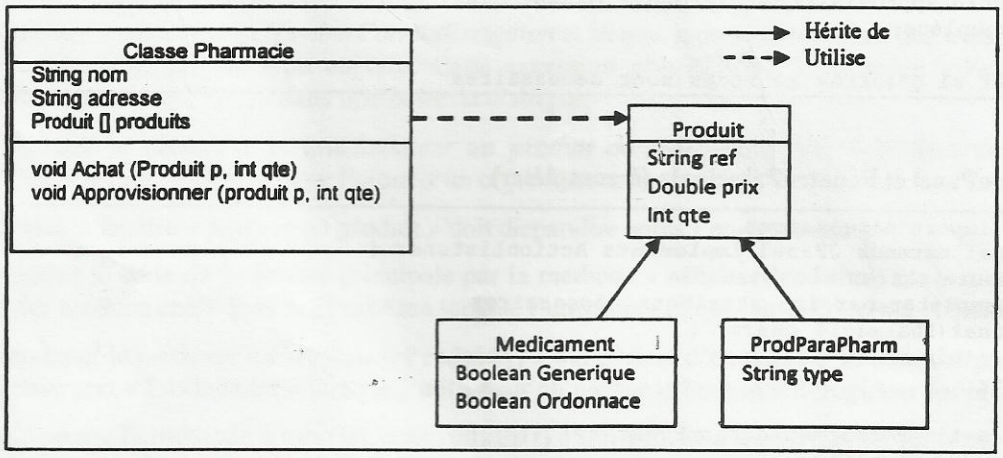
Examen Pratique
Programmation Orienté Objet – Java
Durée : 2 heures

Consignes :

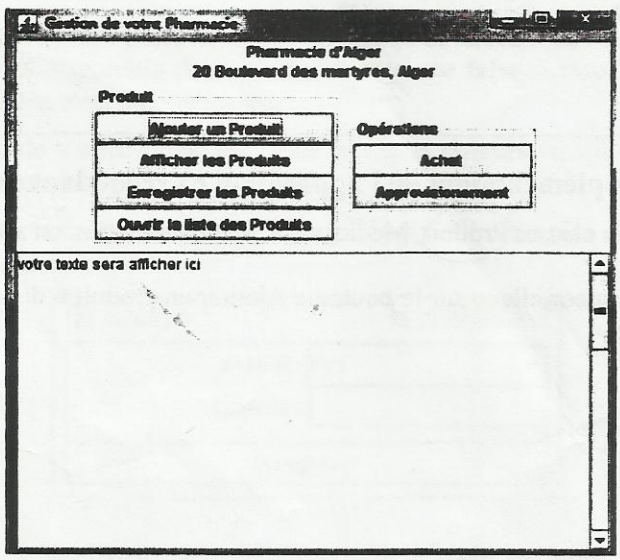
- L'utilisation d'un ordinateur personnel est **strictement interdite**. En cas de problème, signaler rapidement l'anomalie à l'Enseignant.
- Seule la « JDK 6 Documentation » est autorisée (disponible sur le bureau de la machine). Tout autre document est **interdit.**
- Ouvrez Eclipse à travers le raccourcis qui se trouve sur le bureau et renommer le projet **NomPrenomGroupe** par votre nom, prénom et numéro de groupe, le tout collé, sans espaces ni accents. Ce projet contient une partie de l'application déjà implémentée.
- Il est possible d'utiliser des méthodes non implémentées pour implémenter d'autres méthodes.
- Les interfaces graphiques peuvent ne pas être identiques à celles proposées dans l'énoncé.

Enoncé :

Vous devez programmer une application de gestion des produits d'une pharmacie. Voici sa modélisation orientée objet :



Cette application doit être dotée d'une interface graphique permettant d'ajouter des produits, les afficher, les enregistrer, accomplir une opération d'achat ou d'approvisionnement, comme le montre la figure suivante :



Une partie des classes suivantes est déjà implémentée :

1. Classe Pharmacie (à compléter):

```
// tous les import nécessaires
public class Pharmacie
{
    private String nom;
    private String adresse;
    private ArrayList<Client> clients;
    private ArrayList<Produit> produits;

    public Pharmacie(String nom, String adresse){
        this.nom = nom;
        this.adresse=adresse;
        this.produits= new ArrayList<Produit>();
    }

    public boolean Achat(Produit p, int qte){
       //à compléter
    }
    public void Approvisionner (Produit p, int qte){
      //à compléter
    }
//à compléter si d'autres méthodes sont nécessaires
```

2. Classes MonPanel et FenetrePrincipale (à compléter)

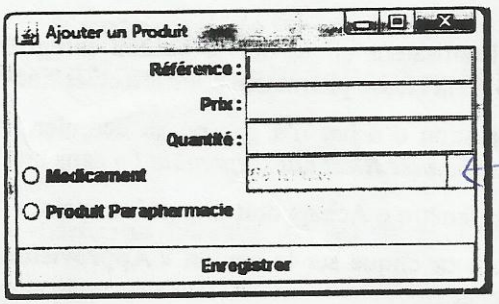
```
// tous les import nécessaires
class MonPanel extends JPanel implements ActionListener {
      Pharmacie pharm;
      //à compléter par les attributs nécessaires
public MonPanel(Pharmacie pharm) {
            this.pharm=pharm;
//à compléter
}
public void actionPerformed(ActionEvent ev)  {
            //à compléter
  }
}
public class FenetrePrincipale {

      public static void main(String args[]) {
         Pharmacie pharm=new Pharmacie("Pharmacie d'Alger", "20 Boulevard des martyres, Alger");
      //à completer
    }
}
```

Questions: Compléter l'implémentation de l'application en répondant aux questions suivantes :

Q1. 3pts - Implémenter les classes Produit, Medicament, ProdParaPharm, en sachant que la classe Produit ne peut pas être instanciée.

