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 IPC2 “CERAMICA-BOT”

201900042 – Rodrigo Alejandro Hernández de León

Resumen

El ensayo presenta la explicación de un programa desarrollado en consola que presenta la solución de un problema donde un robot que se especializa en colocar pisos con patrones de tamaño R x C con cualquier patrón, combinando azulejos de color blanco y color negro pueda hacer su trabajo lo menos costeable posible con las operaciones de voltear e intercambiar pisos desde un patrón inicial hacia uno o más patrones destino, donde por medio de un archivo de entrada de tipo XML se obtendrán distintos patrones para un nuevo estilo de piso haciendo las distintas operaciones de tal forma que sea lo más optimo posible y obtener un costo mínimo para que la empresa de Pisos Artesanales lo tome en cuenta y también se le será proporcionado una serie de instrucciones a realizar en distintos formatos como la vista en consola, en un archivo de texto plano y también en forma gráfica en HTML, para poder indicarle al robot lo que tiene que hacer con los patrones del piso y ejecutarlo.

Palabras clave

XML, pisos, patrones, celdas, columnas, filas, intercambiar, voltear, Python, Graphviz.

Abstract

The essay presents the explication of a program developed in console that presents the solution of a problem where a robot that is specialized in installing floors with patterns of size R x C with any pattern, combining white and black tiles can make his work as cheap possible with the operations of flipping and swapping floors from an initial pattern to one or more target patterns, where by means of an XML type input file, different patterns will be obtained for a new style of floor making the different operations in such a way that it is as optimal as possible and to obtain a minimum cost so that the company of Pisos Artesanales takes it into account and also a series of instructions will be provided to him to make in different formats like the view in console, in a plain text file and also in graphical form in HTML, to be able to indicate to the robot what it has to do with the patterns of the floor and to execute it.

Keywords

XML, floors, patterns, cells, columns, rows, swap, flip, Python, Graphviz.

Introducción

Se desarrollo un programa que simula las operaciones realizadas de un robot que intercambia y voltea las celdas de pisos que están en color blanco o negro y al realizar estos procedimientos los pueda realizar de forma optima ya que cada acción que realiza el robot sea de intercambiar las celdas de los pisos o voltear los pisos del color blanco a negro o viceversa representan un costo cada una de estas operaciones y por lo mismo se planteó un algoritmo para que el robot pueda realizar los pasos de forma optima y simulara la forma en que armaría el patrón solicitado desde un patrón inicial que tendría el piso y para poder desarrollar este algoritmo se tuvo la necesidad de crearlo desde el lenguaje de programación Python, utilizando los conceptos de Tipos de Datos Abstractos y la programación orientada a objetos para poder ejecutar la solución al problema planteado.

Desarrollo del tema

La aplicación consiste en simular el proceso en que un robot que ordena pisos para una empresa de Pisos Artesanales pueda ordenar patrones de tamaño R x C y su proceso sea lo más optimo posible para obtener un costo mínimo de su proceso.

Se ha optado el uso del lenguaje de programación Python debido a que es multiparadigma y multiplataforma y el uso del mismo es una herramienta de uso sencillo para poder implementar la solución del sistema.

Para la elaboración de la solución se provee un archivo con extensión y estructura XML, el cual provee la información y contenido de los pisos y sus respectivos patrones, donde luego se procesan en el programa para luego poder realizar la abstracción sus datos y poder generar menús interactivos y fáciles de visualizar para el usuario, para poder cumplir con las necesidades del programa es necesario que la estructura del archivo sea de la siguiente forma:

```
<?xml version="1.0"?>
    <pisosArtesanales>
        <piso nombre="nombrePiso">
        <R> valorEnteroPositivoMayorOIgualQueUno </R>
        <C> valorEnteroPositivoMayorOIgualQueUno </C>
        <F> valorEnteroPositivoMayorOIgualQueUno </F>
        <S> valorEnteroPositivoMayorOIgualQueUno </S>
        <patrones>
            <patron codigo="codigoPatron">
                RxC_Letras_WparaBlanco_BparaNegro
            </patron>
            …
        </patrones>
    </piso>
    …
</pisosArtesanales>
```

Figura 1. Estructura del archivo XML de entrada.

Fuente: Elaboración propia, 2022.

donde:

pisosArtesanales = es la etiqueta padre.

piso = es la etiqueta que indica el piso con que cuenta pisos Artesanales.

nombre = es el identificador del piso.

R = es la etiqueta que indica la cantidad de filas que conforma el piso.

C = es la etiqueta que indica la cantidad de columnas que conforma el piso.

