INFORME DEL PROYECTO 1 INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 2

**201906572 – Allan Ricardo Barillas Sosa**

**Resumen**

Una estructura de datos en general es una colección de datos, unidos y contiguos por la memoria dinámica del computador, llevando consigo distintos nodos apuntando al consiguiente cada nodo de una lista es un mismo tipo de dato u objeto llamado en la programación.

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

***Abstract***

*A data structure generally can be defined as any collection or group of data organized in such a way that they have associated a set of operations to be able to manipulate them. Data structures are implemented through programming languages and are a model that is characterized by allowing the storage and use of a certain organization of data A data structure can be of two types: Static data structures and Dynamic data structures.*

*Dynamic data structures: They are those in which their memory occupation can increase or decrease during the execution time of a program. In turn, dynamic data structures can be classified as linear and non-linear.*

+++

**Palabras clave**

Nodo(siguiente): El campo almacena la dirección de memoria del nodo sucesor del nodo. Dependiendo del tipo de lista un nodo puede tener más de un enlace.

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

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

Nuevo (): Reserva un espacio en memoria y la asigna a una variable de tipo apuntador.

Leer (): Permite introducir un valor por el usuario a una variable

Mensaje (“ “):Visualiza el mensaje que se encuentra entre comillas

Mostar (): Visualiza el contenido de una variable.

: asignar un valor a una variable

Lista circular sencilla: es aquella en la que sus nodos se encuentran enlazados únicamente por una liga, y el último nodo de la lista apunta hacia el primer nodo de la lista.

Lista Ortogonal: es aquella en la que sus nodos se encuentran encadenados por cuatro ligas.

Nodo: Apuntador al nodo cuya dirección de memoria está en nodo.

Nodo(valor): El campo de información para el nodo apuntado por nodo.

***Keywords***

*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.*

*Node(next)*  *NULL: The chosen field stores a null value to indicate that there is no successor of the node.*

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

*simple circular list: It is the one that has every node bonded by a link, the last node in the list points the direction of the first node in the list.*

*orthogonal list: It is the one in which all nodes are chained by four leagues.*

*New (): Reserves a space in memory and assigns it to a variable of pointer type.*

*Read (): Allows the user to introduce a value to one of the variables*

*Message (“ “): Displays the message enclosed in quotation marks*

*Show (): Displays the content of “x” variable.*

*: assigns a value to a specific chosen variable*

*Node: Pointer to the node whose memory address is at node*

*Node(value): The information field for the node pointed by the node.*

**Introducción**

El tipo de datos abstractos (TDA) es un modelo 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.

**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.

1. 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. Las diferentes estructuras de datos de acuerdo con este concepto difieren en el funcionamiento de los componentes de adquisición y el funcionamiento de la estructura. Hay tres estructuras de filas principales: columnas, colas y listas.

1. 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:

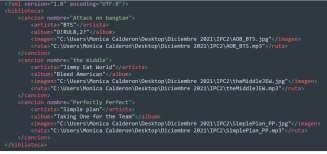
Una lista sencilla es aquella cuyos nodos solo están conectados al nodo, es decir, cada nodo apunta al siguiente nodo de la lista y cada nodo se redirige al nodo anterior de la lista, con la excepción del primer nodo y el último nodo de la lista. En un círculo, el último nodo de la lista apunta al primer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

* 1. 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.
  2. 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.
  3. 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.
  4. Búsqueda. Esta operación va desde el primer nodo a todos los nodos de la lista para comparar el valor de cada nodo.

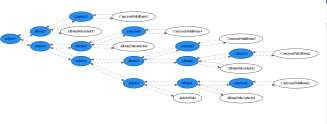
El desarrollo del programa se costa las siguientes opciones que se muestran en el menú principal, las opciones son: Cargar archivos, procesar archivos, escribir archivo de salida, mostrar datos del estudiante, generar gráfica y salir.

1. Cargar archivos: en esta parte se lee un archivo con extensión y estructura xml en el cual se limitan las siguientes etiquetas: matrices: este será necesario para la lectura inicial del archivo, ya que será la etiqueta padre de todo.
   1. biblioteca: esta etiqueta será la que indica que vendrán un conjunto de canciones, las cuales se añadirán a nuestro reproductor de música:
   2. canción: esta etiqueta únicamente podrá estar entro de la etiqueta matriz y contendrán los valores respectivos a cada las canciones, cada canción contiene los siguientes atributos
      1. nombre: indicara el nombre de cada canción
      2. Artista: indica quien es el autor de la canción.
      3. Álbum: es el nombre del álbum al cual pertenece la canción.
      4. Imagen: una imagen que representa la canción.
      5. Ruta: la dirección de donde podemos reproducir la canción.



*Figura I.* XML proporcionado para lectura de datos Fuente: elaboración propia, 2021

Después de leer los datos se verifico que estuvieran en el orden correcto y de lo contrario se ordenaron por medio de un ordenamiento burbuja como se muestra en la figura No. 2.



*Figura II. Diagrama de biblioteca*

Fuente: elaboración propia, 2021

Como se observa en la figura II este diagrama muestra el orden en que las canciones son agregadas a la lista de la biblioteca.

Para procesar la información del xml procdemos a hacer un parse mediante la

biblioteca minidom, para obtener todos los campos necesarios para crear un objeto de tipo canción.

1. 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 grafica se hizo usa de Graphviz el cual se instaló dentro de la IDE por medio del pip intall, la grafica que se genera es la de la figura VI.



*Figura VI.* Creación .dot Fuente: elaboración propia, 2021

1. Reproducir: reproduce las canciones, por medio de lista de reproducción, álbum, artista o cancion.

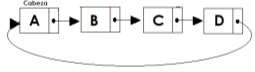
**Conclusiones**

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 navegacion 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 los mismos.
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)

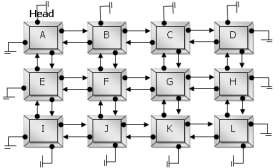
**Apéndice**

Lista circular sencilla:



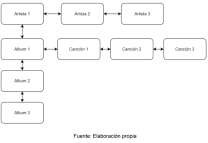
*Apéndice I.* Lista Circular Fuente: elaboración propia, 2021

Lista ortogonal:



*Apéndice II.* Lista Circular Fuente: problemas resueltos de listas ligadas

Lista doblemente enlazada:



Fuente: problemas resueltos de listas ligadas Lista doblemente enlazada: