**MÓDULO OBJETOS.** *Cree un proyecto con su Apellido y resuelva en Java. Entrega: comprimir el desarrollo en .zip (no debe incluir ningún archivo .jar) y enviarlo por correo.*

**1)** Queremos representar libros electrónicos e impresos. De cualquier libro se conoce: título, precio base y el nombre de los autores (a lo sumo 8). Adicionalmente: los libros electrónicos tienen el formato (ej: “.pdf”, “.epub”) y su tamaño en MB; mientras que los libros impresos registran si es tapa dura o no.

**a)** Implemente las clases del modelo con sus atributos y getters/setters adecuados. Provea constructores para iniciar los libros a partir de toda su información y sin autores.

**b)** Agregue a las clases implementadas los métodos necesarios para incorporar la siguiente funcionalidad:

**i-** Agregar al libro un autor cuyo nombre se recibe. Asuma que hay espacio.*.*

**ii-** Obtener el precio final del libro. Para ambos libros el precio final surge de adicionar al precio base el IVA (21% del precio base). Además, para los libros electrónicos se adiciona un impuesto de descarga de 2,5$ por MB.

**iii-**  Obtener la representación String del libro, la cual se compone de título, precio final y el nombre de su primer autor.

**2)** Realice un programa que instancie un libro electrónico y un libro impreso. Cargue 3 autores a cada uno. Para finalizar, muestre la representación String de los libros.

**MÓDULO OBJETOS.** *Cree un proyecto con su Apellido y resuelva en Java. Entrega: comprimir el desarrollo en .zip (no debe incluir ningún archivo .jar) y enviarlo por correo.*

**1)** Queremos representar libros electrónicos e impresos. De cualquier libro se conoce: título, precio base y el nombre de los autores (a lo sumo 8). Adicionalmente: los libros electrónicos tienen el formato (ej: “.pdf”, “.epub”) y su tamaño en MB; mientras que los libros impresos registran si es tapa dura o no.

**a)** Implemente las clases del modelo con sus atributos y getters/setters adecuados. Provea constructores para iniciar los libros a partir de toda su información y sin autores.

**b)** Agregue a las clases implementadas los métodos necesarios para incorporar la siguiente funcionalidad:

**i-** Agregar al libro un autor cuyo nombre se recibe. Asuma que hay espacio.

**ii-** Obtener el precio final del libro teniendo en cuenta que:

- Para los libros electrónicos es el precio base al que se adiciona un impuesto de descarga de 2,5$ por MB.

- Para los libros impresos es el precio base al que se adiciona 500$ si es de tapa dura.

**iii-**  Obtener la representación String del libro, la cual se compone de título, precio final y nombre de sus autores.

**2)** Realice un programa que instancie un libro electrónico y un libro impreso. Cargue 3 autores a cada uno. Para finalizar, muestre la representación String de los libros.

**MÓDULO OBJETOS.** *Cree un proyecto con su Apellido y resuelva en Java. Entrega: comprimir el desarrollo en .zip (no debe incluir ningún archivo .jar) y enviarlo por correo.*

**1)** Queremos representar estanterías de libros. Una estantería mantiene sus libros organizados en N estantes cada uno con lugar para M libros. Un libro posee título, nombre de su primer autor y peso.

**a)** Implemente las clases de su modelo, con sus atributos y getters/setters adecuados. Provea constructores para iniciar: los libros a partir de toda su información; la estantería para N estantes y lugar para M libros por estante (inicialmente no debe tener libros cargados).

**b)** Implemente los siguientes métodos:

- *almacenarLibro*: recibe un libro y lo almacena en el primer estante que tenga lugar. Asuma que hay espacio para almacenar el libro.

*- sacarLibro*: saca y devuelve el libro que se encuentra en el estante X, lugar Y (X e Y se reciben y son válidos). Dicho lugar debe quedar disponible.

- *calcular*:calcula y devuelve la cantidad de libros de un autor cuyo nombre se recibe.

**2)** Realice un programa que instancie una estantería para 5 estantes y 3 libros por estante. Almacene 7 libros en la estantería. A partir de la estantería: saque un libro e informe su representación String; luego,  informe la cantidad de libros de “Borges”.

**MÓDULO OBJETOS.** *Cree un proyecto con su Apellido y resuelva en Java. Entrega: comprimir el desarrollo en .zip (no debe incluir ningún archivo .jar) y enviarlo por correo.*

**1)** Queremos representar estanterías de libros. Una estantería mantiene sus libros organizados en N estantes cada uno con lugar para M libros. Un libro posee título, nombre de su primer autor y peso.

**a)** Implemente las clases de su modelo, con sus atributos y getters/setters adecuados. Provea constructores para iniciar: los libros a partir de toda su información; la estantería para N estantes y lugar para M libros por estante (inicialmente no debe tener libros cargados).

**b)** Implemente los siguientes métodos:

- *almacenarLibro*: recibe un libro y un nro. de estante válido, y lo almacena en el primer lugar libre de dicho estante. Asuma que hay espacio para almacenar el libro.

*- sacarLibro*: saca y devuelve el libro que se encuentra en el estante X, lugar Y (X e Y se reciben y son válidos). Dicho lugar debe quedar disponible.

- *calcular*:calcula y devuelve el número del estante más pesado (teniendo en cuenta el peso de sus libros).

**2)** Realice un programa que instancie una estantería para 5 estantes y 3 libros por estante. Almacene 7 libros en la estantería. A partir de la estantería: saque un libro e informe su representación String; luego,  informe el número de estante más pesado.

**MÓDULO OBJETOS.** *Cree un proyecto con su Apellido y resuelva en Java. Entrega: comprimir el desarrollo en .zip (no debe incluir ningún archivo .jar) y enviarlo por correo.*

**1)** Queremos representar libros electrónicos e impresos. De cualquier libro se conoce: título, precio base y el nombre de los autores (a lo sumo 8). Adicionalmente: los libros electrónicos tienen el formato (ej: “.pdf”, “.epub”) y su tamaño en MB; mientras que los libros impresos registran si es tapa dura o no.

**a)** Implemente las clases del modelo con sus atributos y getters/setters adecuados. Provea constructores para iniciar los libros a partir de toda su información y sin autores.