Q2. 6pts- Lorsque l'utilisateur clique sur le bouton « Ajouter un produit » de la fenêtre principale, la fenêtre suivante doit s'afficher :



a. Créer la classe « **FenAjoutProduit** » qui permet d'afficher cette fenêtre (voir l'annexe pour l'utilisation des boutons radios). Le constructeur de cette classe doit avoir en paramètre la pharmacie « **pharm** » que le système gère.

b. Lorsque l'utilisateur clique sur le bouton ajouter,

- Le programme doit vérifier si tous les champs ont des valeurs

- générer une exception *NumberFormatException* si le type introduit dans l'un des trois premiers champs ne correspond pas au type attendu. Cette exception doit afficher le message « *Veuillez vérifier les information introduites* » dans une boite de dialogue.

- Ajouter le produit créé (*médicament* **ou** *produit de parapharmacie*) à la liste des produits de la pharmacie (voir l'annexe pour l'ajout d'un objet dans un ArrayList).

- Enfin la fenêtre « Ajouter un produit » doit disparaître après l'enregistrement.

Q3. 1pt - Compléter le code de la fenêtre principale par la méthode « **afficherProduits()** » qui permet d'afficher la liste des produits créés dans le JTextArea lorsque l'utilisateur clique sur le bouton « Afficher les Produits».

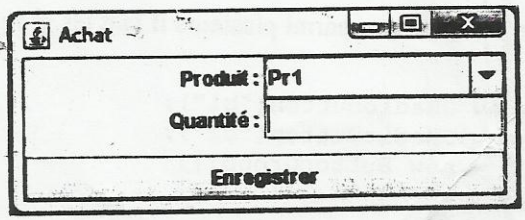
Q4. 1pt - Implémenter la méthode « **enregistrerProduits()** » qui permet d'enregistrer la liste des produits créés dans un fichier text « Produits.txt » lorsque l'utilisateur clique sur le bouton « Enregistrer les produits».

Q5. 2pts - Implémenter la méthode « **ouvrirListeProduits ()** » qui permet d'ouvrir un fichier texte et d'afficher son contenu dans le JTextArea de la fenêtre principale. Cette méthode est appelée lorsque l'utilisateur clique sur le bouton « Afficher la liste des produits». Elle doit permettre de parcourir les dossiers et fichiers à la recherche du fichier contenant la liste des produits avec JFileChooser. Si aucun fichier n'est sélectionné, cette méthode affiche dans une boite de dialogue le message « *Aucun fichier choisi!* ».

Q6. 1pt - Compléter le code de la classe pharmacie en implémentant la méthode « **achat** » qui soustrait la quantité demandée de la quantité disponible du produit. Elle retourne **false** si l'opération d'achat est impossible, et **true** si l'opération d'achat s'est bien déroulée

Q7. 1pt - Implémenter la méthode « **approvisionner** » de la classe pharmacie, qui rajoute la quantité fournie à la quantité disponible du produit.

Q8. 2pts. - Compléter le code de la fenêtre principale afin que lorsque l'utilisateur clique sur le bouton « Achat », la fenêtre suivante s'affiche :



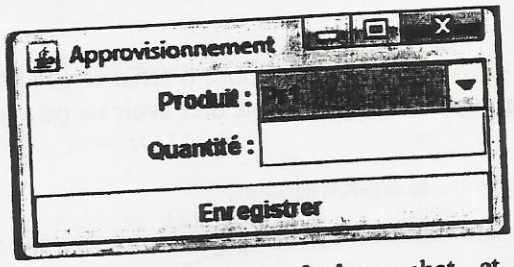
a. Créer la classe « **FenAchat** » qui permet d'afficher cette fenêtre.

b. Lorsque l'utilisateur clique sur le bouton enregistrer, le programme doit appeler la méthode « **achat** » de la classe pharmacie pour effectuer l'achat :

- Si l'opération d'achat n'a pas pu se dérouler, le programme doit afficher le message « *La quantité demandée n'est pas disponible !* » dans une boite de dialogue.

- Sinon, la fenêtre « Achat» doit disparaître après.

Q9. 1,5pts - Lorsque l'utilisateur clique sur le bouton « **Approvisionnement** » de la fenêtre principale, la fenêtre suivante doit s'afficher :



Créer la classe « **FenApprov** », qui ressemble à la fenêtre achat, et qui permet d'appeler la méthode « Approvisionnement » de la classe pharmacie.

Q10. 1,5pts - Créer l'applet « **FenAppletPrincipale** » dans laquelle vous transformez l'application fenêtre principale en une Applet.

Annexe

1. ArrayList

Déclaration d'un ArrayList « ar »:

```
ArrayList<classeObjet> ar= new ArrayList<classeObjet>();
```

Ajout d'un élément « o» à un ArrayList « ar »:

```
ar.add(o) ; // o est un objet de type classeObjet
```

Parcourir un ArrayList « ar »:

```
for(classeObjet o : ar){
        //manipulation de l'objet o et suite du code
}
```

Récupérer l'élément « o » à la position « i » d'un ArrayList « ar » :

```
classeObjet o= ar.get(i);
```

2. Groupe de Radio boutons

Lorsque on doit choisir un radio bouton parmi plusieurs il faut les grouper avec un objet GroupButton pour qu'un seul soit sélectionné à la fois :

```
JRadioButton b1=new JRadioButton("b1");
JRadioButton b2=new JRadioButton("b2");
ButtonGroup group = new ButtonGroup();
group.add(b1);
group.add(b2);
```