IU Digital de Antioquia

INSTITUCIÓN UNIVERSITARIA DIGITAL DE ANTIOQUIA

Unidad 1

Introducción a la estructura de datos

Estructura de Datos

ORD No 74 de 2017 · VIGILADA MinEducación

Estructura de Datos

Presentación

Los esquemas de programación de aplicaciones o software con fines específicos, requieren del conocimiento de las estructuras que deben llevar los datos, con la finalidad de realizar aplicaciones completas, organizadas y estructuradas.

Las estructuras de datos ayudan a realizar la manipulación de datos de manera óptima, teniendo en cuenta los métodos de acceso a la información, a través de ordenamientos de búsquedas, organización, métodos de acceso, entre otros. Estas estructuras impactan directamente el almacenamiento de los datos que se encuentran organizados al interior de una base de datos.

- Video de presentación:
  https://youtu.be/5DXYevfTXN4

Objetivos

Objetivo general

Desarrollar aplicaciones de software bajo esquemas de estructura de datos que permitan la manipulación de los datos de manera ordenada.

Objetivos específicos

- Conocer las diferentes herramientas para la generación de aplicaciones bajo el esquema de estructura de datos.
- Conocer las capacidades y funcionalidades al aplicar la estructura de datos al entorno del mundo real.
- Aplicar correctamente las diferentes estructuras de datos existentes, en el desarrollo de software.
- Generar aplicaciones de software que contengan las estructuras de datos más conocidas y que permitan resolver problemas del mundo real.

Mapa del curso

Esta asignatura te permitirá comprender las estructuras de datos existentes para el desarrollo de software, bajo esquemas de pilas, árboles, colas, vectores, matrices, listas, entre otros.

¡Muchos éxitos durante tu proceso de aprendizaje!

Unidad 1. Introducción a la estructura de datos

Las estructuras de datos son fundamentales para la programación, especialmente, para el paradigma de la programación orientada a objetos. Ellas permiten manipular grandes volúmenes de información de una manera impecable, ordenada y eficiente.

Básicamente, todos los lenguajes de programación hacen uso de las estructuras de datos, y de ahí su importancia para manipularlas adecuadamente.

En la *Unidad 1. Introducción a la estructura de datos*, estudiaremos nueve saberes esenciales que estarán asociados con las actividades de aprendizaje, las cuales han sido diseñadas para fortalecer los conocimientos adquiridos en tu proceso de formación.

1. Tipos de datos.
2. Definición y concepto de estructura.
3. Tipos de estructura de datos.
4. Operaciones con estructura de datos.
5. Tipos de datos abstractos.
6. Definición e implementación de un TDA.
7. Arreglos unidimensionales.
8. Arreglos multidimensionales.
9. Listas.

Tipos de datos

En programación, se tienen dos tipos de datos: simples y estructurados.

Simples o primitivos:

Los datos simples o primitivos son aquellos que, por medio de un identificador, ocupan un espacio en memoria con un solo valor.

- No tienen acciones (denominados métodos) asociadas a los valores almacenados.
- Se pueden visualizar como un casillero donde se ubica un mismo tipo de dato, con un tamaño preestablecido.
- Los más utilizados son: entero (integer), real (float, double, decimal), lógico (boolean), carácter (string, char).

Estructurados:

Caracterizan una forma de representar información, incorporan un grupo de datos o valores almacenados bajo un mismo identificador.

En términos generales, pueden ser: arreglos (arrays), listas (list), pilas (stack), árboles (struct).

Definición y concepto de estructura

Toda la información que se maneja en un programa se encuentra almacenada en la memoria del sistema.

La complejidad de los sistemas de información exige un almacenamiento, consulta y manipulación eficiente de la información. Estos métodos son conocidos como estructuras de datos.

En la estructura de datos, la información se dispone de una forma determinada, legible y más completa.

Esto propicia relaciones entre los datos que la constituyen, y establece operaciones posibles en los datos.

En un principio, se puede visualizar como una tabla, donde se tiene una casilla para el identificador secuencial del dato (índice) y otra para el dato (elemento), como se muestra a continuación:

Índice	Elemento
1	“alfa”
2	“beta”
3	“gamma”

Conclusión:

Las estructuras de datos representan formaciones particulares y eficientes de almacenar y maniobrar los datos. Se pueden dar de forma general para cualquier aplicación o, específica, para un problema dado (Vilanova & Lagonigro, 2010).

Tipos de estructura de datos

Conoce los tipos fundamentales para las estructuras de datos

Estáticas

En las estructuras de datos estáticas, se tiene una cantidad fija de elementos, donde usualmente se define el tamaño desde el inicio del programa y, por ende, el espacio en memoria es constante. En este tipo de datos se encuentran los arreglos.

Dinámicas

Es posible clasificar las estructuras de datos dinámicas en no lineales y lineales.

- **No lineales:**
  Para las estructuras de datos no lineales, un elemento puede estar relacionado con más de uno, bien sea de forma posterior o anterior. En este tipo se encuentran: árboles y grafos.

- **Lineales:**
  Una estructura de datos es lineal si sus elementos se encuentran distribuidos en una secuencia. En este tipo se encuentran: listas, colas pilas.

Para ampliar la información, ingresa al enlace, descarga el libro y lee el Capítulo 2.3 del libro ***Proyecto Latín***.

Operaciones con estructuras de datos

- **Adicionar datos en la estructura:** esta acción u operación, permite añadir nuevos registros a la estructura. Según el tipo de estructura, se puede ubicar en una posición determinada o en la última disponible.

- **Borrar datos en la estructura:** operación de borrado de un registro de la estructura. Al igual que la acción de adicionar, se puede dar a partir de una posición específica, en la primera o última posición disponible.

- **Búsqueda de datos en la estructura:** implica la localización de un registro caracterizado por una determinada posición. Permite el acceso a los registros que cumplan con una o más condiciones dadas. Búsqueda de datos en la estructura: implica la localización de un registro caracterizado por una determinada posición. Permite el acceso a los registros que cumplan con una o más condiciones dadas.

Tipos de datos abstractos

Los tipos de datos abstractos, en adelante TDA o en ADT por sus siglas en inglés (*Abstract Data Type*), se definen como un grupo de datos que cumplen condiciones específicas, características definidas, o datos que los conforman con un conjunto de operaciones, métodos o algoritmos, que manipulan dichos datos; esto establece el comportamiento.

De esta forma, un TDA tendrá una identidad y comportamiento específico, lo cual permite interactuar con los datos.

Se dice que son abstractos porque solo es conocido su comportamiento, más no su funcionamiento interno. Es decir, sabemos qué hace, más no cómo lo hace; esta

abstracción de datos facilita su comprensión y uso. Los TDA se usan fundamentalmente en la programación orientada a objetos.

Para ampliar este tema, ingresa al siguiente enlace:

http://www.cs.princeton.edu/courses/archive/fall13/cos126/lectures/09-DataTypes-2x2.pdf

Definición de implementación de un TDA

Para definir un nuevo TDA, se debe analizar el problema que se busca solucionar, con el fin de plantear y planificar la estructura de datos que puede ayudar a solucionar el caso en cuestión.

Se requiere definir las características principales, analizando los tipos de datos requeridos y las operaciones (métodos - funciones o procedimientos) que se pueden utilizar para la estructura (Joyanes, 2007). Completado este paso, se implementa el TDA, es decir, se codifica o programa en un lenguaje de programación cualquiera.

Ejemplo:

Se requiere una estructura de datos que permita almacenar las medidas de un triángulo. Muestra un mensaje donde se valide si dicho triángulo es equilátero y, adicionalmente, permita calcular el perímetro.

Para este caso, se identifican los datos que se requieren: medidas de los lados.

Como operaciones, se identifican: calcular el perímetro y comparar los lados para mostrar el mensaje respectivo.

Descarga el siguiente documento para estudiar la implementación en código Java.

Actividad - Mapa mental 1

Teniendo en cuenta lo estudiado hasta el momento, realiza cuatro mapas mentales que resuman cada uno de los conceptos aprendidos.

Es decir, debes hacer un mapa mental por tema:

1. Tipos de datos.
2. Tipos de estructura de datos.
3. Operaciones con estructuras de datos.
4. Tipos de datos abstractos.

Te recomendamos hacer uso de textos cortos, imágenes y otros recursos.

Arreglos unidimensionales

Los arreglos o array, se definen como un conjunto de datos los cuales, dependiendo del lenguaje de programación, pueden ser del mismo tipo o pueden ser heterogéneos; los valores almacenados, llamados elementos, son adyacentes y se identifican por el nombre del arreglo, seguido de un índice o subíndice que indica la posición dónde se ubican dentro del arreglo. Los valores son almacenados en un área contigua de memoria interna.

Usualmente, el total de elementos del arreglo se indica cuando se crea este; sin embargo, existen lenguajes de programación que manejan los arreglos de forma dinámica, es decir, no es necesario indicar el total de elementos y su tamaño puede aumentar en tiempo de ejecución (Joyanes, 2001).

Mediante índices, se puede acceder directamente a un dato o valor, y al utilizar solo un índice, se dice que el arreglo tiene una dimensión, y también es llamado vector.

Algorítmicamente, un arreglo unidimensional se puede definir de la siguiente forma:

Definir Edades(4) como entero

En la línea anterior, se ha creado un nuevo espacio en memoria llamado edades, todos los datos que allí se almacenaran serán tipo entero. Se tiene la capacidad de almacenar como máximo 5 valores.

Para acceder a cada posición, se debe indicar el identificador de dicha posición al momento de realizar cualquier operación: asignación, modificación.

El siguiente ejemplo muestra cómo asignar a cada posición un nuevo y diferente valor:

Edades(0)=15

Edades(1)=19

Edades(2)=5

Edades(3)=44

Este tipo de estructura es útil en el momento que es necesario almacenar y procesar un mismo tipo de datos. Un ejemplo de esto puede ser lo siguiente:

Se requiere capturar las notas finales de un grupo de 75 personas, para conocer cuántas están por encima del promedio de todo el grupo.

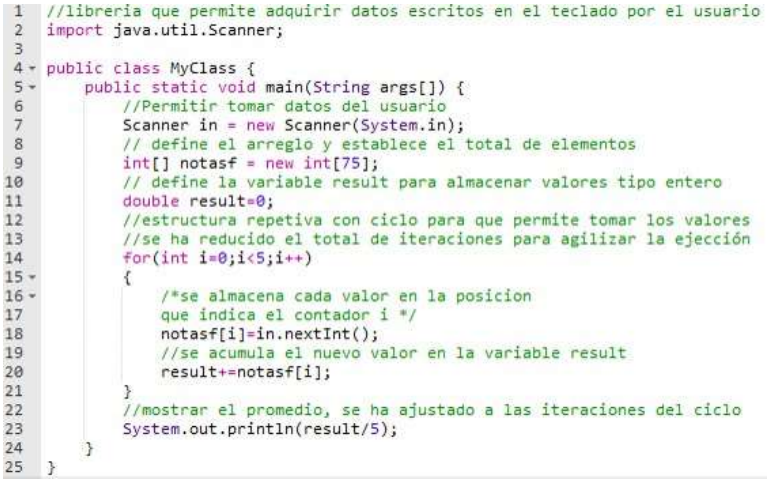
Para este caso es conveniente un arreglo, de manera que todas las notas finales se encuentren almacenadas bajo un mismo identificador y se pueda procesar dato por dato de una forma conveniente y rápida.

El caso se puede resolver por medio del siguiente pseudocódigo:

Pseudocódigo

```
Proceso ejemplo2
        Inicio
        Definir notas(75) como entero
        Definir result como real
        Para i = 0 hasta 75
        Leer notasf(i)
            result = result + notasf(i)
        Fin Para
        Mostrar result/75
    Fin
Fin ejemplo2
```

El anterior pseudocódigo se puede implementar en Java de la siguiente manera:



```java
//libreria que permite adquirir datos escritos en el teclado por el usuario
import java.util.Scanner;

public class MyClass {
    public static void main(String args[]) {
        //Permitir tomar datos del usuario
        Scanner in = new Scanner(System.in);
        // define el arreglo y establece el total de elementos
        int[] notasf = new int[75];
        // define la variable result para almacenar valores tipo entero
        double result=0;
        //estructura repetiva con ciclo para que permite tomar los valores
        //se ha reducido el total de iteraciones para agilizar la ejección
        for(int i=0;i<5;i++)
        {
            /*se almacena cada valor en la posicion
            que indica el contador i */
            notasf[i]=in.nextInt();
            //se acumula el nuevo valor en la variable result
            result+=notasf[i];
        }
        //mostrar el promedio, se ha ajustado a las iteraciones del ciclo
        System.out.println(result/5);
    }
}
```

Como se observa en la imagen, por lo general, el manejo de arreglos viene acompañado de ciclos: mientras o para. Al interior del ciclo, en cada iteración, se pueden realizar los cálculos o validaciones que sean necesarios.

