Pere Quílez y Nicolás Vila

Lab 5 Report

El objetivo de esta práctica, ha sido implementar un sistema capaz de llevar un control del stock y las ventas de una librería. Las clases y métodos a implementar no se incluirán en este documento ya que nos fueron dadas en el propio documento de presentación del Lab.

En primer lugar, la clase *Book* consta de 5 atributos y 5 métodos(a excepción del constructor) donde cada método es simplemente un *getter* de cada uno de los atributos.

A continuación, la clase *Catalog*, extiende, es decir, es también una *BookCollection*. Esta, no tendrá ningún método a parte del constructor, que llama al constructor de la superclase.

La clase *ShoppingCart*, se dedica a controlar qué copias de cada libro se extraen de la tienda y se dipositan en el carrito del usuario para proceder a la compra. Tendrá un atributo *catalog*, de tipo *Catalog* y además, tendrá la *collection* de la superclase de *ShoppingCart*(*BookCollection*). Será en esta donde se guarden los libros con las copias correspondientes en el carrito mientras en el atributo *catalog*, tendremos una copia de las existencias de la tienda. El constructor, en primer lugar llama al constructor de la superclase y se inicializa *catalog* con el catálogo de la tienda. Además hemos redefinido un método *getStock*(al que además de *booktitle*, le entra como argumento la colección en la que debe de buscar) para que sea más sencillo encontrar las copias de ciertos libros en los métodos de *addcopies* y *removecopies*. En cuanto a estos dos métodos, en el primero de ellos, en primer lugar llamamos al método *getStock* para encontrar la existencia del libro en el catálogo de la tienda. En caso de que exista el número de copias, llamaremos de nuevo al método *getStock*, pero ahora para obtener dicho libro en *collection*. Por último, añadiremos las copias que se especifíquen en la *collection* y se hará remove de dichas copias en *catalog*. La implementación de *remove* es la misma que *add* pero de manera inversa.
Para el método de *totalprice* simplemente llamamos a *totalprice* de cada Stock que haya en *collection* (carrito).
Por último, en el método *payment*, a parte de llamar al método *doPayment* en el *return*, lo que se hace es eliminar las copias que haya en collection de cada *Stock* ya que representa que ya las ha comprado y por lo tanto, dejan de estar en el carrito.

La clase *Stock*, a parte de tener la información de cada libro al tener un atributo *book*, también tiene información sobre el número de copias de cada libro, el precio, y la moneda. Tiene métodos getters para obtener dichos atributos además de tener métodos *addcopies*, *removecopies* y *totalprice*, que hacen lo pertinente.

Por último, para hacer la clase *TestStore*, inicializamos 2 catálogos (uno para las existencias de la librería y otro para el carrito de compra). Lo que haremos en primer lugar será leer el fichero con los libros que haya y se pasará cada tipo de dato al correspondiente. A continuación, creamos una variable de tipo *Book* que almacene los datos del libro leído y 2 variables de *Stock* de las cuales una de ellas tendrá el número de copias inicializadas a 0. ésta ultima variable con las copias a 0, será añadida al catálogo del carrito (cart) mientras la otra, al de la tienda (books). Finalmente, creamos una *shoppingcart* inicializada con los libros de la tienda y establecemos el atributo collection de dicha *shoppingcart* con el catálogo de cart (con las copias a 0). Finalmente, llamamos al constructor de *Bookstore* con los parámetros que le corresponde para ejecutar la tienda.

Cabe destacar que tuvimos ciertas dudas primordialmente en las clases de *ShoppingCart* y de *TestStore* ya que en primer lugar, no entendimos bien cómo se relacionaba *catalog* con collection, como se accedía a cada una y, en general, la abstracción del diseño. Por otro lado, tampoco estábamos seguros de donde inicializar cada conjunto de libros (si en el constructor o en el *main*). Sin embargo, al final pudimos ir resolviendo estas dudas.