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.

PROGRAMA: CAMBIO EN PATRONES DE PISOS

202004763 – Cristian Noé Axpuac Aspuac

Resumen

El programa desarrollado se basa en la lectura de un archivo XML, en la implementación de listas enlazadas simples y en la optimización del costo de cambios entre patrones con el objetivo de dirigir un robot poseído por la empresa ‘Pisos Artesanales S.A.’ y su finalidad es el ordenamiento de azulejos de forma optima para sus clientes. Este robot es capaz de ordenar azulejos con R filas y C columnas, además priorizara entre acciones entre voltear e intercambiar azulejos priorizando aquellos que tengan el menor precio y mientras el precio de intercambiar sea menor se priorizara esta acción con la finalidad de evitar costos innecesarios.

Palabras clave

Optimización, cambiar posición, clase, lista, archivo xml.

Abstract

The developed program is based on the reading of an XML file, on the implementation of simple linked lists and on the optimization of the cost of changes between patterns with the aim of directing a robot owned by the company 'Pisos Artesanales SA' and its purpose is the ordering of tiles in an optimal way for your customers. This robot is capable of ordering tiles with R rows and C columns, it will also prioritize actions between flipping and exchanging tiles, prioritizing those that have the lowest price and while the price of exchanging is lower, this action will be prioritized in order to avoid unnecessary costs.

Keywords

Optimization, change position, class, list, xml file.

Introducción

El programa desarrollado utiliza programación orientada a objetos con la finalidad de implementar listas simples enlazadas para el desarrollo de la optimización y traficación de los patrones solicitados mediante un archivo con extensión XML. El programa cuenta con funciones y métodos para la lista enlazada simple implementada y esta lista fue creada a partir de una clase llamada 'Node' la cual utiliza apuntadores con la finalidad de indicar la posición en la lista en la que se encuentra un dato y de esta manera poder extraerlo y almacenarlo cuando se requiera su utilizar.

Desarrollo del tema

Clase: Node():

Esta clase fue utilizada para implementar nodos con atributos data y siguiente, de esta forma se pudo implementar la lista enlazada dado que mediante el ciclo

```
For i in range(0, self.cantidad_datos():
    Self.root.siguiente
```

Donde self.root es el atributo nodo en la lista, se utiliza el ciclo for, se delimita el rango por range(0, self.cantidad_datos() y se recorre por i.

Definición de Nodo:

En estructuras de datos dinámicas un nodo es un registro que contiene un dato de interés y al menos un puntero para referenciar (apuntar) a otro nodo. Si la estructura tiene solo un puntero, la única estructura que se puede construir con él es una lista, y si el nodo tiene más de un puntero se pueden construir estructuras más complejas como árboles o grafos.

Clase: Lista():

Esta es la clase donde mayor cantidad de procesos se llevan a cabo, esta clase construye una lista enlazada simple en la cual se almacena toda la información del archivo xml y además en ellas se llevan a cabo procesos secundarios llamados por la Clase: MAIN().

Definición de Lista simple:

En ciencias de la computación, una lista enlazada es una de las estructuras de datos fundamentales, y puede ser usada para implementar otras estructuras de datos. Consiste en una secuencia de nodos, en los que se guardan campos de datos arbitrarios y una o dos referencias, enlaces o punteros al nodo anterior o posterior. El principal beneficio de las listas enlazadas respecto a los vectores convencionales es que el orden de los elementos enlazados puede ser diferente al orden de almacenamiento en la memoria o el disco, permitiendo que el orden de recorrido de la lista sea diferente al de almacenamiento.

Una lista enlazada es un tipo de dato autorreferenciado porque contienen un puntero o enlace (en inglés link, del mismo significado) a otro dato del mismo tipo. Las listas enlazadas permiten inserciones y eliminación de nodos en cualquier punto de la lista en tiempo constante (suponiendo que dicho punto está previamente identificado o localizado), pero no permiten un acceso aleatorio. Existen diferentes tipos de listas enlazadas: listas enlazadas simples, listas doblemente enlazadas, listas enlazadas circulares y listas enlazadas doblemente circulares.

Clase: graficar():

Esta clase es utilizada con el objetivo de mostrar gráficamente el patrón ingresado, recibe como parámetros el código del patrón a graficar y el patrón a graficar. Utiliza como motor la librería graphviz la cual se basa en un sistema de nodos con el objetivo de generar una gráfica enlazada.

Graphviz:

Instalación:

graphviz proporciona una interfaz simple de Python puro para el software de dibujo de gráficos

Graphviz. Se ejecuta bajo Python 3.6+. Para instalarlo con pip, ejecuta lo siguiente:

$ pip install graphviz

Para una instalación en todo el sistema, esto generalmente requiere acceso de administrador. Para una instalación aislada, puede ejecutar lo mismo dentro de a venvo virtualenv.

La única dependencia es una instalación funcional de Graphviz (página de descarga, versiones archivadas, procedimiento de instalación para Windows).

Después de instalar Graphviz, asegúrese de que su bin/subdirectorio que contiene el dot comando de diseño para representar descripciones de gráficos esté en sus sistemas PATH (a veces lo hace el instalador; configuración PATH en Linux, Mac y Windows): en la línea de comandos, debe imprimir la versión de su instalación de Graphiz.dot -V

Uso:

El paquete graphviz proporciona dos clases principales: graphviz.Graph y graphviz.Digraph. Crean descripciones de gráficos en el lenguaje DOT para gráficos dirigidos y no dirigidos, respectivamente. Tienen la misma API .

Lenguaje Dot:

La siguiente es una gramática abstracta que define el lenguaje DOT. Los terminales se muestran en negrita y los no terminales en cursiva. Los caracteres literales se dan entre comillas simples. Paréntesis (e) indicar agrupación cuando sea necesario. Corchetes [y] encierre elementos opcionales. Las barras verticales |separan las alternativas.

Las palabras clave nodo , borde , gráfico , dígrafo , subgráfico y estricto son independientes de mayúsculas y minúsculas. Tenga en cuenta también que los valores de puntos de la brújula permitidos no son palabras clave, por lo que estas cadenas se pueden usar en otros lugares como identificadores ordinarios y, a la inversa, el analizador aceptará cualquier identificador.

Una identificación es una de las siguientes:

Cualquier cadena de [a-zA-Z\200-\377]caracteres alfabéticos ( ), guiones bajos ('_') o dígitos ([0-9]), que no comience con un dígito;
a numeral [-]?(.[0-9]$^{+}$ | [0-9]$^{+}$(.[0-9]*)?);
cualquier cadena entre comillas dobles ("...") que posiblemente contenga comillas escapadas (\")[1];
una cadena HTML (<...>).

Una ID es solo una cadena; la falta de caracteres de comillas en las dos primeras formas es solo por simplicidad. No hay diferencia semántica entre abc_2y "abc_2", o entre 2.34y "2.34". Obviamente, para usar una palabra clave como ID, debe estar entre comillas. Tenga en cuenta que, en las cadenas HTML, los corchetes angulares deben aparecer en pares coincidentes, y se permiten nuevas líneas y otros caracteres de espacio en blanco de formato. Además, el contenido debe ser XML legal, por lo que las secuencias de escape XML especiales para ", &, < y > pueden ser necesarias para incrustar estos caracteres en valores de atributo o texto sin formato. Como ID, una cadena HTML puede ser cualquier cadena XML legal. Sin embargo, si se usa como un atributo de etiqueta, se interpreta de manera especial y debe seguir la sintaxis de las etiquetas similares a HTML .

Conclusiones

Esta sección debe orientarse a evidenciar claramente las principales ideas generadas, propuestas que deriven del análisis realizado y si existen, expresar las conclusiones o aportes que autor quiera destacar.

Enfatizando, lo importante es destacar las principales posturas fundamentadas del autor, que desea transmitir a los lectores.

Adicionalmente, pueden incluirse preguntas abiertas a la reflexión y debate, temas concatenados con el tema expuesto o recomendaciones para profundizar en la temática expuesta.

