|  |
| --- |
| **PROYECTO No1 | IPC - 2 | “PISOS DE GUATEMALA S.A”** |
| **202200389 – Juan José Rodas Mansilla** |

**Resumen**

La empresa "Pisos de Guatemala, S.A." desea desarrollar un sistema de azulejos reversibles para crear pisos con diversos patrones. Cada piso consiste en una matriz de azulejos cuadrados de dimensiones R por C.

Garantizan la capacidad de cambiar el patrón original sin necesidad de comprar nuevos azulejos. Para realizar esta modificación, se deben respetar las dimensiones del piso y se pueden realizar dos operaciones por azulejo: voltear el color o intercambiarlo con otro adyacente horizontal o verticalmente.

La empresa tiene un costo asociado a estas operaciones: F Quetzales para voltear un azulejo y S Quetzales para intercambiarlos. Debido a un contrato, necesitan minimizar el costo de modificar los patrones.

Debido a que se posee una cuota fija, se le ha solicitado realizar un programa que garantice que, al modificar un patrón en un piso existente, el costo de modificación sea el mínimo para optimizar el uso para este fin.

**Palabras clave**

Reversible, patrón, matriz.

***Abstract***

The company "Pisos de Guatemala, S.A." wishes to develop a system of reversible tiles to create floors with various patterns. Each floor consists of a matrix of square tiles with dimensions R by C.

They guarantee the ability to change the original pattern without needing to buy new tiles. To make this modification, the floor dimensions must be respected, and two operations per tile can be performed: flipping the color or exchanging it with an adjacent one horizontally or vertically.

The company incurs a cost associated with these operations: F Quetzales to flip a tile and S Quetzales to exchange them. Due to a contract, they need to minimize the cost of modifying the patterns.

Because there is a fixed quota, they have been asked to create a program that ensures that when modifying a pattern in an existing floor, the modification cost is minimized to optimize its use for this purpose.

***Keywords***

Reversible, pattern, matrix.

**Introducción**

Se tiene como objetivo desarrollar un programa en Python para gestionar de manera eficiente los patrones de azulejos reversibles dentro de matrices de pisos.

Estas matrices, compuestas por azulejos cuadrados de dimensiones R por C, ofrecen flexibilidad al permitir modificaciones sin necesidad de adquirir nuevos azulejos. El programa garantiza el cumplimiento de las dimensiones del piso y ofrece dos operaciones fundamentales por azulejo: voltear el color y cambiarlo por otro adyacente horizontal o verticalmente.

Estas operaciones conllevan costos, representados por F Quetzales para voltear un azulejo y S Quetzales para intercambiarlos. Dado que la optimización de costos es primordial debido a obligaciones contractuales y cuotas fijas, la empresa busca una solución en Python que minimice los gastos al modificar eficazmente los patrones de pisos existentes. Esta introducción establece el escenario para comprender el objetivo de la empresa de simplificar las modificaciones de patrones mediante un programa en Python optimizado.

**Desarrollo del tema**

Trabajar con listas enlazadas, matrices, nodos y automatizaciones en Python ofrece una amplia gama de posibilidades para la manipulación eficiente de datos y la automatización de tareas. Comenzando con las listas enlazadas, estas estructuras de datos son colecciones lineales donde cada elemento, llamado nodo, contiene un valor y un puntero al siguiente nodo. En Python, las listas enlazadas pueden implementarse mediante clases y referencias de objetos, lo que permite una manipulación dinámica de los datos, ideal para situaciones donde el tamaño de la lista puede variar dinámicamente.  
La programación orientada a objetos (POO) es un paradigma de programación que se basa en el concepto de "objetos", los cuales son entidades que pueden tener atributos (datos) y métodos (funciones). La creación de clases es un aspecto fundamental de la POO, ya que las clases son plantillas o moldes que definen la estructura y el comportamiento de los objetos.

A. Matrices: son estructuras bidimensionales que consisten en filas y columnas, ideales para representar datos tabulares o mallas de información. En Python, las matrices pueden implementarse utilizando listas de listas, lo que proporciona flexibilidad para trabajar con datos multidimensionales.

B. Nodos: son elementos fundamentales en estructuras como las listas enlazadas y los árboles. Cada nodo contiene un valor y referencias a otros nodos, permitiendo la construcción de estructuras de datos complejas como árboles binarios o gráficos. En Python, los nodos pueden representarse mediante clases, lo que facilita su creación y manipulación.

C. Automatización: en Python implica el desarrollo de programas que realizan tareas repetitivas de manera automática. Al combinar el manejo de listas enlazadas, matrices, y nodos con las capacidades de automatización de Python, es posible crear sistemas que procesen grandes cantidades de datos de manera eficiente y efectiva, optimizando procesos y aumentando la productividad.

D. Clases: en Python se define utilizando la palabra clave class, seguida del nombre de la clase y dos puntos (:). Dentro de la clase, se definen los atributos y métodos que poseerán los objetos creados a partir de esa clase. Los atributos son variables que representan datos asociados a los objetos, mientras que los métodos son funciones que definen el comportamiento de los objetos.

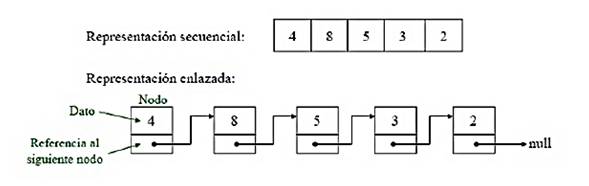
trabajar con listas enlazadas, matrices, nodos y automatizaciones en Python ofrece herramientas poderosas para la manipulación y procesamiento de datos, permitiendo la creación de sistemas sofisticados y eficientes para una variedad de aplicaciones.

