Programación

Práctica 3 Estructuras de Datos Lineales (EDL)

Contenido

Contextualización

En la práctica 1 de la asignatura se desarrolló un software de gestión de una tienda de ropa ficticia. Este software era utilizado por un empleado de la tienda, para realizar diferentes gestiones sobre el inventario de prendas y realizar un registro de ventas.

Recordando la práctica para la gestión del inventario nos basábamos en una clase ClothingItem que tenía como variables name, price and size y la clase Inventory donde almacenábamos las prendas como array (arreglo).

Emplear arrays presenta ciertas limitaciones como que debemos fijar el tamaño del array en su creación. Esto provoca que se reserve un tamaño fijo de memoria contigua, lo que lleva a desperdiciar la memoria si se reserva más espacio del necesario o la imposibilidad de almacenar más elementos si el tamaño máximo es alcanzado.

Por otra parte, cualquier cambio en un array, como insertar o eliminar un elemento en una posición intermedia obliga a mover todos los elementos restantes para hacer hueco al nuevo elemento o llenar el vacío de un elemento eliminado.

En esta práctica vamos a trabajar estructuras lineales empleando una lista enlazada de elementos para la implementación de un carro de la compra. A diferencia de los arrays las listas enlazadas nos aportan flexibilidad ya que los elementos pueden insertarse o eliminarse fácilmente en cualquier posición sin tener que mover otros elementos en memoria, lo que puede ser más costoso en términos de tiempo para los arrays.

Otro aspecto a destacar es que en listas enlazadas solo se utiliza la memoria necesaria para almacenar los elementos y los punteros que los conectan, lo que significa que pueden crecer y reducirse dinámicamente según sea necesario.

En cuanto a eficiencia en una lista enlazada, la inserción y eliminación de elementos, por ejemplo, al principio del almacén se realiza en tiempo constante independientemente de la talla de la lista.

Tareas a realizar

En esta práctica se realizará la implementación necesaria para que el código de la práctica 1 quede adaptado de forma que incluya las opciones de implementar un carrito de la compra. Para el carrito de la compra se empleará una lista enlazada.

La aplicación a desarrollar permitirá a un usuario que elija un producto del inventario y pueda introducir dicho producto y las unidades de éste en el carro de compra. El alumno deberá tener en cuenta el control del stock disponible para los productos que se introducen/eliminan del carrito de compra.

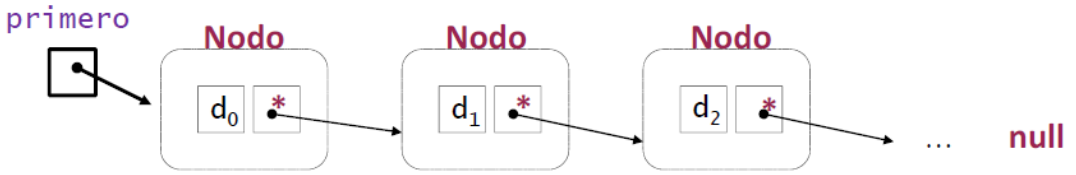
Las funcionalidades que adicionalmente proporcionará la aplicación serán:

- Mostrar los productos del carrito
- Añadir unidades de producto al carrito (y actualización de inventario)
- Eliminar unidades del carrito (y actualización de inventario)
- Confirmación de la compra

Preparación paquetes

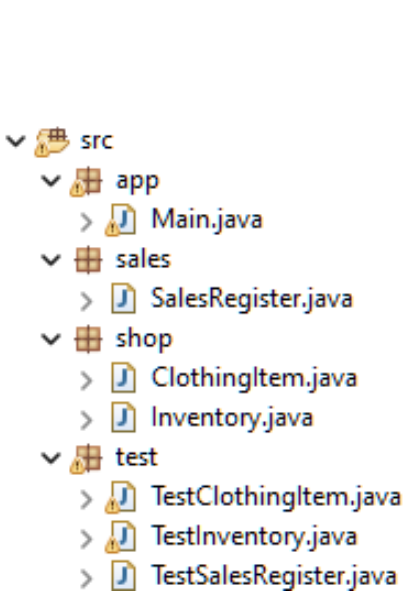
Vamos a crear todo lo necesario para gestionar el carrito de compra de las prendas del inventario.

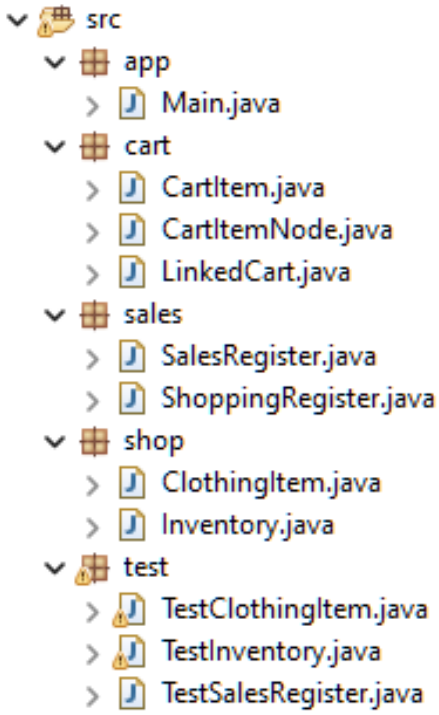
La implementación de la lista de productos del carro la haremos mediante una lista enlazada.



Representación enlazada de secuencias

Primero crearemos un nuevo paquete **cart** en el proyecto, de forma que ahora lo que tengamos en él sean las clases que necesitaremos para cada uno de los elementos de la lista.





Estructura de paquetes del proyecto

Tarea 1 - CartItem

Esta clase representa un elemento contenido en el carro de compra. Cada elemento del carro de compra representará:

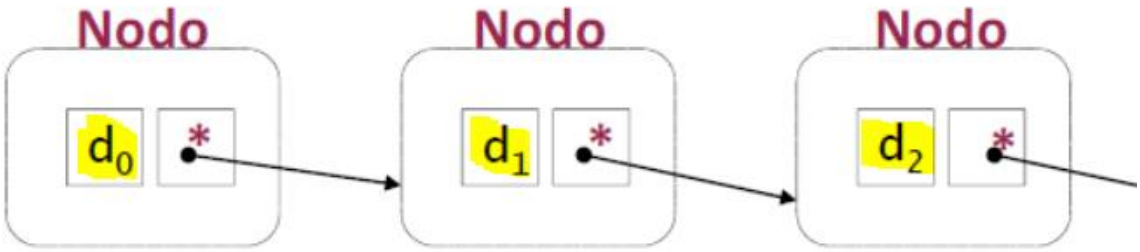
- Un **clothingItem** del inventario y
- las **unidades** que se tienen de él en el carro.

A continuación, se presentan los pasos a realizar.

1. Crea una clase `CartItem` en el paquete `cart`
2. La clase contendrá las siguientes variables de instancia:
    - `clothingItem` de tipo `ClothingItem`
    - `units` de tipo `int`
3. Declara constructores para instanciar objetos de la clase.
4. Aplica el principio de ocultación para evitar el acceso a las variables de instancia de manera directa, y declara *getters & setters* necesarios.
5. Sobrescribe el método `toString`, que devolverá la representación en `String` de `cartItem`.

Tarea 2 - CartItemNode

A continuación, se debe implementar una clase denominada **CartItemNode** en la que se representa la estructura que asocia un elemento del carro de compra **dato** (`CartItem`), y una referencia al siguiente nodo en una estructura de lista enlazada



d0, d1, d....dn son objetos CartItem

Crea esta clase en el paquete `cart` e implementa los dos constructores necesarios para la implementación de la lista enlazada.

Además, debes aplicar el principio de ocultación para evitar el acceso a las variables de instancia de manera directa, y declara getters & setters necesarios.

Tarea 3 - LinkedCart

El carro de compra lo representaremos mediante una lista enlazada de una secuencia de elementos `CartItem`. Crea esta clase en el paquete `cart` y añade en ella las variables de instancia necesarias:

- `first`, referencia al primer nodo de la lista enlazada.
- `size`, tipo `int`

Además, implementaremos las operaciones de la estructura de datos, de tipo lista enlazada.

1. **Constructor** de la lista que inicializa una lista enlazada vacía.
2. Métodos de consulta:
    - Comprobar si la lista **es vacía**.
    - Obtener el **número de elementos** de la lista.
    - **Buscar** un elemento en la lista facilitando los parámetros de itemName y size, devolviendo la posición del elemento o -1 si no lo encuentra
    - **Buscar** (sobrecarga) devuelve el `CartItem` que está en una **posición** dada.
3. Métodos modificadores:
    - **Insertar** un elemento en una **posición** dada i (0<= i <=n)
    - **Eliminar** un `CartItem` que se encuentra en la **posición** i (0<= i <=n-1), devolviendo el elemento de dicha posición.

Se adjuntan a continuación las cabeceras de los métodos de la clase a implementar:

```
package cart;

public class LinkedCart {

    /* Instance variables */

    public LinkedCart() {...}

    public boolean isEmpty() {...}

    public int find(String itemName, char size) {...}

    public CartItem find(int index) {...}

    public int size() {...}

    public void insert(int index, CartItem item) {...}

    public CartItem remove(int index) {...}
}
```

Tarea 4 - ShoppingRegister

Añade esta clase en el paquete ya creado anteriormente sales. En ella se proporcionan los métodos necesarios para la gestión de la tienda a partir de la interacción con el usuario que se realiza desde la clase Main.

Esta clase tendrá únicamente variables y métodos de instancia, y deberá ser instanciada desde la clase Main.

Variable de instancia

- cart de tipo LinkedCart

Respecto al constructor para la clase ShoppingRegister debe inicializar el carrito como una nueva instancia de LinkedCart en su estado inicial.

En cuanto a los métodos de instancia se indican a continuación su funcionalidad.

Añadir artículos al carrito

A partir del inventario, la referencia del producto y la cantidad de unidades a añadir.

1. Implementa un método público llamado addToCart que acepte los siguientes parámetros:
    - inventory de tipo Inventory: El inventario de la tienda.
    - itemName de tipo String: El nombre del artículo a agregar.
    - size de tipo char: El tamaño del artículo a agregar.
    - quantity de tipo int: La cantidad de unidades del artículo a agregar.
2. Dentro del método addToCart, verifica si hay suficiente stock disponible en el inventario para agregar la cantidad solicitada del artículo.
3. Si hay suficiente stock disponible, iterativamente extrae el artículo del inventario, crea un CartItem con el artículo y la cantidad especificada, e inserta el CartItem en el carrito.
4. Si el artículo ya está presente en el carrito, incrementa la cantidad de unidades del artículo en el carrito en lugar de insertar un nuevo CartItem.
5. Si el artículo no se encuentra en el inventario, muestra un mensaje indicando que el producto no fue encontrado.

Eliminar artículos del carrito

A partir del inventario, la referencia del producto y la cantidad de unidades a eliminar.

1. Implementa un método público llamado removeFromCart que acepte los siguientes parámetros:
    - inventory de tipo Inventory: El inventario de la tienda.
    - itemName de tipo String: El nombre del artículo a eliminar.
    - size de tipo char: El tamaño del artículo a eliminar.
    - quantity de tipo int: La cantidad de unidades del artículo a eliminar.
2. Dentro del método removeFromCart, busca el artículo en el carrito por nombre y tamaño.
3. Si el artículo se encuentra en el carrito y la cantidad de unidades en el carrito es suficiente, reduce la cantidad de unidades del artículo en el carrito y devuelve las unidades eliminadas al inventario.
4. Si la cantidad restante del artículo en el carrito es cero, elimina el artículo del carrito.

Confirmar carrito de compra y procesar venta

1. Implementa un método público llamado confirmCart que acepte el parámetro inventory de tipo Inventory.
2. Dentro del método confirmCart, itera sobre los artículos en el carrito.
3. Para cada artículo en el carrito, muestra los detalles del artículo y calcula el costo total del pedido.
4. Devuelve cada artículo al inventario y registra la venta utilizando el método SalesRegister.processSale.
5. Vacía el carrito después de confirmar la compra.

Mostrar productos del carrito

1. Implementa un método llamado showCart que no acepte parámetros.
2. Dentro del método showCart, verifica si el carrito está vacío.
3. Si el carrito está vacío, muestra un mensaje indicando que el carrito está vacío.
4. Si el carrito no está vacío, muestra los detalles de cada artículo en el carrito, incluyendo el nombre, tamaño, precio y cantidad de unidades.

Se adjuntan las cabeceras de los métodos de esta clase:

```
package sales;

/* imports */

public class ShoppingRegister {

   /* instance variables */
   public ShoppingRegister(){...}

   public void addToCart(Inventory inventory, String itemName, char
size, int quantity) {...}

   public void removeFromCart(Inventory inventory, String itemName,
char size, int quantity){...}

   public void confirmCart(Inventory inventory){...}

   public void showCart(){...}
 }
```

Tarea 5 - Actualizar Main

El último paso de la práctica se debe acabar de implementar las opciones de interacción con el usuario.

Se **creará una variable de clase** además del Inventory y Scanner (que ya teníamos), un ShoppingRegister para ejecutar las funcionalidades requeridas, tras la selección de opción del usuario e introducción de la información necesaria.

Es aconsejable realizar una llamada a nextLine() del Scanner cada vez que se utiliza para obtener un valor numérico, dado que el \n se queda en el buffer y la siguiente llamada a nextXXX() omitirá la lectura por teclado.

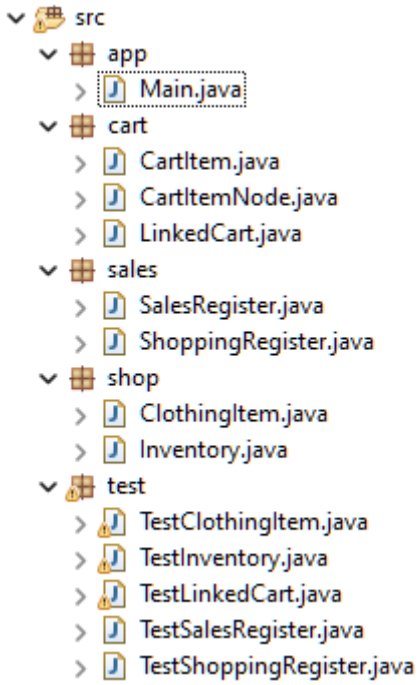
```
*** Bienvenido a Strafalarius ***
Seleccione opción:
1. Agregar nueva prenda al inventario
2. Mostrar inventario
3. Procesar venta
4. Mostrar estadísticas de ventas
5. Agregar prenda al carrito
6. Eliminar prenda del carrito
7. Confirmar carrito de compra
8. Mostrar productos del carrito
9. Salir
Seleccione una opción (1-9):
```

Autotest / Autoevaluación

Junto con el material de las prácticas se encuentran dos ficheros uno para comprobar las últimas tareas de creación de clases:

- Tarea 3 – LinkedCart ClothingItem
- Tarea 4 – ShoppingRegister

El alumno/a deberá incluir en el paquete test los recursos facilitados.



En la clase main añade el siguiente código para realizar el test.

```
TestLinkedCart.checkLinkedCart();
TestShoppingRegister.checkShoppingRegister();
```

Verifica por consola que no da ningún error, y corrige si los hubiera.