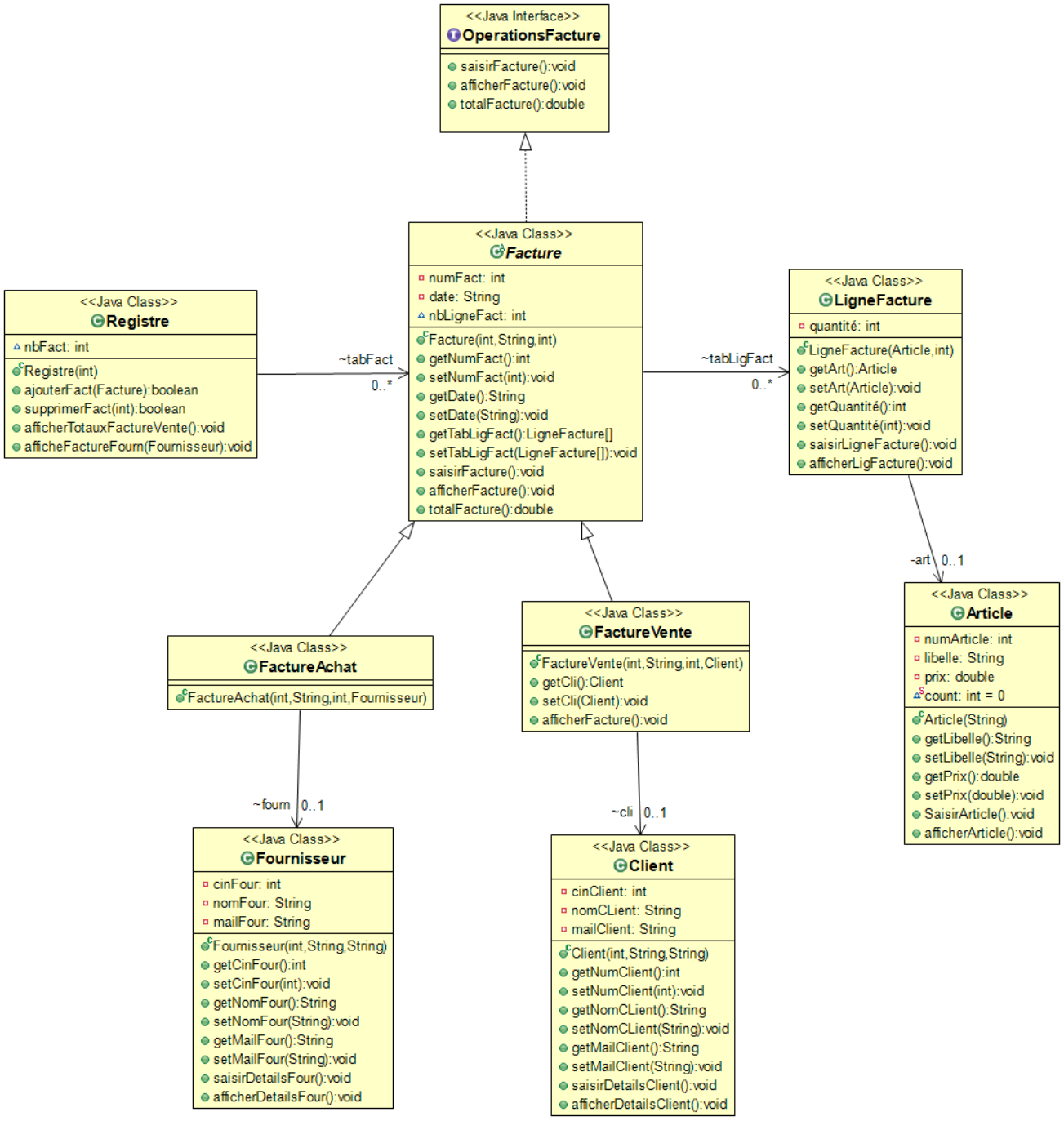
Problème

On propose de construire une petite application pour la gestion des factures achats et des factures ventes pour une petite société commerciale.



A. La classe Article

1) Déclaré les attributs (visibilité privée) de la classe **Article** sachant que **numArticle** est géré en séquentiel (numéro d'ordre de l'article).
2) Donner le code du constructeur de la classe **Article** ayant comme paramètre le libellé et le prix de l'article. (**NB** : numArticle est séquentiel).
3) Donner le code source de la méthode **SaisirArticle()** permettant de donner la main à l'utilisateur pour saisir le libellé et le prix d'un article.
4) Implémenter la méthode **afficherArticle** permettant d'afficher les caractéristiques d'un article.

B. La classe LigneFacture

5) Définir les attributs et le constructeur de **LigneFacture**
6) Définir la méthode **saisirLigneFacture** qui donne la main à l'utilisateur pour saisir les informations d'une **LigneFacture**.
7) Définir la méthode **afficherLigneFacture()** qui affiche toutes les caractéristiques d'une LigneFacture.

C. L'interface OpérationsFacture

8) Définir les méthodes de l'interface **OpérationsFacture**

D. La classe Facture

Bien que la classe **Facture** implémente toutes les méthodes énoncées précédemment **nous allons la déclarer abstraite.**

Une Facture est composée par un ensemble de **LigneFacture**. Elle est caractérisée par son numéro, la date de la facture et le nombre de ligneFacture

9) Donner la déclaration de la classe abstraite **Facture** qui implémente l'interface **OperationsFacture** avec tous ses attributs.
10) Définir un **constructeur** pour **la classe abstraite Facture** permettant d'initialiser les valeurs des attributs et créer un tableau de ligne facture dont la taille maximale (`nbLigneMax`) est passée en paramètre.

11) Générer les getters et setters des attributs

12) Définir la méthode saisirFacture qui donne la main à l'utilisateur pour lire le numéro de facture, la date ainsi que les différentes ligneFacture.

13) Donner le code source de la méthode **afficherFacture( )** permettant d'afficher toutes les informations relatives à une **Facture** (numFact, date et toutes les lignes factures).

E. La classe Client et la classe Fournisseur

14) Définir les classes Client et Fournisseur telle qu'elles sont définies dans le diagramme de classe

F. Les classes FactureVente

15) Définir la classe **FactureVente** qui dérive de la classe Facture. Une **FactureVente** concerne un **Client**.

16) Implémenter le constructeur de la classe FactureVente.

17) Implémenter la méthode saisirFactureVente qui permet de lire toutes les infos d'une factureVente

18) Redéfinir la méthode **afficherFacture( )** permettant d'afficher toutes les informations relatives à une **Facture vente**.

G. Les classes FactureAchat

19) Définir la classe **FactureAchat** qui dérive de la classe Facture. Une **FactureAchat** concerne un **Fournisseur**.

20) Implémenter le constructeur de la classe **FactureAchat**.

21) Implémenter la méthode saisirFactureAchat qui permet de lire toutes les infos d'une factureAchat

22) Redéfinir la méthode **afficherFacture( )** permettant d'afficher toutes les informations relatives à une **Facture vente**.

H. La classe Registre

La classe **Registre** est caractérisée par un tableau de Facture tabFact et le nombre de factures contenues dans un registre **nbFact**.

23) Déclarer la classe **Registre** avec ses attributs.

24) Définir le constructeur de la classe **Registre** permettant de créer le tableau de factures dont la capacité maximale est passée en paramètre.

25) Implémenter la méthode **ajouterFacture** qui permet d'ajouter une Facture passée en paramètre au tableau de facture. Elle retourne true si l'ajout est possible false sinon.

26) Définir la méthode **supprimerFact** permettant de supprimer la facture dont le numéro de facture est passé en paramètre et qui retourne true si cela est possible et false sinon.

27) Définir la méthode **afficheFactureFourn()** qui affiche toutes les caractéristiques des **FactureAchat** du tableau.

28) Définir la méthode **afficherTotauxFactureVente()** qui affiche pour chaque **FactureVente** du tableau le **numFact** et le **total de la Facture.**

I. La classe FacturationTest

Cette classe va vous permettre de tester toutes les fonctionnalités ue vous avez implémentez.