Además de los nodos básicos, es importante comprender otras variantes y aplicaciones de las estructuras de datos en Python. Las listas enlazadas circulares son similares a las listas enlazadas estándar, pero el último nodo apunta de nuevo al primero, formando un ciclo. Este tipo de estructura es útil en situaciones donde se necesita un acceso rápido a los extremos de la lista y se desea evitar recorrerla completamente.

Las listas enlazadas dobles añaden un puntero adicional a cada nodo, permitiendo la navegación en ambas direcciones (hacia adelante y hacia atrás). Esto facilita operaciones como la inserción o eliminación de nodos en cualquier posición de la lista, lo que aumenta la versatilidad de la estructura de datos.

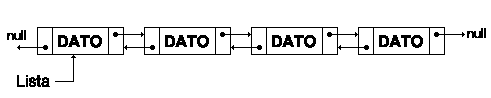
Otras variantes de listas enlazadas incluyen las listas enlazadas con cabecera, donde se agrega un nodo adicional al principio de la lista para simplificar la manipulación, y las listas enlazadas con punteros saltados, donde cada nodo tiene un puntero que apunta a un nodo no contiguo en la lista, lo que permite una búsqueda más eficiente en estructuras de datos grandes.

El uso de estas estructuras de datos en combinación con las capacidades de automatización de Python permite la creación de sistemas complejos y eficientes. Por ejemplo, se pueden desarrollar algoritmos para ordenar listas enlazadas, buscar elementos específicos o realizar operaciones matemáticas en matrices, todo de manera automatizada y optimizada.

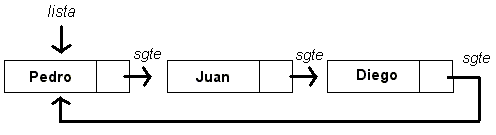


*Figura 1.* Representación Listas Enlazadas

Fuente: monografías.com (2015). Fundamentos de Algoritmos y Programación. Recuperado de https://www.monografias.com/trabajos101/las-istas-enlazadas/las-istas-enlazadas



Fuente: conclase.com (2002). Ejemplo de lista doblemente enlazada con plantillas. Recuperado de https://www.conclase.net/c/edd/cap5



Fuente: blogs.ead.unlp.edu.ar (2012). Lista Circular Pura. Recuperado de https://blogs.ead.unlp.edu.ar/programacion2/tag/listas-circulares/

*Tabla de Funciones principales:*

|  |  |
| --- | --- |
| **CATEGORÍA** | **CATEGORÍA** |
| VARIABLE | XXXXXXXX |
| VARIABLE | XXXXXXXX |
| VARIABLE | XXXXXXXX |
| VARIABLE | XXXXXXXX |
| VARIABLE | XXXXXXXX |

Fuente: elaboración propia, o citar al autor, año y página.

Es conveniente describir brevemente el contenido de una tabla, evitando los aspectos obvios.

En el caso de inclusión de fórmulas, éstas deben elaborarse utilizando el editor de ecuaciones disponible en Word, indicando el significado de cada una de las variables o parámetros que se incluyen.

Deben enumerarme entre paréntesis para poder hacer referencia de esta. Por ejemplo, un modelo de crecimiento exponencial

 (1)

donde:

y = cantidad presente en el tiempo t

yo =cantidad presente al inicio de la observación

k = tasa específica de crecimiento

t = periodo de tiempo (años, minutos, otros)

**Conclusiones**

* Trabajar con nodos, listas enlazadas circulares, listas enlazadas dobles y otras variantes de listas enlazadas en Python proporciona herramientas poderosas para el manejo eficiente de datos y la automatización de tareas, lo que permite desarrollar sistemas sofisticados y efectivos para una variedad de aplicaciones.
* El programa desarrollado en Python para manipular listas enlazadas, matrices, nodos y automatizaciones ofrece una solución versátil y eficiente para gestionar datos y tareas de manera automatizada.
* La combinación de estructuras de datos dinámicas como las listas enlazadas con punteros, matrices bidimensionales y nodos permite una manipulación flexible y escalable de la información. Además, la capacidad de automatización de Python facilita la implementación de algoritmos y procesos que optimizan el uso de recursos y aumentan la productividad.
* En conjunto, este programa proporciona una herramienta valiosa para resolver una amplia gama de problemas de manipulación de datos y automatización de tareas en diferentes contextos y aplicaciones. Su versatilidad y eficiencia lo convierten en una herramienta indispensable para desarrolladores y profesionales que buscan optimizar sus procesos y mejorar su productividad en el ámbito de la programación.
* La programación de objetos y la creación de clases en Python proporcionan una manera organizada y modular de estructurar y trabajar con datos y funciones relacionadas. Al utilizar clases y objetos, se pueden encapsular datos y comportamientos relacionados en entidades cohesivas, lo que facilita la escritura de código claro, reutilizable y mantenible.

**Referencias bibliográficas**

* Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.).
* C. J. Date, (1991). An introduction to Database Systems. Addison-Wesley Publishing Company, Inc.
* Fundamentos de Algoritmos y Programación (2015). Recuperado de <https://www.monografias.com/trabajos101/las-istas-enlazadas/las-istas-enlazadas>
* Lista Circular Pura (2012). Recuperado de <https://blogs.ead.unlp.edu.ar/programacion2/tag/listas-circulares/>