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 2021.

TABLERO GRAFICO IMPLEMENTADO POR UNA MATRIZ DISPERSA

201800500 – Anderson Gerardo Zuleta Galdamez

Resumen

Este proyecto tiene como objetivo realizar interfaces gráficas, programación orientada a objetos, estructuras secuenciales, cíclicas y condicionales, y uno lo mas importante almacenar datos en una matriz dispersa. Una matriz dispersa es una matriz en que la mayoría de los elementos son cero.

El proyecto consiste en un juego el cual contara con un tablero mxn representado a través de una matriz dispersa, dicho juego contara son seis diferentes piezas las cuales pueden ser colocadas a lo largo del tablero de juego. El almacenamiento de los datos de los jugadores se almacenará en listas creadas propiamente por el estudiante. Así mismo los jugadores deberán de escoger uno de los cuatro diferentes colores que se les presentará en pantalla, no se permitirá el mismo color entre jugadores.
Las piezas que el juego contiene saldrán aleatoriamente, mostrando un tamaño para que el jugador pueda ingresarlo al tablero. El juego permitirá con diferentes reportes, uno un tablero grafico mediante Graphviz y una página hecha en HTML.

Palabras clave
Matriz, Dispersión, Modelación, Grafica.

Abstract

the objective of this project performs graphical interfaces, object-oriented programming, sequential, cyclic and conditional structures, and one most importantly store data in a sparse matrix. A sparse matrix is a matrix in which most of the elements are zero.
The project consists of a game which will have a mxn board represented through a sparce matrix, game will have six different pieces which can be placed along the game board. Player data storage will be stored in lists created by the student itself. Likewise, players must choose one of the four different colors that will be presented on the screen, the same color will not be allowed between players. The pieces that the game contains will come out randomly, showing a size so that the player can enter it on the board. The game will allow with different reports, one a graphic board through Graphviz and a page made in HTML.

Keywords
Matrix, Sparce, Modeling, Graphic.

Introducción

La manipulación de estructuras de datos. Nos permite la organización de los datos en una computadora para que puedan ser utilizados de una manera muy eficaz. Existen diferentes tipos de estructuras de datos que son adecuados para la utilización de diferentes tipos de aplicaciones.

En esta aplicación se busca una estructura cuya forma de almacenacion en la memoria sea la más optima. Por eso mismo la creación de una matriz dispersa es una opción muy eficaz, dado que la matriz dispersa es ampliamente usada en la computación científica, especialmente en la optimización a gran escala. Otra área de interés en donde se pueda aplicar de manera exitosa es en la teoría de grafos, teoría de redes y los métodos numéricos.

La aplicación intenta dejar explicado de una forma muy sencilla el manejo óptimo de las posiciones del tablero implementado gráficamente en Python. Para que el usuario vea gráficamente de una manera concreta los movimientos realizados en la aplicación se implementaron reportes gráficos, uno de ellos una página web creada por HTML y un grafo mostrando la colocación de las piezas que el jugador colocaba.

Desarrollo del tema

Una matriz o arreglo bidimensional, es una zona de almacenamiento continuo, que contiene una serie de elementos del mismo tipo, desde el punto de vista lógico una matriz puede verse como un conjunto de elementos en fila, o filas y columnas si tuviesen dos dimensiones.

Cada elemento de la estructura puede ser accedida por medio del índice o posición del elemento en la matriz.

Conceptos:

a. Es un arreglo con dos dimensiones de datos.
b. La cantidad de columnas y filas puede o no ser la misma.
c. En computación, es generalmente representada por arreglos.

La creación de una matriz dispersa (sparce matrix) es relativamente sencilla. Lo importante es conocer los apuntadores de memoria que creamos en sus métodos.

Formas de implementar:

1. Arreglos.
2. Lista enladaza.
3. Enlaces de datos
4. Listas con cabecera.

A continuación, veremos cuales son las formas que se implementa con las diferentes opciones al crear una matriz dispersa.

1. Arreglos: En los arreglos existe desperdicio de memoria. Las posiciones 0, existen. Generalmente es implementado como un arreglo de dos posiciones, también es posible la utilización de un arreglo de una sola posición, pero es necesario más de algún método de indexación. La utilización de arreglos es muy eficiente para acceder a los datos.

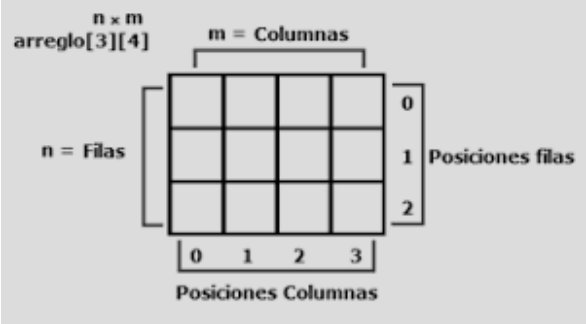

Figura 1. Implementación mediante arreglos.

2. Lista enlazada: Problemas de rendimiento, en los métodos de inserción(), eliminación() y búsqueda(). Se puede implementar con una lista simple o doble, en el cual los elementos se insertan ordenados. Recalcar que es necesario la utilización de algún método de alineación. Cada nodo de la estructura guarda su posición (x,y), su valor y apuntador a siguiente. Viendo el lado de la memoria es un leve desperdicio por lo que podría ser una opción muy satisfactoria, sin embargo, existen problemas de rendimiento para acceder a los datos.

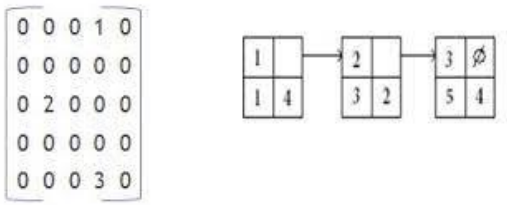

Figura 2. Implementación lista enlazada.

3. Enlaces de datos: Mejora el rendimiento, pero consume memoria adicional almacenando los valores de (x,y) en los nodos. La implementación puede ser una lista simple o doble. En la cual los elementos se insertan ordenados, cada nodo de la estructura guarda su posición (x,y). Se implementa para cada nodo, apuntadores hacia arriba, abajo, siguiente y anterior. Existe un leve desperdicio de memora, el rendimiento de las operaciones son óptimos, pero existe una complejidad adicional al enlazar los datos.

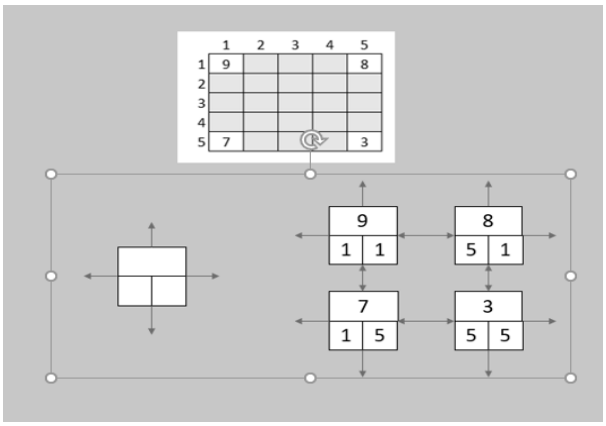

Figura 3. Implementación mediante enlaces.

5. Lista con cabeceras: Mejora el rendimiento y reduce el consumo de memoria solo almacenando solo una coordenada en los nodos. La implementación puede ser mediante una lista simple o doble, en la cual lo elementos son insertado ordenadamente. Cada nodo de la estructura guarda únicamente su valor y los apuntadores son necesarios, no existe ningún desperdicio de memoria y el rendimiento de las operaciones es optimo.

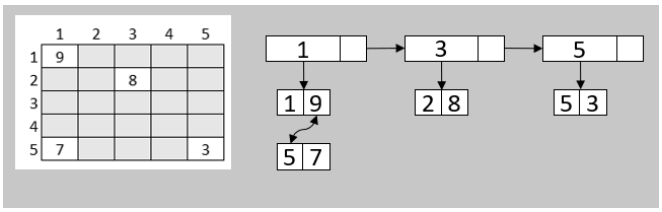

Figura 3. Implementación mediante cabeceras.

Conclusiones

Conociendo las diferentes formas de implementar una matriz dispersa y que el trabajo fue elaborado con cabeceras por el motivo que el desperdicio de la memoria no existía y que las búsquedas de datos eran muy fáciles de encontrar, se procedió a la creación de cabeceras con sus nodos.

Durante la creación de as cabeceras con sus respectivos apuntadores, se percató que la creación de clases para poder interpretar mejor el manejo de la misma. Es decir, una clase para nuestra cabecera y una lista para nuestra matriz, sin perder los apuntadores de las cabeceras con nuestros nodos matriz.

La eficacia del uso de dicho método fue exitosa, sin embargo fue algo complicado en la apuntación de nuestro apuntadores con los nodos siguientes en la inserción de valores para luego poder obtener una grafica por medio de Graphviz.

Se recomienda realizar pruebas constantes para tener un punto de referencia para posibles fallas durante la construcción de la solución. Para poder evitar la complejidad de búsqueda de errores.

Referencias bibliográficas

Graphviz. (s. f.). The DOT Language. graphviz.org. Recuperado 5 de abril de 2021, de https://www.graphviz.org/doc/info/lang.html

pypi. (2020, 24 diciembre). Graphviz. Recuperado 5 de abril de 2021, de, https://pypi.org/project/graphviz/#description

python.org. (s. f.). xml.etree.ElementTree — The ElementTree XML API — Python 3.9.2 documentation. python. Recuperado 5 de abril de 2021, de https://docs.python.org/3/library/xml.etree.elementtree.ht
ml

Python. (s. f.). tkinter — Python interface to Tcl/Tk. Python.org. Recuperado 5 de abril de 2021, de https://docs.python.org/3/library/tkinter.html