Las posiciones del arreglo se pueden indicar explícitamente (notasf[1], es decir, segundo elemento del arreglo notasf) o pueden ser indicadas por medio de una variable, típicamente un contador.

Amplía la información sobre el uso de arreglos en Java, en la página de documentación de Java.

Actividad - Ejercicios de práctica 1

Teniendo en cuenta lo estudiado hasta el momento en la Unidad 1, elabora el pseudocódigo para los siguientes casos:

- Definir dos arreglos unidimensionales que permitan almacenar 7 números ingresados por el usuario.
- Llenar un tercer arreglo con la diferencia de los dos primeros arreglos. Mostrar el promedio de todos los datos.
- Mostrar todos y cada uno de los datos del tercer arreglo.
- Implementar el pseudocódigo en un lenguaje de programación como Java o C++.

Arreglos multidimensionales

Para este tipo de arreglos se hace referencia a cada elemento por medio de dos o más índices. Tienen la misma finalidad de un arreglo unidimensional, pero buscan almacenar los valores en forma de tabla.

Para este tipo de arreglos, se hace referencia a cada elemento por medio de dos o más índices. Tienen la misma finalidad de un arreglo unidimensional, pero buscan almacenar los valores en forma de tabla.

Los arreglos multidimensionales más comunes son aquellos que utilizan dos índices, estructurando la información en forma matricial, o sencillamente como una tabla de datos.

En este caso, un índice indica fila y el otro, indica columna. Los arreglos bidimensionales son llamados matrices.

Algorítmicamente, un arreglo unidimensional se puede definir de la siguiente forma:

Definir Edades(3)(2) como entero

Para acceder a cada posición, se deben indicar los identificadores de dicha posición en fila y columna al momento de realizar cualquier operación: asignación, modificación.

El siguiente ejemplo muestra cómo asignar a cada posición un nuevo y diferente valor:

Edades(0)(0)=11

Edades(0)(1)=33

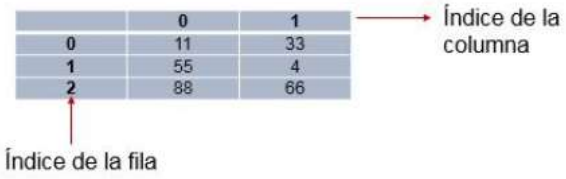
Edades(1)(0)=55

Edades(1)(1)=4

Edades(2)(0)=88

Edades(2)(1)=66

En este caso, el arreglo se puede visualizar de la siguiente forma:



Este tipo de estructuras se puede aprovechar si a cada índice se le asocia un concepto para almacenar los valores, es decir, los datos en las columnas tienen un significado, mientras que los datos en las filas tienen otro.

Por ejemplo, en una matriz se pueden almacenar las notas de diferentes materias para un grupo de estudiantes. Esto será posible dado que cada fila representara a un estudiante, mientras que las columnas representaran una materia diferente, cada una.

Para el pseudocódigo del ejemplo, se requieren dos ciclos anidados; el primero controla el desplazamiento de las filas, y el segundo el desplazamiento de las columnas.

Pseudocódigo

```
Proceso ejemplo3
        Inicio
        Definir notasf(3)(5) como entero
        Definir result como real
        Para i = 0 hasta 3
            Para j = 0 hasta 5
                Leer notasf(i)(j)
                result = result + notasf(i)(j)
            Fin Para
        Fin Para
        Mostrar result/15
    Fin
Fin ejemplo3
```

El anterior pseudocódigo se puede implementar en Java de la siguiente forma:

```java
import java.util.Scanner;
public class MyClass {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        int notasf[][] = new int[3][5];
        double result=0;
        for(int i=0;i<3;i++)
        {
            for(int j=0;j<5;j++)
            {
                notasf[i][j] = sc.nextInt();
                result += notasf[i][j];
            }
        }
        System.out.println(result/15);
    }
}
```

Como se muestra en el ejemplo, para almacenar un valor en el arreglo bidimensional, se utilizan dos índices. Para el caso de tres dimensiones, la estructura puede visualizarse como un cubo. Java soporta hasta 14 índices para un arreglo multidimensional.

