Estructura de Datos y Algoritmos 1

Tema 01

- Definiciones importantes

Dato: Valor de un atributo de un objeto

Información: Significado que le doy al dato en un contexto determinado.

Estructura de datos: Colección de datos en un conjunto determinado organizado de alguna manera que facilita su manipulación.

1. Estructuras de datos

Estructura de datos + Algoritmos = Programas

Un nodo es la referencia básica de la estructura. nodo ⟹ hoja

⚠ → ¿Qué se entiende por manipulación? ⚠

- Inserción
- Eliminación
- Búsqueda
- Actualización.

• Clasificación de la estructura de datos.

Según su tipo → HOMOGÉNEAS (MISMO TIPO)
→ HETEROGÉNEAS (DISTINTO TIPO)

Según su organización → LINEALES 1:1 → LISTA ENLAZADA (SUCESOR Y ANTECESOR)
→ JERÁRQUICAS 1:N → PADRE-HIJO (NODO RAÍZ Y NODOS HOJA)
→ RED (GRAFOS) N:M → GRAFOS

Según su gestión de memoria → ESTÁTICAS } ESTÁTICAS
→ DINÁMICAS } ESTRUCTURADAS
→ ELÁSTICAS } ESTRUCTURADAS

Según su soporte de almacenamiento → MEMORIA PRINCIPAL (RAM)
→ MEMORIA SECUNDARIA

⚠ → ¿Cual es la diferencia entre dinámica y elástica según su gestión de memoria? ⚠

- Dinamica: Se asigna memoria en tiempo de ejecución y el tamaño puede cambiar, pero ese tamaño es cambiado explícitamente
- Elástica: Igual que la dinámica pero la gestión del cambio de tamaño es automática

⚠ → ¿Qué parámetro duplica el tamaño del array en Java? ⚠

```
newCapacity = oldCapacity + (oldCapacity >> 1);
```

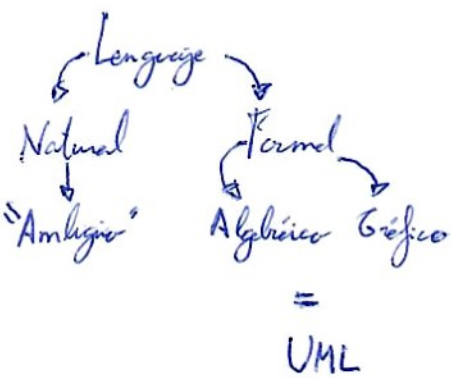
2. Algoritmos

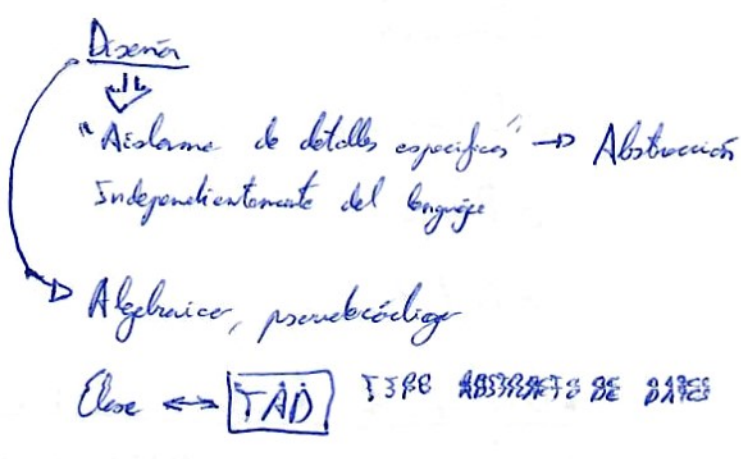
Un algoritmo es una secuencia ordenada de pasos diseñada para la resolución de un problema determinado y exenta de ambigüedad. No deja de ser un método de cálculo.

Importante ⚠ → Un algoritmo debe ser preciso, finito y definido.

- Preciso: Exento de ambigüedad
- Finito: En algún momento debe terminar
- Definido: La entrada es igual a la salida

¿Qué notaciones tenemos para describir algoritmos? ⚠





Los lenguajes se clasifican en paradigmas

Paradigma	
Estructurado	O.O.
Pascal / C	C++ / Java / C#

Dado un problema debemos elegir la estructura de datos más eficiente para resolver dicho problema

3. Tipo abstracto de datos (TAD)

Conjunto de datos y descripción de operaciones definidas sobre ellos mediante una especificación formal, independiente de cualquier detalle de implementación.

TAD formado por especificación e implementación.

Especificación → Sintaxis + semántica → Visible al usuario

Implementación → Representación + Algoritmos → Oculto

4. Especificación de un TAD

Se trata de definir el TAD sin ambigüedades.

Esta sección no es muy relevante

5. Análisis de algoritmos

Siempre debemos analizar el problema con detenimiento para elegir la estructura de datos más eficiente

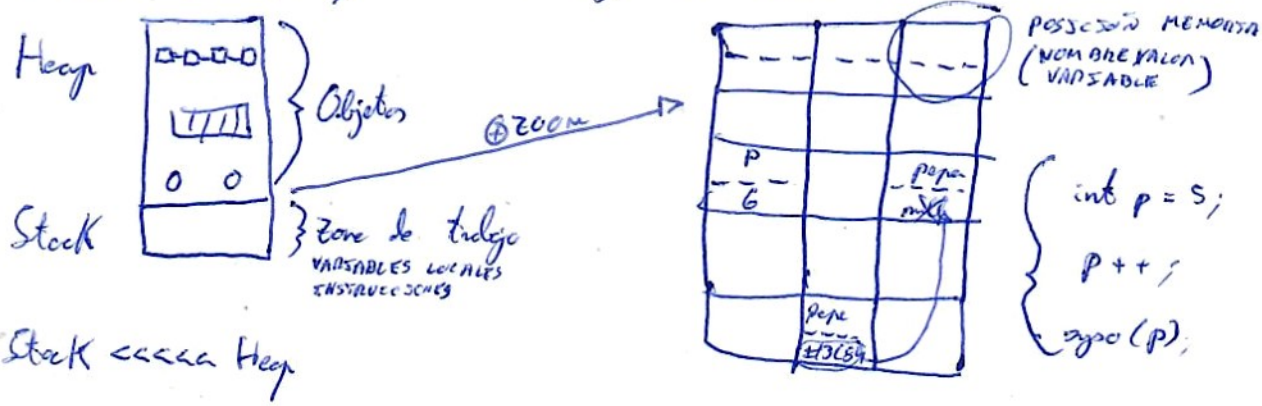
Tenemos algoritmos iterativos y recursivos en

Dentro del análisis de algoritmos podemos encontrar el tiempo de ejecución, la ocupación de la memoria, otros factores y la influencia de la estructura de datos

Es importante recordar el orden $O(1) < O(\log n) < O(n) < O(n \log n) < O(n^2)$

Gestión de Memoria (JAVA)

Podemos ver la pila de la siguiente manera



Stack <<<<< Heap

Cada variable primitiva tiene sus operaciones

int, float, double, Integer, Float, Double, String

ArrayList<Integer> pepe = new ArrayList<>();

Todo se almacena en una dirección de memoria del objeto en concreto

pepe.add(...); ///Acceder

pepe apunta a un conjunto o arraylist, pepe es un puntero a ArrayList

ArrayList<Integer*> pepa = pepe → Referencias que apuntan al mismo sitio/Objeto.

pepa.clear();
sysp(pepe);

pepe → □ ← pepa ArrayList vacío

Para evitarlo, NO utilizamos clone. Hacemos uso de constructor copia.

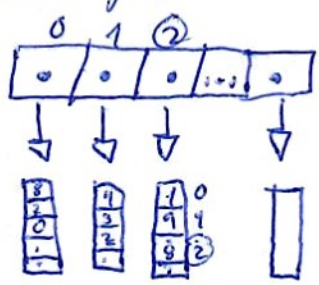
ArrayList<Integer> pepa = new ArrayList(Pepe);

Si hacemos que pepa apunte a pepe pepe = pepe, ahora ambas serán sinónimos

ArrayList<ArrayList<Integer>> ant = new ArrayList<>() → Matriz

(Importante dibujar estructuras siempre)

ant apunta a un ArrayList



Para acceder → ant.get(2).get(2) ⟹ 8