Referencias bibliográficas

Graphviz, (27 de enero del 2022). *Dot Languaje* https://www.graphviz.org/doc/info/lang.html

National Institute of Standards and Technology (16 de Agosto del 2004). Definition of a linked list. https://xlinux.nist.gov/dads/HTML/linkedList.html

Anexos

Clase NODE():

```
class node:
    def __init__(self, data = None, siguiente = None):
        self.data = data
        self.siguiente = siguiente
```

Funciones de la Clase MAIN():

```
class main:
    def menu(self):···

    def menu2(self):···

    def leerArchivo(self):···
    def voltear(self, x):···

    def optimizacion(self, COD1, COD2, PAT1, PAT2, f, s):···

    def graficar_piso(self, x, y):···
```

Uso de Tkinder():

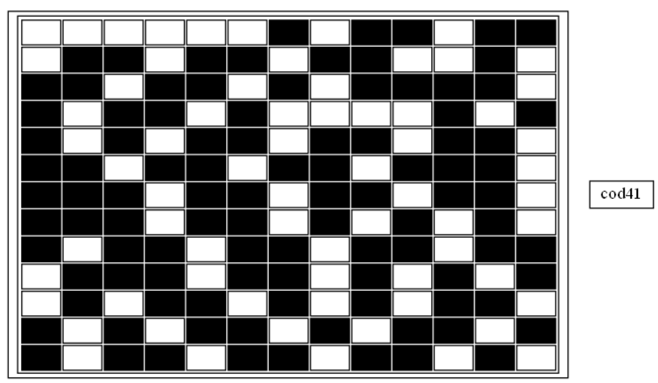
```
def menu(self):
    print("****Seleccione una opcion****\n   1.Cargar Archivo\n   2.Salir")
    op = input()
    if op == '1':
        raiz = Tk()
        myArchivo = filedialog.askopenfilename(title="Abrir Archivo")
        Button(raiz, text= "Abrir Archivo" ).pack
        if myArchivo != None:
            self.archivo = minidom.parse(myArchivo)
            M.leerArchivo()
            print('Archivo Cargado')
            M.menu2()
            raiz.mainloop()
        else:
            print('Archivo no encontrado')
    if op == '2':
        exit()
```

Clase Graficar():

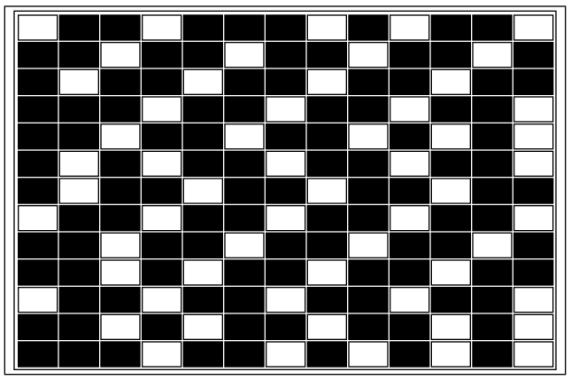
```
import graphviz
class graficar:
  def grafica(patron, name):
    name2 = str(name)+'.gv'
    h = graphviz.Digraph('g', filename=name2, format='png',
                         node_attr={'shape': 'record', 'height': '.1'})
    h.node('tab', label='''<<TABLE>'''+patron+'''</TABLE>>''')
    h.node(name)
    h.view()
```

Patrón de muestra 13x13:

```
 <piso nombre="ejemplo04">
<R> 13 </R>
<C> 13 </C>
<F> 1000 </F>
<S> 1 </S>
<patrones>
<patron codigo="cod41">
WWWWWWBWBBWBBWBBWBBWBBWWBWBBWBBWBWBBBBWBWBBWBWWWWBWBBWBWBBWBBWBBWBBWBBWBBWBBBWBBBWBBWBBWBBWBBBWBBWBWBWBWBW
</patron>
<patron codigo="cod42">
WBBWBBBWBWBBWBBWBBWBBWBBWBBWBBWBBWBBWBBWBBBBBWBBWBBWBBWBBWBBWBBWBWBWBWBWBBWBBWBBWBWBBWBBWBBWBBWBBWBBWBBWBBWBE
</patron>
```



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.



cod42

Diagrama de clases: