Implement a **shop management application**. To do this, a menu will be displayed on the terminal through which the user will be able to choose between different options.

We will work with digital and physical products. The properties of the products are:

- Digital Product: shall consist of "name", "type", "price", "id" and "size" (in MB).
- Physical Product: shall consist of "name", "type", "price", "id" and "weight" (grams).

The type of the products shall be limited to only "Book", "CD", "Painting", "Software" or "DVD". Depending on the type, the product will be Digital or Physical.

The menu shall have the following options:

- **a)** **Add new product**: Adds a new product by asking the different parameters to the user. Ensure that the ID is new (duplicated IDs are not accepted) and the Type is one of the accepted five.
- **b)** **Show ALL the products details**: Prints all products details (one per line).
- **c)** **Show details of a given product name**: Prints the details of a product according to its name. If there're different products with the same name, all must be displayed (two "El Quijote" books or a Queen DVD and a CD).
- **d)** **Show the details of a given type of products**: Prints all products of a given Type.
- **e)** **Modify the price of a product**: Increases/Decreases the price of a product according to its ID and a given value by the user.
- **f)** **Show products above/below a given price**: Prints the products that are above and below a price value given by the user.
- **g)** **Calculate total SUM of all products**: Prints the SUM of all products price.
- **h)** **Calculate SUM of a given type of products**: Prints the SUM of all products price of a given Type.
- **i)** **Sort the list of products by a selectable field**: Prints the details of all products sorting the output by Name, Type, Price or ID
- **j)** **EXIT**: Program finished

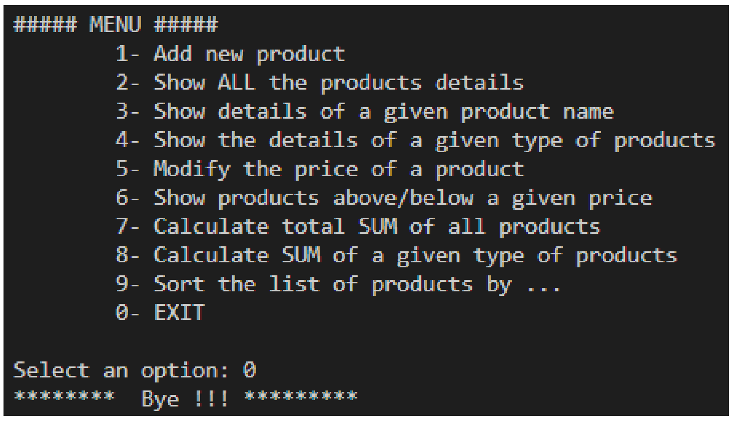
The student is free to choose how to implement all the functionalities, but the grade of the activity will be based on the suitability of the chosen structures, the correct implementation of classes and methods, as well as the use of the elements explained in class. Any code based on implementations not explained in class could be considered plagiarism.
Bad coding practices and code with poor coupling and cohesion will decrease the final grade.
All project must be implemented **in a single file**.

ERROR messages:

User must always get a reply when selecting an option or performing a task. That means that if an action returns and empty or incorrect value, the users must be informed. In case a single action does not returns a message, the corresponding grading item will be '0'.

EXCEPTIONS

Program shall handle the exceptions wherever they may happen. On an exception, a message must be displayed and the action must be repeated (question, user keyboard input …) until properly done. In case a single exception is not handled or the action not repeated, the corresponding grading item will be '0'.



```
##### MENU #####
        1- Add new product
        2- Show ALL the products details
        3- Show details of a given product name
        4- Show the details of a given type of products
        5- Modify the price of a product
        6- Show products above/below a given price
        7- Calculate total SUM of all products
        8- Calculate SUM of a given type of products
        9- Sort the list of products by ...
        0- EXIT

Select an option: 0
********  Bye !!! *********
```

Physical Products examples

ID	Type	Name	Price	Weight (g)
101	Book	El Quijote	25.54	800
202	Painting	Dogs and Cats	124	300
300	Book	El Quijote	552.50	1500
104	Book	Hamlet	32.01	700
…				

Digital Products examples

ID	Type	Name	Price	Size (MB)
305	CD	Hamlet	30	15
306	SW	MS Office	230.5	1100
408	DVD	Queen	78.5	3500
109	DVD	Hamlet	80	2800
…				