F = es la etiqueta que indica el precio en que le cuesta a la empresa de Pisos Artesanales la operación de voltear un piso.

S = es la etiqueta que indica el precio en que le cuesta a la empresa de Pisos Artesanales la operación de intercambiar pisos.

patrones = es la etiqueta que indica el conjunto patrones que puede contener el piso.

patron = es la etiqueta que indica un patrón del piso.

codigo = es el identificador del patron.

RxC_Letras_WparaBlanco_BparaNegro = indica un conjunto de caracteres W y B con un tamaño de

longitud de R x C, donde W representa el color Blanco y B representa el color Negro en el patron.

Obtenido este formato se necesita leer el archivo y posteriormente procesarlo y para el mismo se necesito la ayuda desde un inicio de la abstracción de datos por medio de una librería importada en Python llamada xml.etree.ElementTree la cual fue utilizada para obtener los datos de los pisos y patrones que contiene el archivo de entrada.

Luego se procedió a guardar los datos por medio del uso de la memoria dinámica y el paradigma de la programación orientada a objetos para que fuera una ayuda a encontrar la solución al problema.

Dentro del almacenamiento de los datos se tuvo la necesidad de usar los tipos de datos abstractos donde con ellos se utilizó las creaciones de los Nodos de pisos, nodos de patrones y nodos de celdas y del mismo poder crear una lista enlazada simple para almacenar los pisos, otra lista enlazada simple para almacenar los patrones y por último una lista doblemente enlazada para almacenar las celdas.

En la lista simple enlazada de pisos se guardan todas las propiedades de los pisos donde se incluía el nombre del piso, la cantidad de filas del piso, la cantidad de columnas del piso, el costo de voltear las celdas, el costo de intercambiar las celdas y una lista enlazada simple que es la lista de patrones.

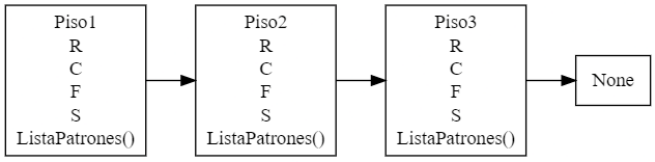

Figura 2. Representación gráfica de la Lista Simple Enlazada de Pisos.

Fuente: Elaboración Propia, 2022.

En la lista simple enlazada de patrones se guardan todos los patrones obtenidos de un piso en específico donde por medio del nodo Patron se guardan las siguientes propiedades que son el código del patron y una lista doblemente enlazada de celdas.

Figura 3. Representación gráfica de la Lista Simple Enlazada de Patrones.

Fuente: Elaboración Propia, 2022.

En la lista doblemente enlazada de celdas se guarda en cada nodo cada carácter del patron de W y B en el archivo teniendo en sus atributos el identificador de la celda, número de fila, número de columna y el color.

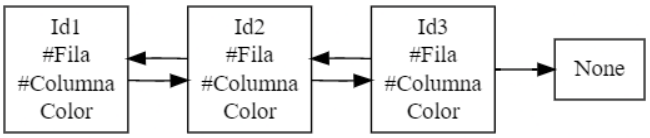

Figura 4. Representación gráfica de la Lista Doblemente Enlazada de Celdas.

Fuente: Elaboración Propia, 2022.

Con el paradigma de programación orientada a objetos se plateó los diferentes tipos de datos abstractos para almacenarlos en la estructura de datos correspondiente, esto abrió la posibilidad de poder trabajar los algoritmos de manera más sencilla y eficiente con el uso de la memoria dinámica.

Un ejemplo de lo mencionado anteriormente fue la creación de los Nodos y Listas donde en el nodo piso tiene relación con la lista simple enlazada de pisos, al mismo tiempo también se enlaza con la lista de patrones que contiene los nodos Patron y que este mismo contiene a la lista de Celdas y dentro sus nodos Celda aplicando la conexión de objetos para iniciar con la solución al problema dado. En las

siguientes imágenes se mostrará la conexión del nodo con la lista, poniendo como ejemplo la lista de Pisos:

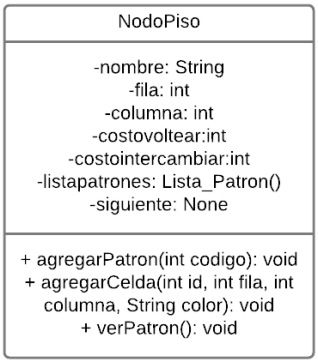

Figura 5. Modelo del tipo de dato NodoPiso.

Fuente: Elaboración Propia, 2022.

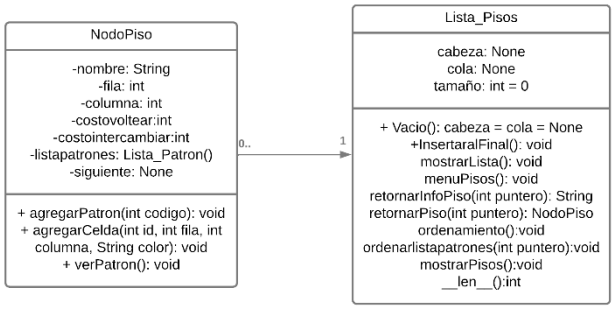

Figura 6. Modelo del tipo de dato Lista_Pisos.

Fuente: Elaboración Propia, 2022.

Con las estructuras ya planteadas, se realizaron los siguientes algoritmos:

a. Procesar Archivo: Este algoritmo se encarga de realizar dos tareas donde primero abstrae los datos del archivo XML y los guarda en sus respectivas estructuras, luego le pide al usuario si desea ver los datos obtenidos del archivo de entrada en orden alfabético utilizando el algoritmo de ordenamiento Burbuja donde se tuvo que utilizar una lista temporal para hacer el intercambio de nodos en la lista. Y también muestra la otra opción para que el usuario pueda realizar las distintas operaciones que puede hacer con los pisos cargados en el programa.

b. Menú de operaciones: Este algoritmo se encarga de cuando el usuario desea iniciar con el menú de pisos pueda mostrar los pisos cargados en el sistema, luego obteniendo el acceso de las celdas de cada patron muestre el patron inicial del piso y luego se crea un menú para que seleccione un patron destino al cual quiere convertir ese patron inicial.

c. Graficar Patron inicial: Este algoritmo tuvo la necesidad de la herramienta Graphviz instalada en el dispositivo. Para poder cumplir con mostrar la grafica del patron inicial se tuvo que llamar al primer elemento de la lista de patrones y obtener su lista de celdas donde se obtuvo la cantidad de filas y columnas y poder recorrer la lista de celdas y junto con un poco de lenguaje dot se implementó la gráfica, creando el archivo con extensión dot y poder abrirlo como tipo PNG.

d. Operar Patron: Este algoritmo se encarga de que tomaremos una lista doblemente enlazada de celdas la cual será la lista de celdas del patron inicial y otra lista doblemente enlazada que será nuestra lista auxiliar obtenida desde el patron destino elegido por el usuario y se hará las respectivas comparaciones de cada nodo y verifica si sus colores son similares de lo contrario necesitara de operar haciendo el intercambio de nodos hacia la derecha o hacia

abajo y si no cumple para poder realizar el procedimiento de intercambio de celdas entonces se procede a voltear la celda y viéndolo en el punto de vista de la programación se realiza el procedimiento de cambiar el valor del color del nodo al cual esta valuando y en ambos casos se aumenta un contador de costo total y al mismo tiempo se llama a un método para que grafique cada procedimiento hecho con las celdas y luego al cumplir con terminar de recorrer el patron y convertir a que se convierta en el patron destino, se muestra una imagen del resultado de la operación hecha con las listas y mostrando como quedo el orden de la lista en consola, junto con mostrar el costo total de las operaciones.

e. Reportes: En este algoritmo se encarga de hacer tres tareas, donde una es para obtener el texto de los procedimientos hechos con la operación de los patrones y mostrarlos en consola, otra es obteniendo el texto de los procedimientos, almacenarlos y creando el archivo en formato TXT y como parte de otra tarea es también obteniendo el contenido que se obtuvo desde el método de operar patron y poder redactar y crear un archivo en formato HTML con imágenes para que tenga mejor visibilidad de las operaciones hechas por el programa para llegar al patron destino y en los tres casos se muestra el costo total de las operaciones realizadas por el sistema.

Conclusiones

Se cumplió el propósito de conocer, aprender e implementar las estructuras de datos para dar una solución eficiente y sencilla al problema utilizando los datos leídos en un archivo de entrada y almacenados en los mismos.

La solución del problema da un ejemplo de como un pequeño programa puede realizar una simulación de lo que un robot especializado en ordenar patrones de pisos puede cambiar patrones de forma eficiente.

El uso de estructuras de datos para manejar memoria dinámica llega a ser útil para la solución de problemas no únicamente de memoria sino de simulación de optimización de procesos que llegan a ser muy importantes en el ámbito laboral.

Es importante conocer desde el lenguaje de programación al cual se va a utilizar, hasta el uso del paradigma para poder realizar los procesos necesarios al construir los algoritmos que determinan la solución del problema.

