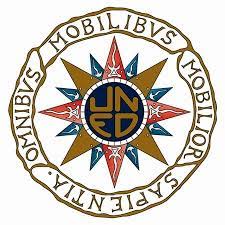


*Grado en Ingeniería Informática*

**Memoria de práctica de Estrategias de Programación y Estructuras de Datos**

*“2022- 2023”*



Escuela Técnica Superior de Ingeniería Informática- UNED

***Autores:*** *Sergio Sánchez LópeZ*

***Tutores****: Roberto Barchino Plata*

# Anotaciones previas

* El formato original del documento es Word. En este formato se dispone de desplegables para facilitar su lectura y anotaciones para ver las diferentes referencias bibliográficas de una forma más directa.

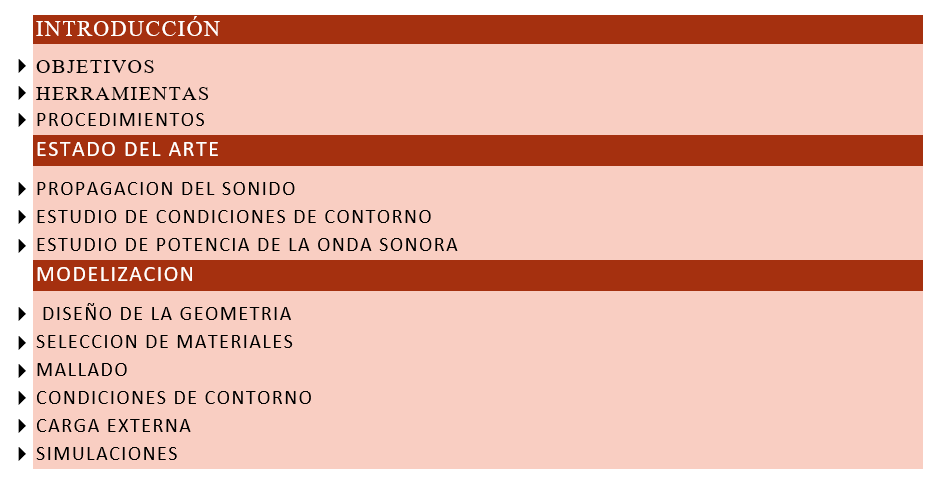
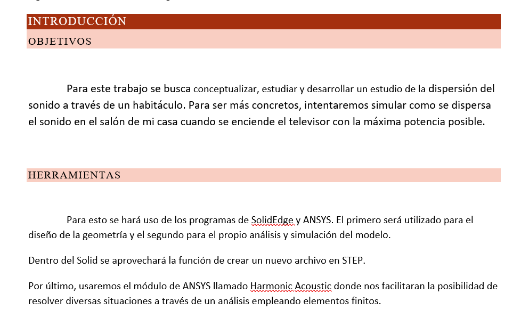


Fig. Imagen ilustrativa de formato original en Word.

# ÍNDICE

1. Introducción
   1. Presentación del problema
   2. Ponderación
   3. Forma de entrega
2. Estado del arte
   1. Estructuras básicas de datos
      * Secuencias
      * Árbol general
3. Planteamiento de la práctica
   1. Stock
      * tipo de Datos Abstracto
      * Preguntas teóricas
   2. Implementación 1
      * secuencia ordenada de pares indexados
      * Preguntas teóricas
   3. Implementación 2
      * Árbol general
      * Preguntas teóricas
4. Diseño de la práctica
   1. StockIF.java
   2. StockPair.java
   3. StockSequence.java
   4. StockTree.java
   5. Clases:
      * Node.java
      * RootNode.java
      * InnerNode.java
      * InfoNode.java
   6. Clase Main.java
      * Funcionamiento
5. Implementación
   1. Pasos
6. Ejecución y juegos de prueba
7. Estudio del coste
   1. Preguntas teóricas
8. Conclusiones
9. Bibliografía

# Introducción

## pRESENTACIÓN DEL PROBLEMA

Según el diccionario, se define stock como aquel conjunto de mercancías o productos que se tienen almacenados en espera de su venta o comercialización. Desde el punto de vista de los Tipos Abstractos de Datos, podríamos pensar en un stock como en una estructura indexada que almacena valores enteros no negativos (las unidades almacenadas de cada producto) a los cuales se accede mediante índices que se expresan con una cadena de caracteres, que representarían los productos.

Este tipo de estructuras indexadas suelen implementarse mediante *tablas hash*, las cuales se estudiarán en la asignatura de Programación y Estructuras de Datos Avanzadas, que sigue a la nuestra durante el primer cuatrimestre de segundo curso. Así pues, en nuestra práctica emplearemos otras estructuras para implementar un stock.

En esta práctica consideraremos *dos implementaciones alternativas* de un stock en la que almacenaremos los productos y el número de unidades que están almacenadas. **La primera implementación utilizará una secuencia de pares** , mientras que **la segunda empleará un árbol general** en el cual los valores se almacenarán en los nodos hoja y el índice que lleva a cada valor se obtendrá mediante el recorrido desde la raíz del árbol al nodo hoja correspondiente.

