|  |
| --- |
| **Proyecto 2 “CHAMPIN WARRIORS S.A”** |
| **Carnet - 201513677** |

**Resumen**

Las estructuras de datos son parte fundamental para el desarrollo de programas complejos que se diseñan cuidadosamente, este es el caso a el proyecto de “Chapin Warriors S.A”, ya que se tenía que tener una forma determinada de organizar los datos de un grupo de información para poder utilizarlos de la manera más efectiva posible. El manejo de una buena lógica de programación, son un beneficio para hacer de estas estructuras de más beneficio en el proceso.

La estructura antes menciona se aplica en el funcionamiento del programa para la forma más optima de poder generar el menor monto de “Chapín Warriors S.A”, ", la cual consiste en encontrar la seria de pasos más óptimo haciendo referencia a una lista enlazada donde están almacenadas las posiciones como un color en un espacio asignado para cada ciudad en una estructura de una matriz dispersa. El usuario que coloca el robot a utilizar por el cual quiere cambiar a uno nuevo y obtener una misión satisfactoria o una respuesta si la puede realizar o no.

La forma de aplicar las estructuras a la resolución es al almacenar los datos de cada ciudad y robot en una lista simple y obtener un a lista de ciudades y por cada lista de pisos otra lista simple que contenga los patrones para cada ciudad incrustado en la segunda estructura que será una matriz dispersa como uno de sus parámetros para cada uno de ellos, para poder acceder de una forma más ordenada y sencilla para la manipulación del algoritmo del camino más óptimo.

**Palabras clave**

Tipo de datos abstractos, matriz dispersa, memoria dinámica, lista doblemente enlazada, estructura.

***Abstract***

Data structures are a fundamental part for the development of complex programs that are carefully designed, this is the case with the "Chapin Warriors SA" project, since they had to have a certain way of organizing the data of a group of information in order to use them in the most effective way possible. The handling of a good programming logic, are a benefit to make these structures of more benefit in the process.

The aforementioned structure is applied in the operation of the program for the most optimal way to be able to generate the least amount of “Chapín Warriors SA”, ", which consists of finding the most optimal series of steps by referring to a linked list where they are stored the positions as a color in an allocated space for each city in a sparse matrix structure The user who places the robot to use for which he wants to change to a new one and get a successful mission or an answer if he can perform it or no.

The way to apply the structures to the resolution is to store the data of each city and robot in a simple list and obtain a list of cities and for each list of floors another simple list that contains the patterns for each city embedded in the second structure that will be a sparse matrix as one of its parameters for each of them, to be able to access it in a more orderly and simple way for the manipulation of the algorithm of the most optimal path.

***Keywords***

*Abstract data type, sparse matrix, dynamic memory, doubly linked list, structure.*

**Introducción**

Las estructuras de datos en la computación son utilizadas en la optimización a gran escala, otras de las áreas de interés en donde se pueden aplicar la representación de contenido de una manera ordenada y dinámica ya que solo se ocupa el espacio utilizado necesario, y no espacio vacío que no nos trae ninguna función.

En las operaciones que se realiza en el "Robot " se utiliza la estructura de la matriz dispersa para implementar la simulación de una ciudad con caminos . En donde se irán almacenando en cada uno de los nodos la información de sus posiciones y el contenido.

La lista enlazada contiene como atributo dos punteros de forma que uno apunta al siguiente y el otro al predecesor, lo cual lo hace un TDA dinámico lineal y se aplica para todos los datos de nuestros robots y ciudades.

Dado que este tipo de estructuras ocurren se desarrolla distintos métodos que permiten la inserción de datos, intentando que sea lo más eficiente en la computadora y su almacenamiento en memoria.

**Desarrollo del tema**

1. **Abstracción**

La abstracción es un proceso mental mediante el cual el ser humano tiene la capacidad de extraer los rasgos esenciales de “algo” para representarlos por medio de un lenguaje gráfico o escrito.

1. **Estructura de datos**

Una estructura de datos, en general se puede definir como cualquier colección o grupo de datos

organizados de tal forma que tengan asociados un conjunto de operaciones para poder manipularlos.

Las estructuras de datos se implementan a través de los lenguajes de programación y son un modelo que se caracteriza por permitir el almacenamiento y utilizar una determinada organización de datos.

1. **Estructura de datos dinámicos**

Son aquellas en las que su ocupación en memoria puede aumentar o disminuir durante el tiempo de ejecución de un programa. A su ves las estructuras de datos dinámicos se pueden clasificar en lineales y no lineales:

Estructuras lineales. Son aquellas en las que se definen secuencias como conjuntos de elementos entro los que se establece una relación de predecesor y sucesor. Las estructuras de datos basadas en este concepto se diferencian por las operaciones de acceso a los elementos y manipulación de las estructuras. Existen tres estructuras lineales especialmente importantes en las que se encuentran las listas

Estructuras no lineales. Son aquellas en las que no existe una relación de adyacencia entre sus elementos, es decir, un elemento puede estar relacionado con cero, uno o mas elementos. Existen dos estructuras no lineales especialmente importantes: los árboles y los grafos

1. **Listas Dobles**

Una lista doble es aquella en que sus nodos se encuentran encadenados por dos apuntadores, es decir, cada nodo apunta al siguiente nodo de la lista, así como al nodo que le antecede en la lista. Una lista doble puede ser implementada como lineal o circular. En una lista lineal, el apuntador siguiente del ultimo nodo y el apuntador anterior del primer nodo apunta hacia un valor nulo, mientras que en una lista circular el apuntador siguiente del ultimo nodo apunta hacia el primer nodo de la lista y el apuntador anterior del primer nodo apunta hacia el ultimo nodo de la lista



Lista doble lineal debe contener como mínimo tres campos: uno para almacenar la información y los otros dos para guardar la dirección de memoria del nodo antecesor y sucesor.



1. **Listas Simples**

Es una estructura de datos lineal y dinámica que se compone de un conjunto de nodos en secuencia enlazados mediante un apuntador o referencia

* La lista simple tiene un apuntador inicial
* El último nodo de la lista apunta a nulo



La lista enlazada se compone de nodos (objetos instanciados pertenecientes a la clase Node), cada uno de los cuales tiene dos únicas tareas: guardar la información de la posición i y ofrecer una referencia a la posición i+1.

1. **Operaciones Básicas**

En una lista se pueden efectuar operaciones por medio de algoritmos que se deben desarrollar de acuerdo con el tipo de lista. Algunas de las operaciones básicas son

* **Recorrido.** Esta operación consiste en visitar todos los nodos que forman parte de una lista. Para recorrer todos los nodos de la lista es necesario posicionarse en el primer nodo de la lista y después avanzar hacia el apuntador siguiente hasta encontrar el final.
* **Inserción.** Esta operación consiste en agregar un nuevo nodo a una lista. La ubicación del nuevo nodo puede ser al inicio, al final o en cualquier posición dentro de la lista
* **Borrado.** Esta operación consiste en eliminar un nodo de la lista y redireccionar los apuntadores al nodo antecesor y al sucesor. El borrado también se aplica al primer nodo de la lista, como al último y una posición intermedia
* **Búsqueda.** Esta operación consiste en recorrer todos los nodos de la lista desde el primer nodo para ir comparando el valor de cada nodo con el valor que se está buscando hasta encontrar el nodo con el valor indicado o encontrar el fin de la lista.

1. **Programación Orientada a Objetos**

Este enfoque de programación es adecuado para programas que son grandes, complejos y se actualizan o mantienen activamente.

La organización de un programa orientado a objetos también hace que el método sea beneficioso para el desarrollo

1. **Listas ortogonales**

Es aquella en la que sus nodos se encuentran encadenados por cuatro apuntadores, es decir, cada nodo se encuentra doblemente enlazado en forma horizontal, y cada nodo se encuentra doblemente enlazado en forma vertical. Esta lista puede ser implementada como lineal o circular. Este tipo de listas se puede utilizar para representar matrices.

Diagrama

Descripción generada automáticamente

**Ventajas y Desventajas**

Las listas son estructuras de datos que son dinámicas, esto significa que adquieren espacio y liberan espacio a medida que se necesita. Pueden definirse estructuras más complejas a partir de las listas.

Sin embargo, hay una advertencia. Como regla general siempre hay que tener cuidado al manejar direcciones de espacios de memoria, porque es posible que se acceda a una localidad de memoria de la cual no se desea cambiar su contenido.

**Conclusiones**

El uso de la programación orientada a objetos, nos facilita y beneficia en el uso y optimización de memoria en las aplicaciones.

Al implementar tipos de datos abstractos bajo el concepto de programación los nodos son clases y dependiendo del tipo de lista se aplica una o varias clases de sus operaciones.

Al utilizar estructuras secuenciales en Python se observa como funcionan los apuntadores, ya que son secuenciales dependemos de estos.

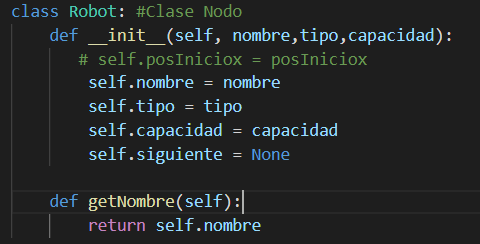
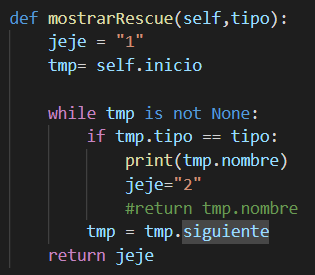
**Bibliografías**

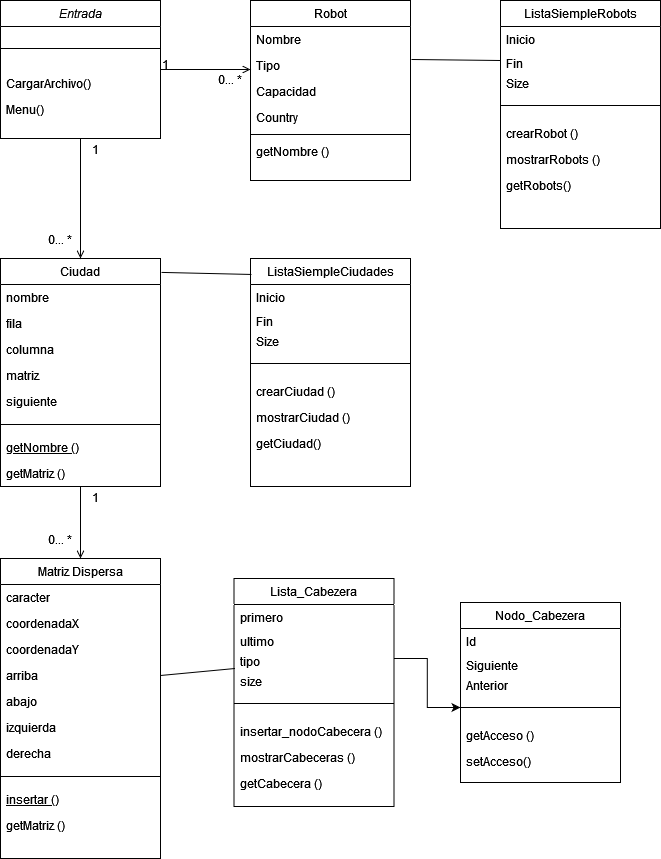
Monica Carreño, Andres Sandoval, (2012). *Listas.* VABCS – Mexico.

Joyanes Aguilar, Luis (2007). Estructuras de datos, McGraw-Hill, España.

**Apéndices**

**def mostrarRescue():** Función que utilice para poder encontrar la coincidencia de todos los robots, en donde revisaba si eran del mismo tipo para poder mostrarlos y asi poder realizar o no las misiones solicitadas, haciendo referencia a un atributo de tipo Robot en una lista simple de robots.



**Archivos de salida**