Referencias bibliográficas

USAC, T. d. (7 de Febrero de 2022). *Proyecto 1, UEDi*. Obtenido de https://uedi.ingenieria.usac.edu.gt/campus/pluginfile.php/272548/mod_resource/content/0/%5BIPC2%5DProyecto%20_1.pdf

Apéndice

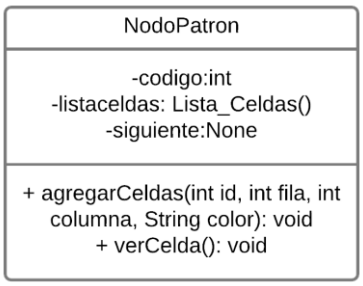

Figura 7. Modelo del tipo de dato NodoPatron.

Fuente: Elaboración Propia, 2022.

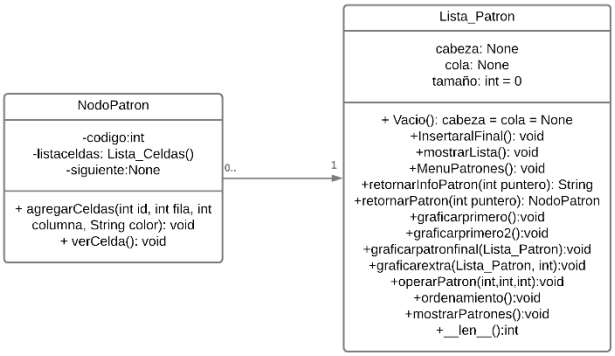

Figura 8. Modelo del tipo de dato Lista_Patron.

Fuente: Elaboración Propia, 2022.

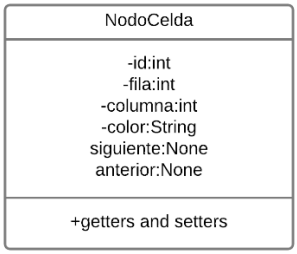

Figura 9. Modelo del tipo de dato NodoCelda.

Fuente: Elaboración Propia, 2022.

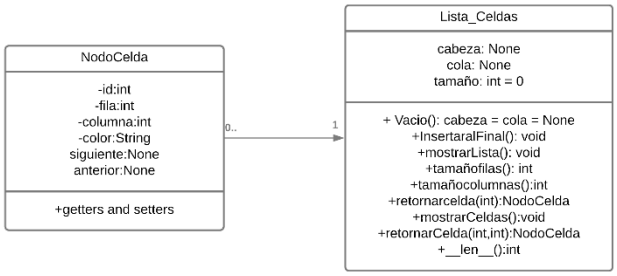

Figura 10. Modelo del tipo de dato Lista_Celdas.

Fuente: Elaboración Propia, 2022.

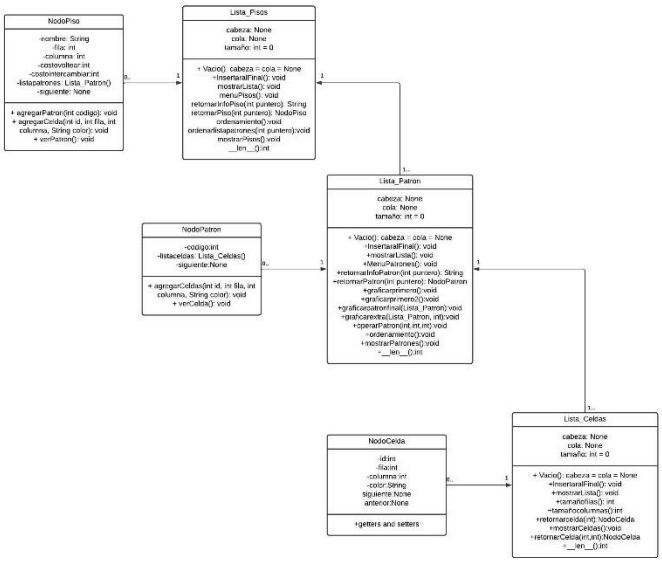

Figura 11. Diagrama de clases de las estructuras utilizadas para la solución del problema.

Fuente: Elaboración Propia, 2022.

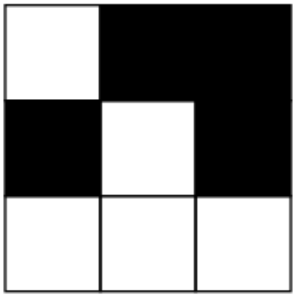

Figura 11. Ejemplo del uso de la herramienta Graphviz para graficar patrones.

Fuente: Elaboración Propia, 2022.