## PONDERACIÓN

La práctica supone un 20% de la calificación de la asignatura, y es necesario aprobarla para superar la asignatura. Además, será necesario obtener, al menos, un 4 sobre 10 en el examen presencial para que la calificación de la práctica sea tenida en cuenta de cara a la calificación final de la asignatura.

La calificación de la práctica será otorgada por el tutor , en función de las respuestas a las preguntas teóricas, la superación del juego de pruebas público y del juego de pruebas privado y el diseño y resultados del estudio empírico de costes. También se podrá tener en cuenta el grado de participación activa en las sesiones de tutorización relativas a la práctica. La puntuación se distribuye de la siguiente forma:

* 5 puntos en función del código de la práctica: corrección, presentación, documentación. Para aprobar es imprescindible que el código supere el juego de pruebas privado que sólo tendrán los tutores y tutoras.
* 2 puntos en función del estudio empírico del coste. Se valorará el diseño del juego de pruebas y la presentación y análisis de los resultados empíricos.
* 3 puntos en función de las respuestas a las preguntas teóricas. Se valorará la corrección de las respuestas, su justificación y el nivel de detalle.
* Hasta un punto adicional en función de la participación e interacción en las tutorías del centro orientadas a la práctica

La práctica es un trabajo individual de cada estudiante. **Tanto el código como las respuestas a las preguntas teóricas y el diseño del juego de pruebas individual para el estudio empírico del coste deben ser originales de cada estudiante.** Si se detectan casos de plagio en cualquiera de estos aspectos se suspenderá a los estudiantes involucrados y podrán ser objeto de sanciones por parte de la universidad.

## FORMA DE ENTREGA

Tanto el plazo de entrega como la forma en la que se entrega, son establecidos por cada tutor/a de forma independiente en cada centro Asociado, por lo que deberían ser consultados

La documentación que debe ser entregada es la siguiente

* Memoria de práctica, en la que se deberán responder a las preguntas teóricas y se incluirá el diseño y resultados del estudio empírico de costes
* Implementación en Java de la práctica, de la cual se deberá aportar únicamente el **directorio src del proyecto Java.**
* Juego de pruebas diseñado para el estudio empírico de costes.

Adicionalmente se deberá de entregar una copia de la práctica al Equipo Docente a través del enlace que se habilitará en el Curso Virtual en las siguientes fechas:

* Desde el 01/05/2023 al 16/06/2023 para la convocatoria de junio.
* Desde el 15/07/2022 al 10/09/2023 para la convocatoria de septiembre.

Nota: la entrega de la copia de la práctica se realiza al equipo docente, no al tutor o tutora que se encargará de su evaluación. Si no se entrega la práctica al tutor o tutora, ésta no se evaluará y constará como suspensa.

# ESTADO DEL ARTE

## Estructuras de datos básicas

Una estructura de datos permite establecer una forma particular de almacenar y organizar información

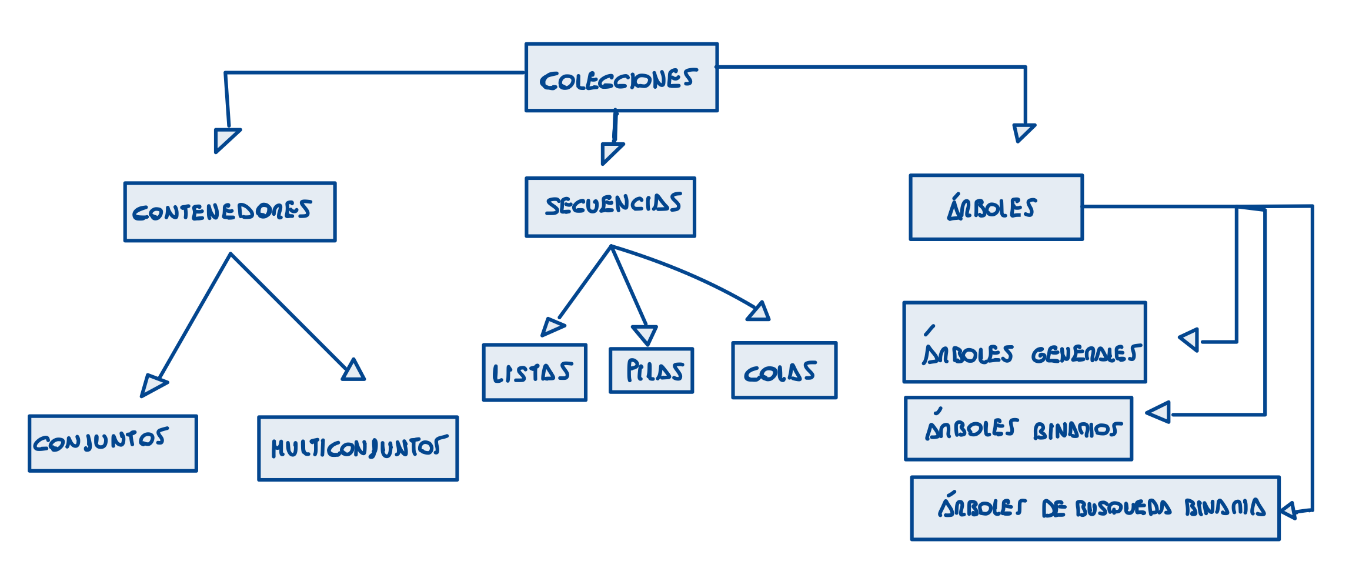


Fig. esquema de las estructuras básicas de datos

##### SECUENCIAS

Es una colección donde los datos forman una secuencia, es decir, que los elementos están ordenados en un orden lineal explícito.