**b)** Agregue a las clases implementadas los métodos necesarios para incorporar la siguiente funcionalidad:

**i-** Agregar al libro un autor cuyo nombre se recibe. Asuma que hay espacio.*.*

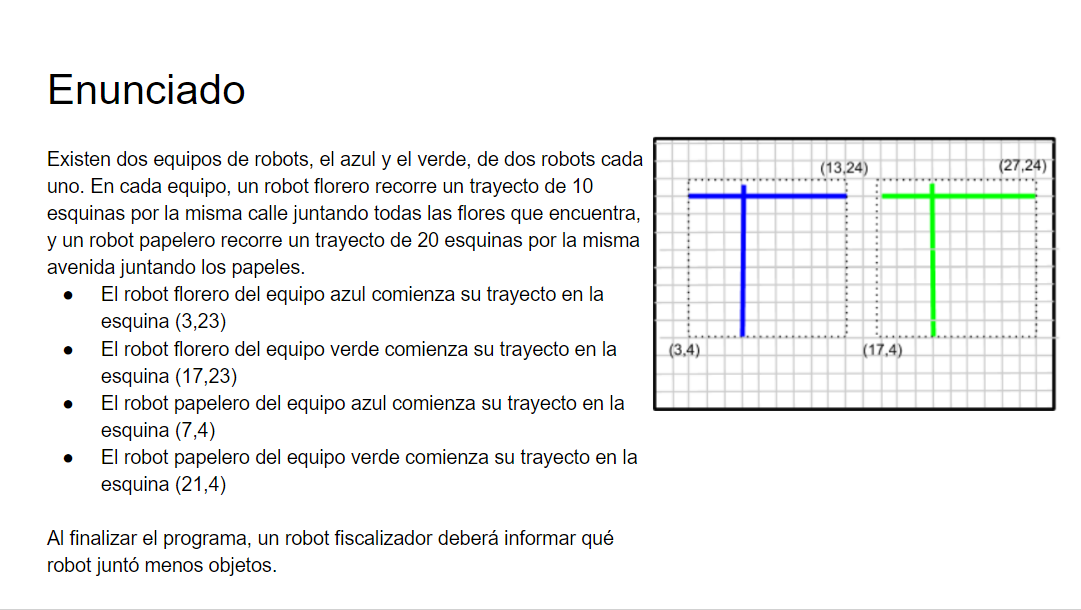
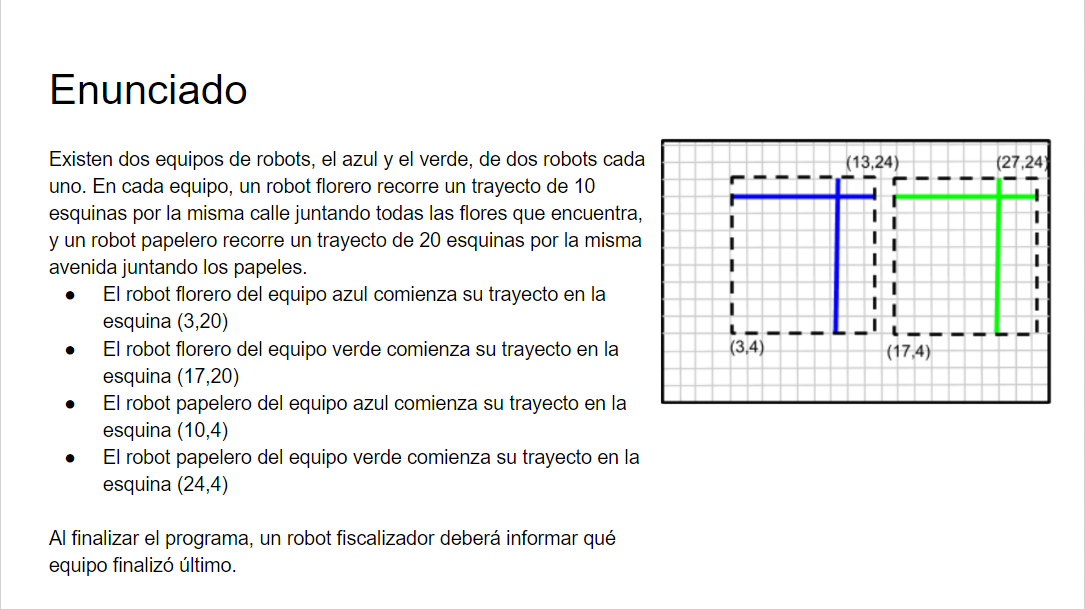
**ii-** Obtener el precio final del libro. El precio final es el precio base al que se adiciona el IVA (21% del precio base).

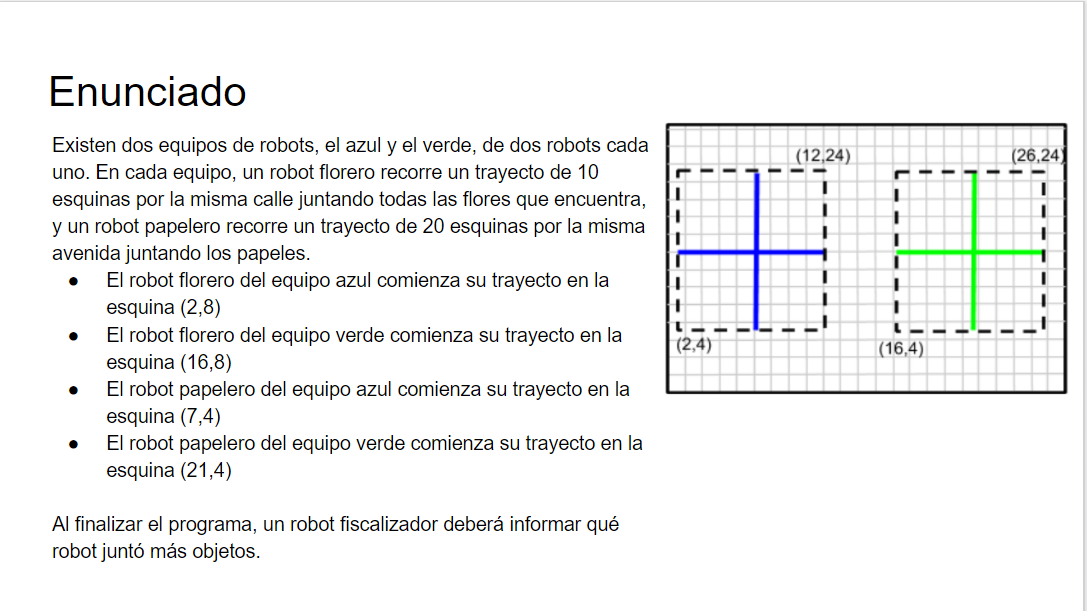
**iii-** Obtener la representación String del libro siguiendo el formato de ejemplo:

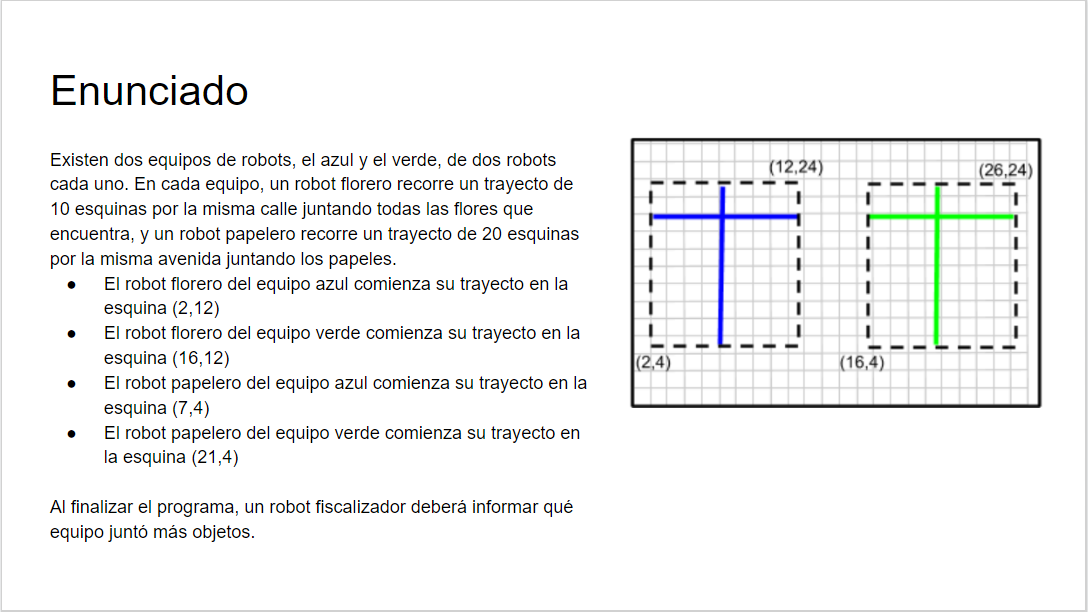
Libro electrónico: “*Título, Nombre de los autores, precio final, formato, tamaño en MB”*

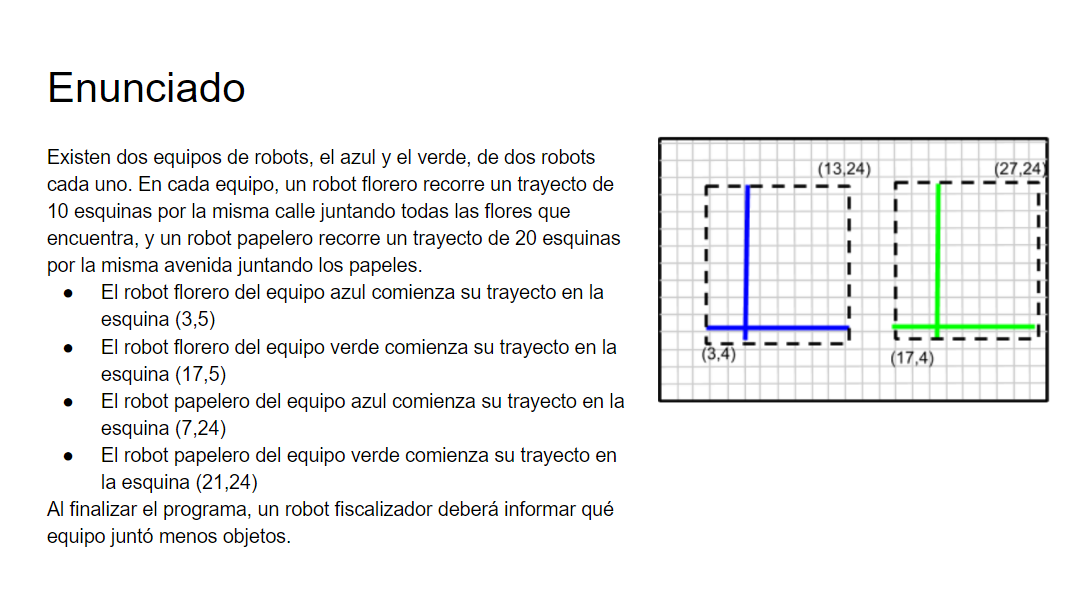
Libro impreso: *“Titulo, Nombre de los autores, precio final, tapa dura”*

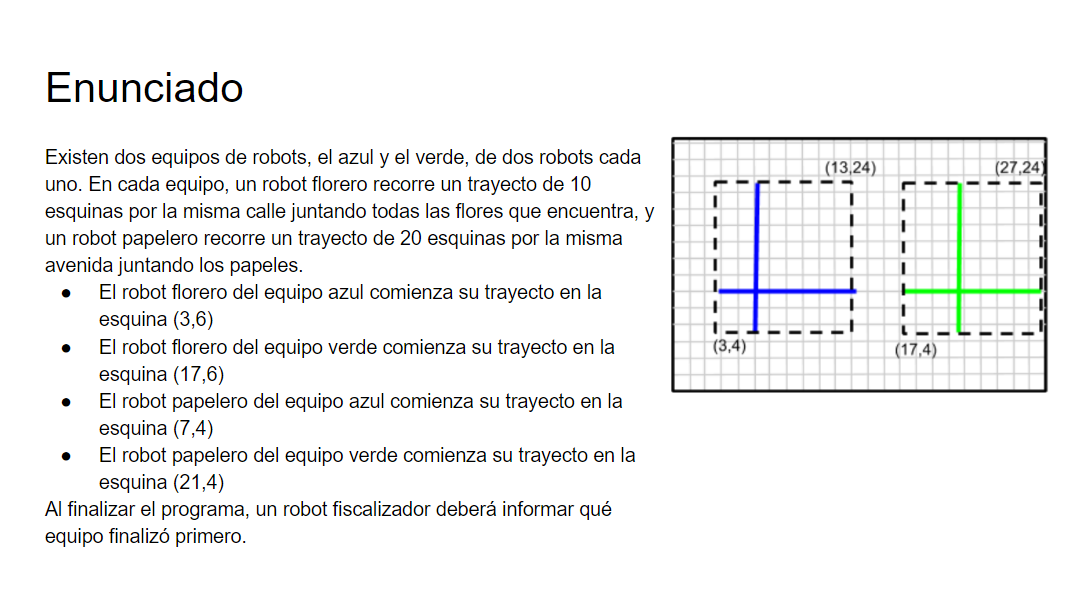
**2)** Realice un programa que instancie un libro electrónico y un libro impreso. Cargue 3 autores a cada uno. Para finalizar, muestre la representación String de los libros.











Imperativo

Se lee información acerca de las ventas de productos realizadas en las 5 sucursales de una empresa.

Cada sucursal realizó a lo sumo 200 ventas. De cada venta se conoce el código de producto, cantidad

vendida y monto total de la venta. Las ventas de cada sucursal se leen de manera consecutiva y

ordenadas por código de producto. La lectura por cada sucursal finaliza al completar las 200 ventas o

cuando se lee el código de producto -1, el cual no se procesa. Implementar un programa para que a

partir de la información leida, resuelva los siguientes ítems:

a) Utilizando la técnica de merge o merge acumulador según corresponda, generar una lista que

contenga la cantidad total vendida para cada código de producto, ordenada por código de

producto.

b) Realizar un módulo recursivo que reciba la lista generada en el punto a y retorne la cantidad de

productos para los cuales la cantidad total vendida supera las 500 unidades.

Imperativo

Se lee información acerca de las ventas de productos realizadas en las sucursales de una empresa. De

cada venta se conoce fecha, código de producto, cantidad vendida y monto total de la venta. La lectura

finaliza cuando se lee el código de producto -1, el cual no se procesa. Implementar un programa para

que a partir de la información leida, resuelva los siguientes ítems:

a) Generar un árbol binario de búsqueda ordenado por código de producto, donde cada nodo

contenga el código del producto y la cantidad total vendida. El código de producto no puede

repetirse en el árbol.

b) Realizar un módulo que reciba el árbol generado en a. y dos códigos de producto y retorne la suma

de todas las cantidades vendidas para los códigos de productos comprendidos en el rango.

Nota: La información se lee en forma desordenada. Puede existir mas de una venta para un mismo

código de producto.

Imperativo

Se lee información acerca de las ventas de productos realizadas en las sucursales de una empresa. De

cada venta se conoce fecha, código de producto, cantidad vendida y monto total de la venta. La lectura

finaliza cuando se lee el código de producto -1, el cual no se procesa. Implementar un programa para

que a partir de la información leida, resuelva los siguientes ítems:

a) Generar un árbol binario de búsqueda ordenado por código de producto, donde cada nodo

contenga el código del producto y el monto total vendido. El código de producto no puede repetirse

en el árbol.

b) Realizar un módulo que reciba el árbol generado en a. y un código de producto y retorne la suma

de todos los montos vendidos para los códigos de productos mayores al código recibido.

Nota: La información se lee en forma desordenada. Puede existir más de una venta para un mismo

código de producto.

Imperativo

Se lee información acerca de las ventas de productos realizadas en las sucursales de una empresa. De

cada venta se conoce fecha, código de producto, cantidad vendida y monto total de la venta. La lectura

finaliza cuando se lee el código de producto -1, el cual no se procesa. Implementar un programa para

que a partir de la información leída, resuelva los siguientes ítems:

a) Generar un árbol binario de búsqueda ordenado por código de producto, donde cada nodo

contenga el código del producto y la cantidad total vendida. El código de producto no puede

repetirse en el árbol.

b) Realizar un módulo que reciba el árbol generado en a. y una cantidad, y retorne la cantidad de

códigos de producto cuya cantidad total vendida superan la cantidad recibida.

Nota: La información se lee en forma desordenada. Puede existir mas de una venta para un mismo

código de producto.

Imperativo

Se lee información acerca de las ventas de productos realizadas en las 5 sucursales de una empresa.

Cada sucursal realizó a lo sumo 250 ventas. De cada venta se conoce el código de producto, cantidad

vendida y monto total de la venta. Las ventas de cada sucursal se leen de manera consecutiva y

ordenadas por código de producto. La lectura por cada sucursal finaliza al completar las 250 ventas o

cuando se lee el código de producto -1, el cual no se procesa. Implementar un programa para que a

partir de la información leida, resuelva los siguientes ítems:

a) Utilizando la técnica de merge o merge acumulador según corresponda, generar una lista que

contenga la cantidad total vendida para cada código de producto, ordenada por código de

producto.

a) Realizar un módulo recursivo que reciba la lista generada en el punto a y retorne la suma de las

cantidades vendidas para los productos con código menor que 80.

Imperativo

Se lee información acerca de las ventas de productos realizadas en las 5 sucursales de una empresa.

Cada sucursal realizó a lo sumo 180 ventas. De cada venta se conoce el código de producto, cantidad

vendida y monto total de la venta. Las ventas de cada sucursal se leen de manera consecutiva y

ordenadas por código de producto. La lectura por cada sucursal finaliza al completar las 180 ventas o

cuando se lee el código de producto -1, el cual no se procesa. Implementar un programa para que a

partir de la información leida, resuelva los siguientes ítems:

a) Utilizando la técnica de merge o merge acumulador según corresponda, generar una lista que

contenga el monto total vendido para cada código de producto, ordenada por código de producto.

b) Realizar un módulo recursivo que reciba la lista generada en el punto a y retorne la cantidad de

productos para los cuales el monto total vendido es inferior a 300.000 pesos.