PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

الجمهورية الجزائرية الديمقراطية الشعبية

MINISTRY OF HIGHER EDUCATION
AND SCIENTIFIC RESEARCH

SCHOOL OF APPLIED SCIENCES
--TLEMCEN--

وزارة التعليم العالي والبحث العلمي

المدرسة العليا في العلوم التطبيقية
-تلمسان-

Course : Object Oriented Programming
Practical Work N°2 : Constructors, Inheritance and Interface

I- Supermarket items

Write a class **Article** to represent items sold in a supermarket.
Each one has four instance variables:

private long reference; // a number that uniquely identifies the item
private String description; // the description of the item in text form
private float priceET; // the unit price of the item excluding tax
private int quantityInStock; // the number of units of the item available in stock

and provides a number of methods

public Article(long reference, String description, float priceET, int quantityInStock) : Immediate constructor that assigns a value to each instance variable.
public long getReference() public String getDescription() public int getQuantityInStock() : Accessor methods that allow retrieving the value of each instance variable.
public void restock(int numberOfUnits) : method to increase the available quantity of the item
public boolean sell(int numberOfUnits) : This method records the sale of a certain number of units of the item, and therefore decreases the available stock accordingly. If the number of units is greater than the available quantity, then the stock is not modified and the method returns false; otherwise, it returns true.
public float priceET() : This method calculates and returns the price excluding tax of the item.
public float prixIAT() : This method calculates and returns the price including all taxes of the item.
public String toString() : A string expressing the reference, the description, and the price of the item.
public boolean equals(Article anArticle) : a.equals(b) is true if and only if a and b represent the same item (i.e., if they have the same reference).

II. Inheritance

Inheritance is the ability of a class (the subclass) to inherit members (attributes and methods) from another class (the superclass). The subclass A can override methods of its superclass B in order to specialize them.
Create the following City class:

```
class City{
private int nbHabitants;
private String Cityname;
public City(String Cityname);
String getCityname();
int getnbHabitants();
void setnbHabitants(int nb);
String toString(); }
```

1- Provide the code corresponding to the functions of the City class.
2- Input 5 different cities into a test class that uses the City class, and use a for loop to call the **toString()** method for all the cities.
3- Overload the constructor of the City class by defining a constructor with two arguments (a String and an integer).
4- Create a **Capital** class that inherits from the City class. This class will include an additional instance variable: **countryName**. Test different cases: explicit or implicit call to the superclass constructor; existence or absence of a no-argument constructor.
5- Override the **toString()** method, calling the superclass's **toString()** method. It should display: Capital of **countryName: cityName; numberOfInhabitants.** Test it.
6- Change the access modifiers of the member variables in the superclass from **private** to **protected**. Can these variables be accessed from outside the City class?

III. Interface

For managing a library, we are asked to write an application dealing with documents of various types: books, which can be novels or textbooks, dictionaries, etc. All documents have a registration number (an integer) and a title (a string). Books have an author (a string) and a number of pages (an integer). Novels may optionally have a literary prize (an integer constant, among: GONCOURT, MEDICIS, INTERALLIE, etc.), while textbooks have a school level (an integer), and dictionaries have a language (a string).

The objects Novel and Textbook can be handled as Books. Define the classes Book, Novel, Textbook with the appropriate inheritance relationships, as well as Dictionary and the Document interface, as suggested by the previous description. In each of these classes, define:

1- The constructor that takes as many arguments as there are instance variables and only initializes these variables with the values of the arguments.
2- If you declared the instance variables as private, also define public "accessor" methods (get...) to retrieve the values of these variables.
3- Write an executable TestDocuments class that creates and displays several documents of different types.