Listas

Una lista es un conjunto de datos donde cada elemento tiene un único antecesor y un único sucesor y la cantidad total de elementos es variable.

Te invitamos a conocer más sobre este concepto en el recurso que presentamos a continuación.

- **Listas**
  Una lista es un conjunto de datos donde cada elemento tiene un único antecesor y un único sucesor y la cantidad total de elementos es variable; es decir, se guarda la posición en que se hallan el siguiente y el anterior elemento de la lista. Se puede concebir como una cadena. Frecuentemente, cada elemento se denomina nodo, el cual incluye el valor y la referencia a otra posición.

- **Listas simples**
  Una lista simple es una estructura de datos en la cual cada elemento se enlaza o apunta al próximo elemento. En este caso, cada nodo contiene información de la posición actual y una referencia a la posición siguiente.

Cada nodo está formado por un dato (D) y una referencia al siguiente nodo(n).

Particularmente, la primera posición no tiene dato, solo referencia al siguiente nodo y representa la cabecera o punto de inicio de la lista. La última posición no tiene referencia a un siguiente nodo, su valor es nulo (null).

- **Listas dobles**
  En las listas dobles, cada nodo tiene dos referencias: la primera referencia apunta al nodo anterior, y la otra apunta al nodo siguiente.

Cada nodo está formado por un dato (D), una referencia al siguiente anterior (ra) y una referencia al siguiente nodo (rs).

Particularmente, el primer nodo en su primera referencia (ra) apunta nulo y el último nodo en su segunda referencia (rs) apunta a nulo.

- **Listas circulares**
  En las listas circulares, se tiene una estructura similar a la lista simple, pero el último nodo apunta al primero, entonces cada nodo siempre tiene un antecesor y sucesor.

Para implementar todas estas listas en un lenguaje de programación, se requieren dos clases, una que almacene el dato y otra que permita manejar referencias.

Glosario

A

Algoritmo

Conjunto ordenado de operaciones sistemáticas que permite hacer un cálculo y hallar la solución de un tipo de problemas.

Arreglos

Zona de almacenamiento contiguo que contiene una serie de elementos del mismo tipo.

C

Ciclos

Sentencia que ejecuta repetidas veces un trozo de código, hasta que la condición asignada a dicho bucle, deja de cumplirse.

Clase

Plantilla para la creación de objetos de datos según un modelo predefinido.

D

Datos simples

Representan un solo valor.

E

Estructura de datos

Forma particular de organizar datos en una computadora, para que puedan ser utilizados de manera eficiente.

J

JAVA

Lenguaje de programación de propósito general, concurrente, orientado a objetos, que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible.

M

Métodos

Subrutina cuyo código es definido en una clase y puede pertenecer tanto a una clase como a un objeto.

O

Objeto

Unidad dentro de un programa de computadores que consta de un estado y de un comportamiento, que, a su vez, constan respectivamente de datos almacenados y de tareas realizables durante el tiempo de ejecución.

Bibliografía

- Gutiérrez, X. F. (2004). *Estructuras de datos: especificación, diseño e implementación.* Ediciones UPC.

- Joyanes, A. & Zahonero, M. (2005). Programación en C: metodología, algoritmos y estructura de datos (2ª. ed.). http://ebookcentral.proquest.com/lib/iudasp/detail.action?docID=3195036

- Joyanes, A. & Zahonero, M. (2007). *Estructura de datos en C++.* McGraw-Hill. http://ebookcentral.proquest.com/lib/iudasp/detail.action?docID=3194764

- Juganaru, M. (2014). *Introducción a la Programación.* Grupo Editorial Patria. http://ebookcentral.proquest.com/lib/iudasp/detail.action?docID=3227908

- Proyecto Latín. (2014). *Estructuras de Datos*. LaTIn. https://openlibra.com/es/book/estructuras-de-datos

- Sáez, X. (2011). *Prácticas de programación. Estructuras de datos básicas: secuencias.* Eureca Media.

- Vilanova, R. & Lagonigro, R. (2010). *Tipos estructurados de datos: Tablas y tuplas*. UOC. https://openlibra.com/es/book/tipos-estructurados-de-datos-tablas-y-tuplas

IU Digital de Antioquia
INSTITUCIÓN UNIVERSITARIA DIGITAL DE ANTIOQUIA

ORD No 74 de 2017 - VIGILADA MinEducación

www.iudigital.edu.co