INFORME DEL PROYECTO 2
INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 2

201906572 – Allan Ricardo Barillas Sosa

Resumen

Se le llama estructura de datos en la programación a un conjunto de espacios en la memoria estática ordenados de una manera especificada donde cada uno de los espacios en la memoria llamados nodos están contiguos unos a otros.

Un nodo en la programación orientada a objetos es un objeto, el cual puede contener todas las características de un objeto añadiendo el nombre de la ubicación del espacio en la memoria asignado.

Abstract

A data structure in programming is a set of spaces in static memory arranged in a specified manner where each of the spaces in memory called nodes are contiguous to each other.

A node in object-oriented programming is an object, which can contain all the characteristics of an object by adding the name of the space location in the allocated memory.

+++

Palabras clave

Cabecera: Listado de nodos utilizados para guiar el orden de los nodos añadidos a una matriz ortogonal.

Nodo(siguiente) ← NULL: El campo almacena un valor nulo para indicar que no existe sucesor del nodo.

Primero: dirección en donde se encuentra el primer nodo de la lista

Buscar Nodo: Devuelve un nodo buscado por su id.

Robot : Objeto capaz de realizar misiones.

Ciudad: Objeto/Nodo que cuenta con un área donde se encuentran, conjunto de militares, civiles y recursos.

ChapinFighter: Robot con objetivo de recolectar los recursos de la ciudad.

ChapinRescue: Robot con objetivo de recuperar los civiles distribuidos en la ciudad.

Keywords

Header: List of nodes used to guide the order of nodes added to an orthogonal array.

Node(next): The field stores the memory address of the successor node of the respective node. Depending on the type of list, a node can have more than a single link.

Head: An address where the first node of the list is located.

Find Node: Returns a node buscado by id.

Robot: Object capable of performing missions.

City: Object/Node that has an area where they are located, set of military, civilians, and resources.

ChapinFighter: Robot with the aim of collecting the city's resources.

ChapinRescue: Robot with the aim of recovering the civilians distributed in the city.

Introducción

El tipo de datos abstractos (TDA) es un modelo que describe datos u objetos y actividades que se puedenrealizar en él. Y se le llama abstracto porque laintención es que quien lo use no necesite conocer los detalles de la representación interna o la ejecución de las actividades, lo que permite la colocación de datosy actividades y gracias a esto se puede ocultar, cómoTDA es. implementado.

La complejidad primordial del proyecto fue realizar las misiones y encontrar un camino por el cual el robot sea capaz de realizar la misión deseada, para la creación de este algoritmo se busco poder encontrar la primera de muchas opciones de camino, con el cual se intento hacer un movimiento aleatorio para poder probar todas las posibilidades de movimientos disponibles para el robot y que este pudiera cumplir con las restricciones establecidas en el enunciado.

Desarrollo del tema

Este programa fue desarrollado a través de una solución integral que implementó tipos de datos hipotéticos (TDA) e imágenes (Graphviz) bajo el concepto de Planificación Orientada a Objetos (POO).

El tipo de datos de resumen es una estructura que describe datos u objetos y actividades que se pueden realizar en él. Y se le llama abstracto porque la intención es que quien lo use no necesite conocer los detalles de la representación interna o la ejecución de las actividades, lo que permite la colocación de datos y actividades y gracias a esto se puede ocultar, cómo TDA es. implementado.

Debido a esto, podemos usar TDA para separar el código del código que está implementando.

Al usar TDA, puede crear listas que son estructuras de datos flexibles, lo que significa que se les da espacio y mantienen el espacio según sea necesario. Las estructuras más complejas y ampliamente utilizadas, como la configuración de lista, se pueden definir a partir de la lista. A veces, los gráficos se definen como listas cerradas. También se utiliza como columnas de lista para tablas.

a. Estructuras lineales: Son aquellas en las que el orden se define como los conjuntos de elementos sobre los que se construye la relación predecesora y sucesora.
b. Las diferentesestructuras de datos de acuerdo con este conceptodifieren en el funcionamiento de los componentes de adquisición y el funcionamiento de la estructura. Hay tres estructuras de filas principales: columnas, colas y listas.
c. Estructuras no lineales. Son aquellas en las que no existe una relación estrecha entre sus componentes, es decir. Un elemento puede estar vinculado a cero, uno o más elementos. Hay dos diseños principales que no son de línea: árboles y gráficos.

Listas ortogonales:
Una lista ortogonal es aquella en el que los nodos están conectados por cuatro enlaces, es decir, cada nodo está conectado horizontalmente dos veces y cada nodo está conectado verticalmente dos veces. La lista ortogonal se puede diseñar como una línea o un círculo. Estos tipos de listas se pueden utilizar para representar matrices.

