Programmation
Examen

Nous nous intéressons à des arbres constitués de nœuds ayant zéro, un ou deux fils. Les nœuds constituant ces arbres sont des instances des classes *Nœud*, *NoeudUnaire* et *NoeudBinaire*.

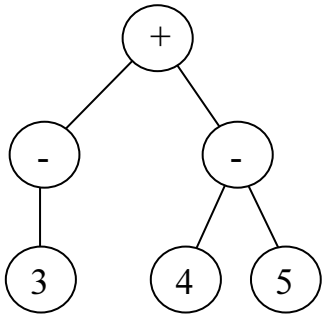
```
public class Noeud {
   private String nom;
   public Noeud(String nom) ;

}
public class NoeudUnaire  extends Noeud{
   private Noeud fils;
   public NoeudUnaire(String nom, Noeud nf) ;
   public Noeud getFils() ;
}
public class NoeudBinaire  extends Noeud{
   private Noeud filsG;
   private Noeud filsD;
   public NoeudBinaire(String nom, Noeud ng, Noeud nd)
   public Noeud getFilsG() ;
   public Noeud getFilsD() ;
}
```

Question 1. (4 points)

Ecrire les définitions des constructeurs des trois classes, ainsi que les définitions des méthodes *getFils()*, *getFilsG()* et *getFilsD()*.

Nous désirons ajouter deux fonctionnalités : le calcul du nombre de noeuds de l'arbre et le calcul de la forme parenthésée.

La forme parenthésée de l'arbre 

est `+(-(3),-(4,5))`.

Nous programmerons l'ajout de ces 2 fonctionnalités de 3 façons différentes : en ajoutant des méthodes aux classes, en créant des méthodes externes aux classes, et en implantant le modèle de conception « visiteur ».

Ajout de méthodes

Question 2. (3 points)

Ajouter aux trois classes la méthode *int nbNoeuds ()* qui permet de calculer le nombre de noeuds d'un arbre donné par un nœud qui est sa racine.

Question 3. (3 points)

Ajouter aux trois classes la méthode *String formeParenthesee ()* qui permet de calculer la forme parenthésée d'un arbre donné par un nœud qui est sa racine.

Une classe de méthodes utilitaires

Nous définissons la classe Utilitaire de la façon suivante :

```
public class Utilitaires {
  …
  public int nbNoeud(Noeud arbre){…}
  public String formeParenthesee(Noeud arbre){…}
}
```

Question 4. (3 points)

Définir la méthode *int nbNoeuds ()* de la classe *Utilitaires* qui permet de calculer le nombre de noeuds d'un arbre.

Question 5. (3 points)

Définir la méthode *String formeParenthesee ()*de la classe *Utilitaires* qui permet de calculer la forme parenthésée d'un arbre.

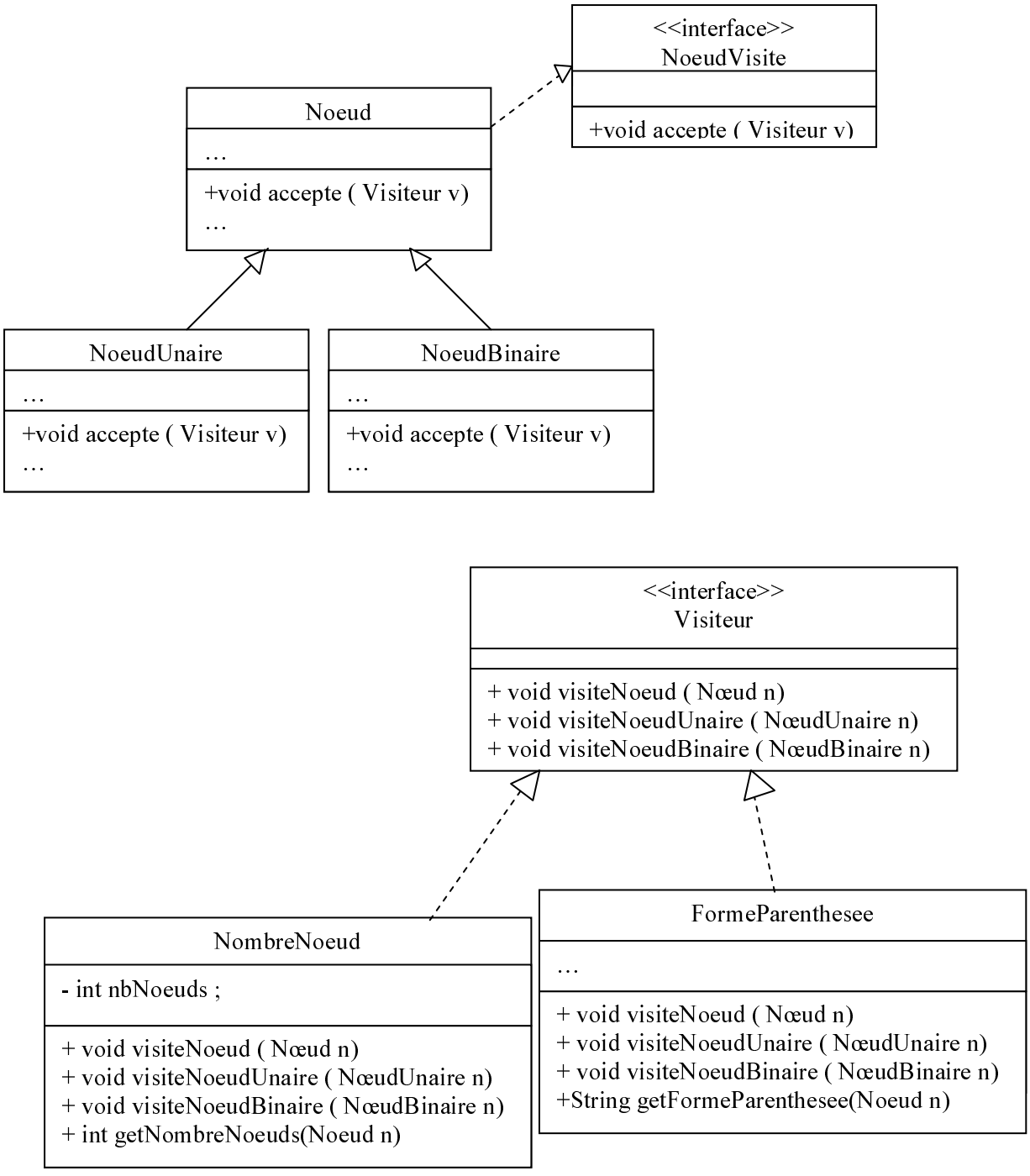
Le patron de conception « visiteur »

La classe *Nœud* et ses classes dérivées implémentent l'interface *NoeudVisite*, qui contient une seule méthode public *void accepte(Visiteur v*). Cette méthode accepte la visite d'un objet, et appelle la méthode de l'interface visiteur qui correspond à sa classe : la méthode *accepte* de la classe *Nœud* est définie par : `public void accepte(Visiteur v){ v.visiteNoeud(this); }` et la méthode *accepte* de la classe *NoeudUnaire* est définie par `public void accepte(Visiteur v){v.visiteNoeudUnaire(this);}`

Question 6. (4 points)

Définir les méthodes de la classe *NombreNoeud*, la méthode *getNombreNoeuds(Nœud n)* est définie par :

```
public int getNombreNoeuds(Noeud n){
    nbNoeuds = 0;
    n.accepte(this);
    return nbNoeuds;
}
```


<<interface>>
NoeudVisite
+void accepte (Visiteur v)
Noeud
…
+void accepte (Visiteur v)
…
NoeudUnaire
…
+void accepte (Visiteur v)
…
NoeudBinaire
…
+void accepte (Visiteur v)
…
<<interface>>
Visiteur
+ void visiteNoeud (Nœud n)
+ void visiteNoeudUnaire (NœudUnaire n)
+ void visiteNoeudBinaire (NœudBinaire n)
NombreNoeud
- int nbNoeuds ;
+ void visiteNoeud (Nœud n)
+ void visiteNoeudUnaire (NœudUnaire n)
+ void visiteNoeudBinaire (NœudBinaire n)
+ int getNombreNoeuds(Noeud n)
FormeParenthesee
…
+ void visiteNoeud (Nœud n)
+ void visiteNoeudUnaire (NœudUnaire n)
+ void visiteNoeudBinaire (NœudBinaire n)
+String getFormeParenthesee(Noeud n)