos, es necesario conocer el camino de menor costo entre dos vértices dados:

- Distribución de productos a una red de establecimientos comerciales.
- Distribución de correos postales.
- Sea G = (V, A) un grafo dirigido ponderado.

El problema del camino más corto de un vértice a otro consiste en determinar el camino de menor costo, desde un vértice u a otro vértice v. El costo de un camino es la suma de los costos (pesos) de los arcos que lo conforman.

Proyecto 1

Para este proyecto se utilizaron dos clases llamadas: App y Algoritmo. Cada clase mencionada contaba con su método constructor con las variables necesarias para cada una de ellas.

En la clase app, se encontraba el menú del sistema el cual cuenta con 5 funcionalidades las cuales son:

1) Leer Archivo, 2) Opciones, 3) Menú y 4) Graphviz

En Leer Archivo, se debe introducir un archivo .xml a través de una ventana emergente para poder determinar la trayectoria del robot y guardamos esta información en nuestro sistema.

En Opciones, daremos inicio al programa donde desplegaremos el menú inicial y con ello ejecutaremos la función de Leer archivo para que el usuario seleccione su archivo de entrada

En Menú podremos partir en las diferentes funcionalidades que cuenta el programa, podemos ingresar si deseamos “Mostrar Gráficamente un piso”, “Cambiar un tipo de piso”, “Mostrar todos los pisos” y “Salir”. Cabe mencionar que si no se selecciona una opción correcta el menú se volverá a imprimir en consola, creando un bucle donde sea necesario tomar una opción correcta.

En Graphviz contamos con una función importante del programa ya que con ella podremos graficar el piso que deseemos, en esta funcionalidad se ingresará el nombre del piso que deseamos y el patrón que pertenezca a dicho nombre. Con eso se imprimirá una imagen la cual estará en formato jpg.

Conclusiones

- Haciendo uso de la POO en este algoritmo hace más ordenado el proceso de programación ya que cada clase que use tendrá sus propios atributos y métodos que puedo usar en distintas fases de mi programa, consiguiendo una mejor optimización en memoria y evitar sobrescribir el código.

-Con el algoritmo de Dijkstra se puede concluir que es bastante eficiente debido a que nos retorna

siempre el nodo con el camino más corto y los movimientos optimizados.

- La utilización de estructuras dinámicas como TDA permite que el programa se ejecute con la memoria necesaria y no afecte su rendimiento, además de permitir la separación de los datos en nodos que guardan la información necesaria, creando una Lista enlazada simple para almacenar los pisos y asociando a cada nodo de esta otra lista (Lista Doble enlazada)

De esta manera se cuenta con una estructura con una lista en una sola dirección que contiene información de los patrones y una lista con apuntadores en dos direcciones que permiten recorrer las coordenadas hacia adelante.

Referencias bibliográficas

1. Algoritmo de Dijkstra Joemmanuel Ponce Galindo
   Mexico
2. Escuela Politécnica Superior de México. GUÍA DOCENTE DE PROGRAMACIÓN ORIENTADA A OBJETOS
3. Estructura de datos Tipos Abstractos de Datos(TADs) Prof. Montserrat Serrano Montero
   Universidad de Valladolid
4. TORANZOS, Fausto. Introducción a la Teoría de Grafos. OEA. 1976

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.

Anexos:

Diagrama De Clases