Listas sencillas circular:
Una lista sencilla es aquella cuyos nodos solo están conectados al nodo, es decir, cada nodo apunta al siguiente nodo de la lista y cada nodo se redirige al nodo anterior de la lista, con la excepción del primer nodo y el último nodo de la lista. En un círculo, el último nodo de la lista apunta al primer nodo de la lista.

Las listas pueden ser operadas por algoritmos que deben desarrollarse según el tipo de lista. Algunas de las actividades básicas que se pueden realizar en la lista son:

Recorrido. Esta operación implica visitar todos los nodos que forman parte de la lista. Para recorrer todos los nodos de la lista, es necesario ubicarse en el primer nodo de la lista y luego moverse hacia el nodo que apunta al siguiente enlace y así sucesivamente hasta encontrar el final de la lista.

Inserción. Esta operación consiste en agregar un nuevo nodo a la dirección. La ubicación del nuevo nodo puede estar al principio, al final o casi en cualquier lugar.

Borrado. Esta operación consiste en eliminar un nodo del directorio y redireccionar los enlaces del nodo anterior y posterior en el caso de un nodo que se encuentre en la posición intermedia. La eliminación se puede aplicar tanto al primer nodo de la lista como al último nodo de la lista.

Búsqueda. Esta operación va desde el primer nodo a todos los nodos de la lista para comparar el valor de cada nodo.

El desarrollo del programa se costa las siguientes opciones que se muestran en el menú principal, las opciones son: Cargar archivos, procesar archivos, escribir archivo de salida, mostrar datos del estudiante, generar gráfica y salir.

- Cargar archivos: en esta parte se lee unarchivo con extensión y estructura xml en elcual se limitan las siguientes etiquetas: ListaCiudades: este será necesario para la lecturainicial del archivo, ya que será la etiqueta padre de todo.
- Ciudades: esta etiqueta será la que indica que vendrán un conjunto de ciudades, las cuales se añadirán a nuestra lista de ciudades.
- Robot: esta etiqueta indicara los robots con los que contamos para realizar las misiones de rescate o recolección de recursos, esto según el tipo de robot con el que contemos, los tipos de robots son ChapinFighter y ChapinRescue.
- nombre: indicara el nombre de cada ciudad.
- Fila #: según el numero con el que este acompañando, este esta indicando las posiciones de esa fila y con que tipo de ocupación cuenta.

Militares: esta etiqueta encierra a todas las unidades militares que encontraremos dentro de la etiqueta de ciudad actual.

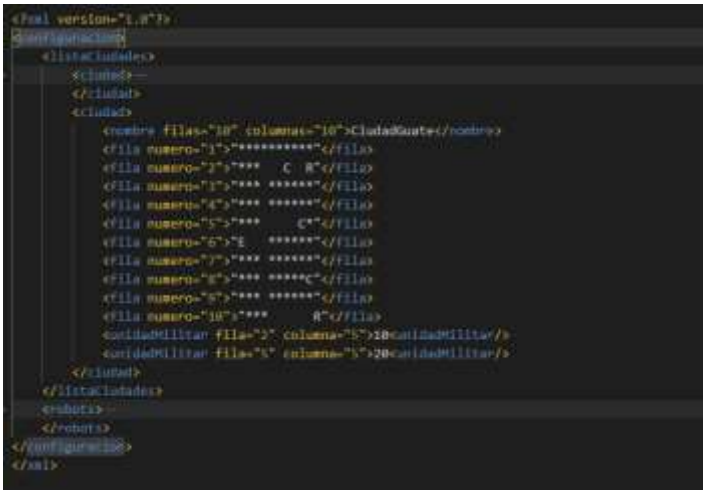


Figura I. XML proporcionado para lectura de datos

Fuente: elaboración propia, 2021

Después de leer el xml con los campos correctos, se procede a la creación de la matriz donde nos muestra las posiciones intransitables, la posición de civiles, recursos, entradas y unidades militares de la figura 2.

