

Algoritmos y Estructuras de Datos I

**PROYECTO INDIVIDUAL – PARTE II**

Bettina Carrizo

Contenido

[Introducción 3](#_Toc483246910)

[Problema planteado 3](#_Toc483246911)

[Análisis de alternativas 5](#_Toc483246912)

[Alternativas de combinación: 6](#_Toc483246913)

[Alternativa I: 6](#_Toc483246914)

[Alternativa II: 6](#_Toc483246915)

[Alternativa III: 6](#_Toc483246916)

[Algoritmos 6](#_Toc483246917)

[Selección y justificación de alternativa a implementar 7](#_Toc483246918)

[Conclusiones 8](#_Toc483246919)

[Guía del usuario 8](#_Toc483246920)

# Introducción

El presente documento tiene como objetivo abordar el análisis de la implementación de un sistema de control de Stock, requerido por UCUPharma, cadena internacional de Farmacias, cuyos requerimientos fueron modificados por parte de la empresa contratante, con el fin de satisfacer nuevas necesidades de la misma.

En el mismo, se plantean alternativas pertinentes a las modificaciones en cuestión, y se elige una, a nuestro parecer la más eficiente, para que sea utilizada en la implantación. En todo momento se explicará claramente las ventajas y desventajas de cada una, en términos de tiempo de ejecución y consumos de memoria, y finalmente se concluirá con una justificación de la alternativa seleccionada.

## Problema planteado

Además de las características planteadas en el primer informe, la empresa UCUPharma precisa un programa más completo para la gestión integral de su empresa en cuanto a stock se refiere. Por ello, detallamos debajo, a parte de las funcionalidades ya solicitadas, las que serán agregadas en estas instancias:

**Funcionalidades iníciales:**

* **Alta y Baja de productos:**

Ingresar y eliminar productos de la farmacia.

* **Alta y actualización de productos mediante un archivo .csv:**

Cargar los datos del sistema o actualizar los ya existentes mediante la lectura de un archivo de stock.

* **Ventas:**

Realizar ventas, afectando el stock de los productos y llevando un control de las ventas realizadas.

* **Reintegro de stock:**

Efectuar devoluciones, afectando el stock del producto y la venta realizada.

* **Reporte de ventas:**

Dado un rango de fechas, se obtiene un reporte de las ventas realizadas entre ellas. Para ello se lista un detalle de esas ventas, con totalizadores por cantidad de producto y monto total de cada venta.

* **Búsqueda de productos por código, descripción corta y larga:**

Al realizar una búsqueda se devuelven el/los productos encontrados basándose en el criterio utilizado.

**Nuevas funcionalidades:**

* **Reporte de ventas por producto:**

Dado un producto, listar todas las operaciones de venta del mismo, reflejando el código de producto, la fecha en la cual se realizó la venta, la cantidad de producto vendido y el total facturado en cada venta.

* **Listado de productos por áreas de aplicación:**

Emitir un listado agrupado y ordenado por áreas de aplicación. El mismo deberá estar indentado; el nombre de área de aplicación empieza en la columna 1 y los productos correspondientes en la 5. Dentro de cada área de aplicación los productos deben listarse ordenados en forma ascendente por descripción corta.

* **Reportes de movimientos de stock por áreas de aplicación:**

Dada un área de aplicación, listar los movimientos de todos los productos del área, tanto compras como ventas, con identificador, nombre y cantidad de cada operación.

* **Listado de vencimientos próximos al año:**

Dado un año, emitir un listado con todos los productos existentes en stock que vencen en el mismo.

* **Promedio de ventas mensuales por producto:**

Dado cierto producto, indicar el promedio de ventas mensuales, desde que se comenzó a vender el mismo.

* **Eliminación de producto:**

Dado un producto, eliminarlo del stock completamente, para que no sea posible su venta a futuro.

* **Eliminación de áreas de aplicación:**

Dada un área de aplicación, eliminar la misma de la categoría de productos a vender, impactando sobre todas las estructuras existentes.

**Consideraciones importantes:**

En esta parte del proyecto no existen nuevas consideraciones importantes, pero si destacamos el formato en el cual se presentan los productos de la farmacia, que en estas instancias se le agrega dos atributos importantes a los mismos; la categoría de área a la cual pertenecen y la fecha de vencimiento del mismo. Por lo tanto se detalla a continuación los atributos que contendrá cada producto:

* id: Numérico :: int.
* fechaCreacion: Fecha con formato YYYY-MM-DD hh:mm:ss :: Date
* ultimaModificacion: Fecha con formato YYYY-MM-DD hh:mm:ss :: Date
* precio: Numérico :: Double
* descripcionCorta: Cadena de caracteres :: String
* descripcionLarga: Cadena de caracteres :: String
* areaAplicacion: Cadena de caracteres :: String
* estado: Cadena de caracteres :: String
  + Posibles valores:
    - Activo
    - Inactivo
    - Null
* fechaVencimiento: Fecha con formato YYYY-MM-DD hh:mm:ss :: Date
* refrigerado: Valor booleano (True, False) :: Boolean
* requiereReceta: Valor booleano (True, False) :: Boolean

**Formato para carga de productos mediante archivo .csv:**

Ejemplo de línea de archivo con formato adecuado para el ingreso de nuevos productos.

10004;2017-03-30 17:50:00:13;2017-04-10 10:32:05;150;BioGrip Plus; BioGrip Plus Caja 10 Sobres;Medicina General; 2019-03-30 17:50:00;Activo;false;false

**Formato para Alta/Modificación de stock mediante archivo .csv:**

Ejemplo de línea de archivo con formato adecuado para la modificación de stock.

10004;300

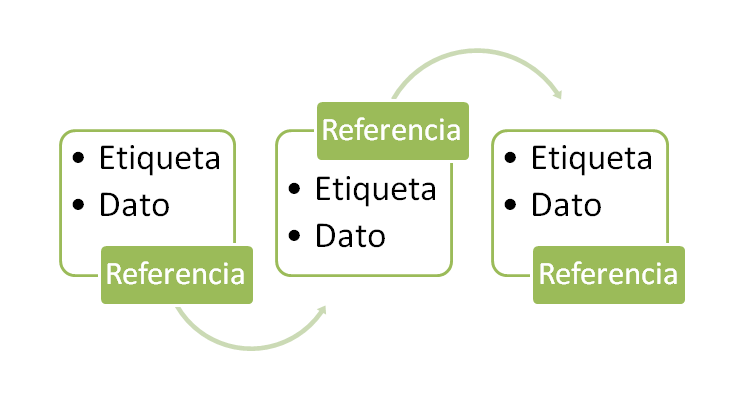
Donde el primer valor es el código del producto y el segundo la cantidad de stock que hay del mismo.

# Análisis de alternativas

En esta sección se presentan las alternativas a implementar en cuanto a estructuras de datos se refiere, para el almacenamiento de la información correspondiente. En esta etapa del proyecto se tendrán en cuenta dos alternativas a utilizar, las Listas Enlazadas Simple, tratadas en el informe anterior, y un nuevo tipo de estructura de datos; Árboles Binarios. Se introducirán las mismas y se discutirán varias combinaciones posibles de ellas para el desarrollo de este programa.

Listas enlazadas simples:

Las listas enlazadas simples son un tipo de dato abstracto, en el cual la información que se almacena no es continua, sino que está referenciado uno a uno. En estas listas se guardan objetos llamados Nodos, en ellos se encuentra la información que se desea guardar, y un atributo que hace referencia al siguiente Nodo de la lista.



Este tipo de datos es muy utilizado cuando no se sabe exactamente cuánta información se va a almacenar, ya que se puede redimensionar fácilmente en cualquier momento, agregando un Nodo nuevo donde se desee.

La desventaja de la implementación de las listas con enlace simple, es que cuando se quiere buscar un dato dentro de ella, hay que recorrerla desde el inicio Nodo por Nodo hasta llegar al que se está buscando, debido a que los Nodos no están indexados y las referencias son simples, se puede ir únicamente hacia el nodo siguiente, y no hacia el anterior.

Árboles binarios:

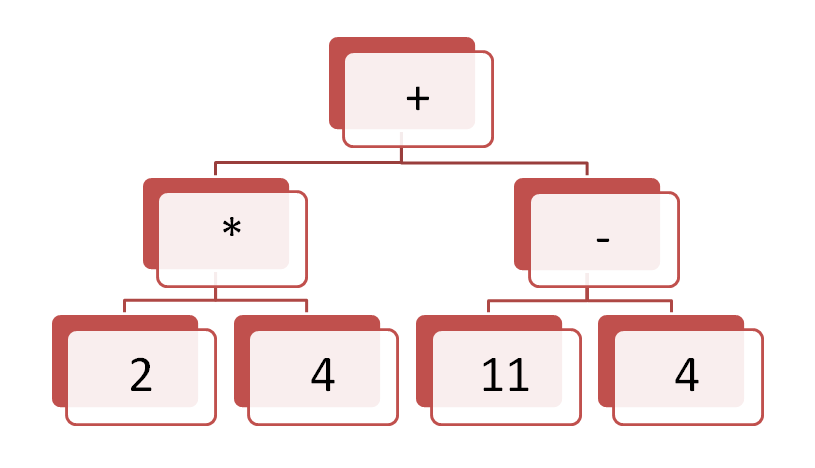
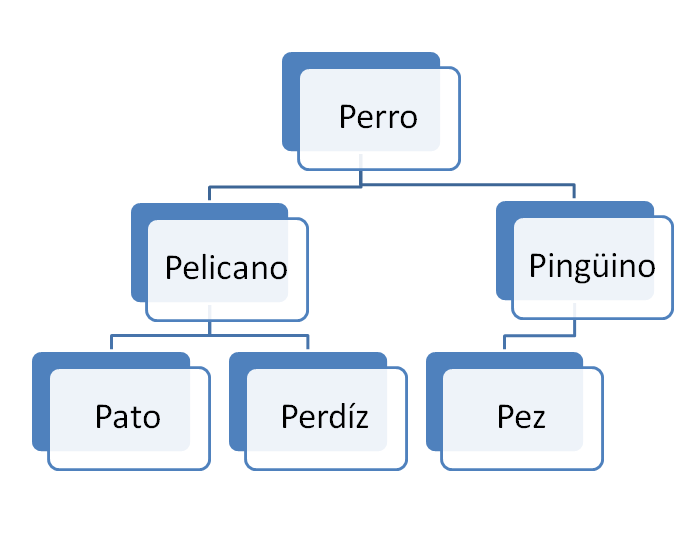
Un árbol es una manera de agrupar nodos en una jerarquía *raíz/padre/hijo*, donde únicamente uno de los nodos sirve como raíz, y el resto se organiza debajo de la misma. Un nodo que tiene una referencia debajo de él se le denomina *nodo padre*. Y ascendentemente, un nodo que es referenciado por uno que se encuentra encima, es un *nodo hijo*.

Particularmente, los árboles binarios son aquellos que se forman por agrupaciones pares de hijos para cada padre, por lo tanto, un *nodo padre* puede tener únicamente dos hijos, a los cuales se les denomina; *hijo izquierdo* e *hijo derecho*.

Los nodos que no tienen hijos, son denominados *nodos hoja.* Un nodo puede ocupar el puesto de un *nodo padre* y un *nodo hijo*, o ser un *nodo hijo* y un *nodo hoja*, con la salvedad de que si se trata de un árbol que tiene únicamente raíz, la misma será *raíz* y *hoja* a la vez.

Los árboles binarios son muy utilizados en temas de búsqueda, procesos de decisión y expresiones aritméticas.

A continuación se pueden ver ejemplos de árboles implementados con ese fin.



En los árboles binarios se pueden implementar tres métodos distintos para el recorrido de sus elementos, a estos se les denomina: *preorden*, *inorden* y *postorden*. En el recorrido en *preorden* un nodo es visitado antes que sus descendientes. En el *inorden*, el nodo es visitado después que su subárbol izquierdo y antes que su subárbol derecho. Finalmente, en *postorden* el nodo es visitado después que sus descendientes.

Los árboles binarios, o n-arios, son una alternativa interesante en cuanto a búsquedas de información refiere, ya que, correctamente implementado, se puede alcanzar órdenes de ejecución logarítmicos, agilitando mucho la obtención de información, que, en el peor de los casos mantiene el orden n convencional.

### Alternativas de combinación:

Las dos estructuras de datos presentadas se pueden implementar combinadas de distintas maneras en el programa, y dependiendo de estas alternativas, cuál será su consumo de memoria y eficiencia de ejecución. A continuación se presentan varias de las posibles.

### Alternativa I:

Una de las formas en las cuales se puede organizar la información de UCUPharma, es mediante la utilización de un árbol binario para contener los productos de la farmacia, y utilizar listas enlazadas para contener las ventas y las compras. Para contener las áreas de aplicación y luego poder listarlas fácilmente, se puede alojar a las mismas en una lista de listas enlazadas, donde cada primer lista es un área, y contiene listas internas con referencias a los productos.

Esta forma permite que los productos sean fácilmente insertados dentro del árbol binario, y que su búsqueda y su eliminación sean igual de sencillas. Los documentos de compras y ventas pueden estar ubicados en listas enlazadas, ya que como para su búsqueda se utilizan rangos fechas, el orden de tiempo de ejecución no varía entre uno y otro. Las áreas de aplicación también son fácilmente manejables desde listas, pudiendo permitir su eliminación y la de sus productos anidados.

Esta alternativa permite una flexibilidad importante en el tratamiento de la información, y mantiene bajos costos de memoria y tiempos de orden de ejecución.

### Alternativa II:

Como segunda alternativa se puede implementar los productos en un árbol binario, los documentos de venta y compras en un único árbol binario distribuido por fecha, y las áreas de aplicación en listas con listas enlazadas dentro.

Con esta implementación, los productos son fácilmente modificables, al igual que en la alternativa anterior. La estructura de los documentos de venta y compra es más compleja, pero al estar ordenados por fecha y encontrarse en un mismo árbol, las funciones como “Reporte de movimientos de stock por área”, se podrán realizar de manera mucho más eficiente al tener los movimientos de stock en un único lugar. Las áreas de aplicación se siguen manejando con listas de listas y su redimensionamiento es muy ágil, por lo cual a futuro se pueden crear o eliminar áreas con mucha facilidad.

### Alternativa III:

Finalmente se plantea el almacenamiento de productos en listas simplemente enlazadas, compras y ventas en árboles binarios distintos y áreas de aplicación en listas que contienen listas.

Con esta última alternativa se vuelve más lento el proceso de búsqueda, inserción y eliminación de productos y por lo tanto más ineficiente, las demás estructuras de información se mantienen inmutables, por lo que conservan su agilidad y practicidad a la hora de hacer modificaciones.