Entre ellos existen una serie de relaciones de adyacencia de antecesión y sucesión, es decir, que salvo el primer elemento, todo elemento tiene un elemento antecesor y , salvo el último, todo elemento tiene un elemento sucesor.

* Esta organización no condiciona la forma en la que se representa la secuencia
* No existe un orden que dependa de los valores concretos de cada uno de los elementos
* En la secuencia tal cual, no existe un orden implícito, aunque las listas podrían implementar un orden implícito.

En cuanto a las operaciones a poder realizar podemos recorrer los elementos , es decir, hacer un iterador de la secuencia que empiece por el primer elemento de la secuencia y que llegue hasta el último.

##### ÁRBOL GENERAL

Este tipo de estructura (caso particular de grafo) , es una colección donde los datos no están almacenados de manera lineal, donde se crea una jerarquía entre ellos. En este caso debemos tener claro que el elemento Padre es el predecesor de cualquier elemento y el sucesor de este se denomina elemento Hijo.

En los árboles generales, el número de hijos no esta acotado y puede haber casos donde pueden haber cero hijos.

En Cuanto a las operaciones a poder realizar serían las siguiente:

* Modificar la raíz del árbol
* Obtener todos los hijos del árbol
* Obtener un hijo en concreto
* Insertar un hijo en una posición concreta del árbol
* Eliminar un hijo de una posición concreta (para esto debe existir el hijo)

# PLANTEAMIENTO DE LA PRÁCTICA

La práctica consiste en :

1. Razonar e implementar dos soluciones alternativas para el tipo de datos abstracto stock, una basada en secuencias y otra basada en árboles generales. **La implementación utilizará los tipos de datos programados por el Equipo Docente.**
2. Implementar un programa que aplique ambas implementaciones.
3. Ejecutar el programa anterior sobre juegos de prueba empleando ambas implementaciones y realizar estudios comparativos del tiempo de ejecución de cada una de ellas.

## STOCK

##### TIPO DE DATOS ABSTRACTO

El tipo de datos abstracto Stock que vamos a implementar funciona como una estructura indexada mediante índices formados por cadenas de caracteres. Así pues, este tipo abstracto de datos incluye las operaciones descritas en la siguiente interfaz:

**package** es.uned.lsi.eped.pract2022\_2023;

**import** es.uned.lsi.eped.DataStructures.SequenceIF;

/\* Representa un Stock en el que se indexan valores enteros

\* bajo un índice formado por una cadena de caracteres

\*/

**public** **interface** StockIF {

/\*\* Devuelve el valor indexado bajo el índice p.

\* **@PRE**: p != ""

\* Si no existe un valor indexado bajo el índice p,

\* devuelve el valor -1.

\*/

**public** **int** retrieveStock(String p);

/\*\* Indexa el valor u bajo el índice p.

\* **@PRE**: p != "" y u >= 0

\* Si ya había un valor bajo el mismo índice,

\* el nuevo valor sustituye al anterior.

\*/

**public** **void** updateStock(String p, **int** u);

/\* Devuelve una secuencia de todos los pares <p,u>

\* presentes en el stock tales que:

\* -El valor indexado bajo el índice p es u

\* -El índice p comienza por la cadena prefix

\* Además, la secuencia deberá estar ordenada según

\* el orden alfabético de los productos

\*/

**public** SequenceIF<StockPair> listStock(String prefix);

}

Las implementaciones de StockIF utilizarán la siguiente clase:

**package** es.uned.lsi.eped.pract2022\_2023;

// Representa un par <producto,unidades> del Stock

**public** **class** StockPair {

**private** String producto;

**private** **int** unidades;

//Constructor

**public** StockPair(String p, **int** u) {

**this**.producto = p;

setUnidades(u);

}

/\*\* Modifica el número de unidades del par

\* **@PRE**: u >= 0

\*/

**public** **void** setUnidades(**int** u) {

**this**.unidades = u;

}

//Devuelve el nombre del producto

**public** String getProducto() {

**return** **this**.producto;

}

//Devuelve el número de unidades

**public** **int** getUnidades() {

**return** **this**.unidades;

}

//Construye una cadena para representar el par

**public** String toString() {

**return** "("+**this**.producto+","+Integer.*toString*(**this**.unidades)+")";

}

}

##### PREGUNTAS TEÓRICAS

Antes de continuar , reflexiona y responde a la siguiente pregunta:

1. ¿Qué tipo de secuencia sería la más adecuada para devolver el resultado de la operación listado(prefix)? ¿Qué consecuencias tendría el uso de otro tipo de secuencias? Razona tu respuesta.