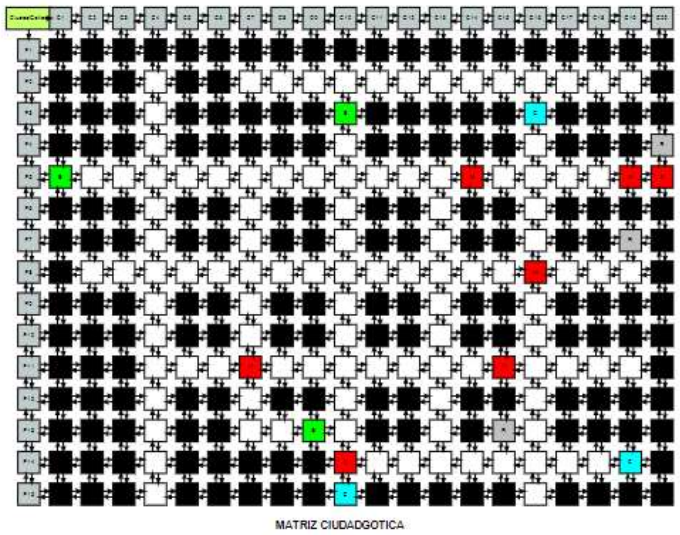


Figura II. Diagrama de una ciudad

Fuente: elaboración propia, 2022

Para procesar la información del xml procedemos a hacer un parse mediante la biblioteca minidom, para obtener todos los campos necesarios para crear un objeto de tipo canción.

a. Generar gráfica: al elegir esta opción se mostrará un menú con los nombres de las matrices que se encuentran guardas en la lista doblemente enlazada de la biblioteca.

   Para generar la gráfica se hizo usa de Graphviz el cual se instaló dentro de la IDE por medio del pip install, la gráfica que se genera es la de la figuraVI.

```
def reporteGraph(self):
    archivo = open("lista.dot", "w")
    archivo.write('digraph grid{')
    archivo.write('layout=dot \n labelloc = "t"')
    archivo.write('edge [weigth=1000 style=dashed color=dimgrey]\n\n
    archivo.write('rankdir="LR"\n')
```

Figura VI. Creación .dot

Fuente: elaboración propia, 2022

b. Realizar misión: busca un camino por el cual el nodo inicial (entrada seleccionada) pueda encontrar un camino por medio de los nodos contiguos hasta llegar al nodo final (recurso o civil deseado de obtener).

Conclusiones

1. Para el desarrollo de este proyecto es necesario tener claro los conceptos de tipos de datos abstractos y visualización a través de graphiz.

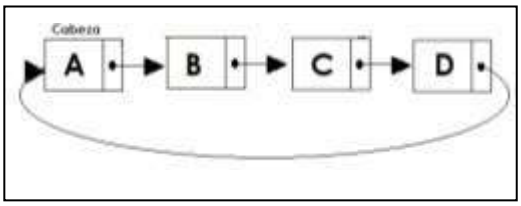
2. El uso de matrices doblemente enlazadas facilita la navegación de una estructura de varias listas al trabajar con tipos de datos abstractos.

3. Es necesaria la implementación orientada a objetos en el desarrollo de este tipo de proyectos en donde se hace uso de TDA ya que hace más fácil la manipulación de estos.

4. Se debe tener en cuenta que el proyecto se desarrollo con los siguientes datos técnicos:
   Lenguaje utilizado: Python
   IDE utilizado: Visual Studio Code
   Sistema operativo: Windows 10(64 bits)

Apéndice

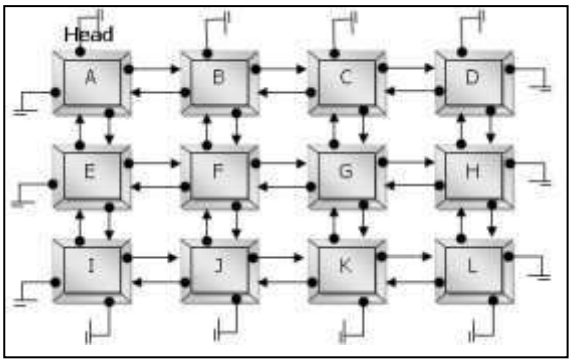
Lista circular sencilla:



Apéndice I. Lista Circular

Fuente: elaboración propia, 2021

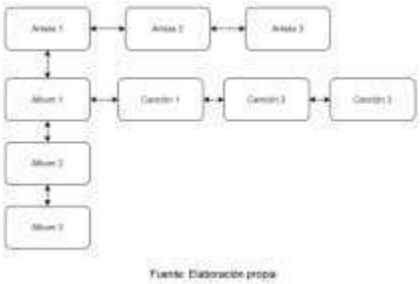
Lista ortogonal:



Apéndice II. Lista Circular

Fuente: problemas resueltos de listas ligadas

Lista doblemente enlazada:



Fuente: problemas resueltos de listas ligadas

Lista doblemente enlazada:

Anexos:

Diagrama de clases

Diagrama de Clases.pdf