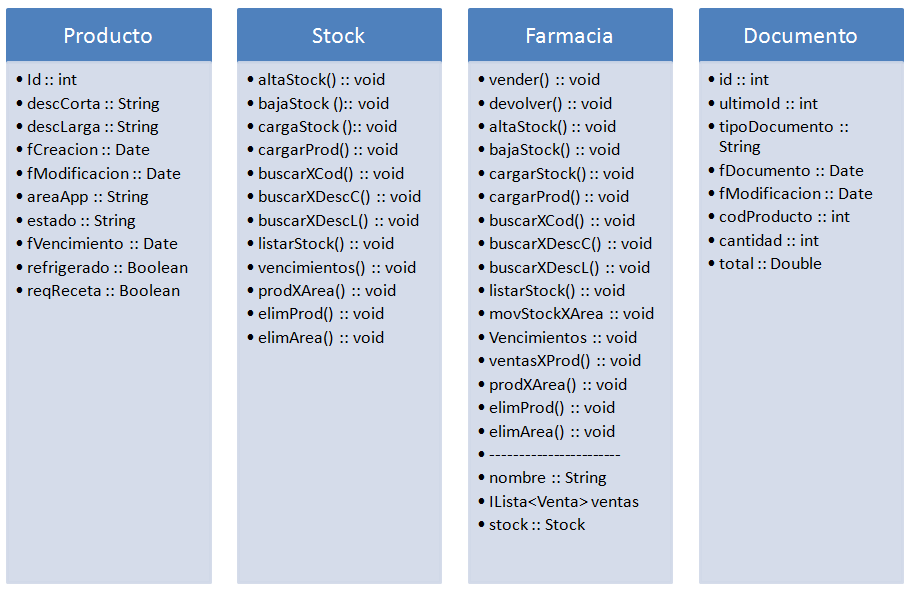
### Algoritmos

* Alternativa I:
  + Productos: O(log) 🡪 En el mejor de los casos.
  + Compras y ventas: O(n).
  + Áreas de aplicación:O(m\*n) 🡪 Siendo m la cantidad de áreas, y n la cantidad de productos contenidos en ellas.
* Alternativa II:
  + Productos: O(log) 🡪 En el mejor de los casos.
  + Compras y ventas: O(n).
  + Áreas de aplicación:O(m\*n) 🡪 Siendo m la cantidad de áreas, y n la cantidad de productos contenidos en ellas.
* Alternativa III:
  + Productos: O(n).
  + Compras y ventas: O(n).
  + Áreas de aplicación:O(m\*n) 🡪 Siendo m la cantidad de áreas, y n la cantidad de productos contenidos en ellas.

# Selección y justificación de alternativa a implementar

La alternativa que mejor se adapta a las necesidades de este proyecto, es la segunda, ya que su combinación de listas y árboles de esa manera en particular la hace una de las más eficientes en términos de consumo de memoria y órdenes de ejecución. Utilizando esta alternativa se puede fácilmente agregar y modificar productos, poder listar compras y ventas desde una misma búsqueda, y tener una relación entre los productos y las áreas de aplicación.

A continuación, se muestra un diagrama de clases a utilizar en la implementación. En la misma, se utilizan clases e interfaces extras, utilitarias, ILista, Lista, INodo, Nodo, ManejadorArchivos, ManejadorFechas, IProducto, IDocumento.



# Conclusiones

Luego de analizada la especificación del cliente sobre el sistema requerido, y tras haber hecho un análisis de las distintas combinaciones de alternativas para la implementación del mismo, teniendo en cuenta el orden de procesamiento y el consumo de memoria de cada uno; se considera a ésta, una propuesta de solución integral, que no solo cumple con los requerimientos solicitados, sino que también tiene en cuenta la usabilidad y extensibilidad del programa a futuros locales de la línea.

Se culmina el presente documento, con gratitud hacia el lector por dedicar su tiempo, indicando una breve instrucción del uso del sistema mediante consola, basándose en el programa inicial, hasta el momento el desarrollado, y añadiendo la forma en la que se verán las futuras características del mismo.

# Guía del usuario

En esta sección se explica brevemente la forma de uso del sistema, a modo introductorio para el usuario.

Cuando se inicia el programa se mostrará un menú que indica las diferentes tareas que se pueden realizar:

1. Cargar/modificar productos desde un archivo.
2. Ingresar un producto nuevo de forma manual.
3. Agregar stock a un producto desde un archivo.
4. Agregar stock a un producto manualmente.
5. Bajar stock a un producto manualmente.
6. Realizar una venta.
7. Realizar una devolución de stock.
8. Buscar un producto por su código.
9. Buscar un producto por su descripción corta.
10. Buscar un producto por su descripción larga.
11. Listar los productos por ordenados por nombre.
12. Mostrar las ventas realizadas entre un rango de fechas dado.
13. Listar movimientos de stock de un producto.
14. Listar por área de aplicación.
15. Listar movimientos de stock por área de aplicación.
16. Listar vencimientos próximos al año.
17. Promedio de ventas mensuales de un producto.
18. Eliminar producto.
19. Eliminar área de aplicación.
20. Salir.

Para acceder a ellas, se ingresa por consola el número de la opción deseada, y se siguen las instrucciones que se muestran a continuación.