Université Mohammed Premier
Oujda
Faculté des Sciences
Département d'Informatique

Année universitaire 2019/2020
Filière SMI, Semestre 5
Examen de la session normale
06 Janvier 2020

Examen en POO-Java
Durée : 1h30

Exercice 1 (7pts)

Soit le programme en Java suivant :

```
class Vehicule{
  public void afficher(){
    System.out.println("J'ai quatre roues");
  }}
class Voiture extends Vehicule{
  public void afficher(){
    System.out.println("Je suis une voiture");
    }}
class Transport{
  public void demarrer(Vehicule v){
    System.out.println("Un vehicule demarre");
    v.afficher();
    }
  public void demarrer(Voiture v){
    System.out.println("Une voiture demarre");
    v.afficher();
    }}
public class EssaiVehicule{
  public static void main(String args[]){
    Transport.demarrer(new Vehicule());
    Transport.demarrer(new Voiture());
    Vehicule A = new Voiture();
    Transport.demarrer(A);
    }}
```

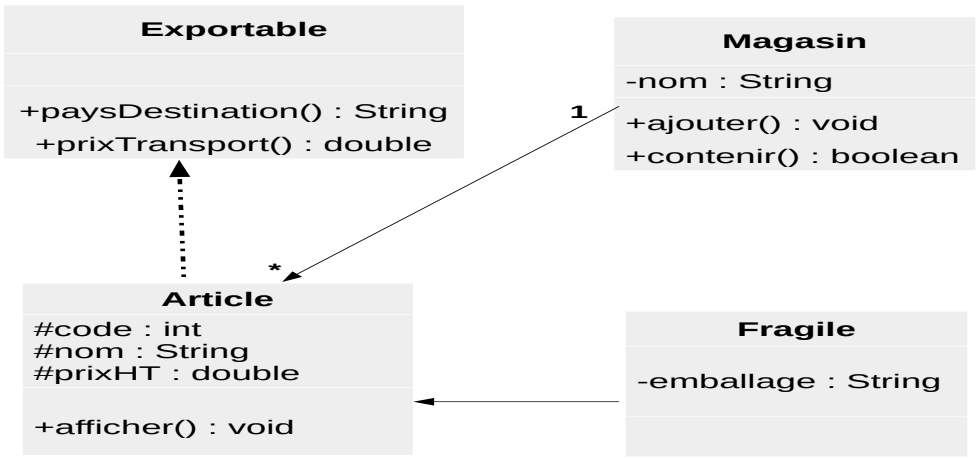
La compilation de ce code génère l'erreur suivante : *error : non-static method demarrer(Vehicule) cannot be referenced from a static context Transport.demarrer(new Vehicule()).*

1. Corrigez l'erreur (les erreurs) ;
2. Qu'affiche le programme ci-dessus après correction ?

Exercice 2 (13pts)

La phase d'analyse et de conception du problème de la gestion des ventes d'articles dans un magasin a conduit au diagramme de classes ci-dessous.

1. Programmez l'interface **Exportable** telle qu'elle est indiquée dans le diagramme de classes.



2. Écrivez la classe **Article** qui implémente l'interface **Exportable** et qui contient :
   — trois attributs : un code qui s'incrémente automatiquement, un nom et un prix hors taxe (prixHT) ;
   — un constructeur qui a deux arguments (nom et prixHT) et qui initialise les trois attributs ;
   — le programme doit lancer une exception si le prix hors taxe est négatif ou nul ;
   — pour implémenter la mèthode *paysDestination()*, on demande à l'utilisateur d'entrer le pays voulu ;
   — la mèthode *prixTransport()* retourne 5% du Prix Hors Taxes de l'article concerné ;
   — une mèthode *afficher()* qui affiche le code, le nom, le pays et le prix total. Ce dernier est la somme du prixHT et le prix de transport.
3. Écrivez la classe **Fragile** qui hérite de la classe **Article** et qui contient :
   — un attribut emballage (de plus) ;
   — une redéfinition de la méthode *prixTransport()* pour que le prix de transport d'un article fragile devienne deux fois le prix de transport d'un Article normal.
4. Écrivez la classe **Magasin** qui contient un nom et un ensemble d'articles (le nombre n'est pas défini).
   — Ajoutez une méthode *ajouter()* qui permet d'ajouter un article au magasin ;
   — Ajoutez une méthode *contenir()* qui vérifie si un article est disponible au magasin.
5. Écrivez la classe principale **Test** qui contient la mèthode *main()*.
   — Déclarez un tableau de quatre articles. Puis, instanciez les deux premiers éléments par des articles et les deux derniers par des fragiles ;
   — Déclarez un magasin qui contient les trois premiers éléments du tableau ;
   — Vérifiez si le magasin contient le quatrième élément du tableau, puis ajoutez ce dernier élément au magasin.