Dans ce TD nous étudions une généalogie inversée. Chaque personne connaît uniquement ses parents, ce qui peut être modélisé par un arbre binaire donné par la classe `Personne`.

```
class Personne {
  private final String prenom, nomDeFamille;
  private Personne mere, pere;

  public Personne (String prenom, String nomDeFamille) {
    ...
  }
  public Personne (String prenom, String nomDeFamille,
                   Personne pere, Personne mere) {
    ...
  }
}
```

Exercice 1

1. Écrire une méthode `boolean estFrereOuSoeur(Personne p)` qui teste si la personne courante (`this`) est un frère ou une sœur de `p`.
2. Écrire une méthode `boolean estCousinGermain(Personne p)` qui teste si la personne courante (`this`) est cousin germain de `p`.
   Rappel : deux personnes sont cousins germains si un parent de l'un et le frère ou la sœur d'un parent de l'autre.

Exercice 2

3. Écrire une méthode `int nbAscendantsVivants()` qui renvoie le nombre d'ascendants de `this` (c'est-à-dire le nombre de nœuds accessible depuis `this` dans l'arbre).
4. Écrire une méthode `boolean possedeCommeAscendant(Personne p)` qui teste si la personne courante (`this`) a pour ascendant `p`.

Exercice 3 La *distance d'ascendance* entre deux personnes est le nombre de générations les séparant. Par exemple la distance entre un père et son fils est 1, entre un grand-père et sa petite-fille est 2, etc. La distance d'ascendant entre quelqu'un et lui même vaut 0, et si aucun des deux n'est l'ascendant de l'autre, leur distance est −1.

5. Écrire une méthode `int distanceDAscendance(Personne p)` qui donne la distance d'ascendance entre `this` et `p`.
6. Écrire une méthode `void afficheAscendantUn(Personne p)` qui si `p` est un ascendant direct de `this` affiche la lignée comme ceci :

   $$\underbrace{\text{Jean Dupont}}_{\texttt{this}}\text{, enfant de Jeanne Martin, enfant de }\underbrace{\text{Michel Martin}}_{\texttt{p}}$$

7. Écrire une méthode `int nbDeGenerations()` qui renvoie la distance d'ascendance maximal entre `this` et un de ses ascendants. En d'autres termes, cette méthode calcule la longueur de la branche la plus longue de l'arbre.

Exercice 4

7. Écrire une méthode `boolean Verification()` qui renvoie `true` si l'arbre respecte les règles de la famille "traditionnelle" :
   - Un enfant possède le même `nomDeFamille` que son père.
   - Les deux parents d'un enfant ne peuvent être ni frère et sœur, ni cousins germains.

Exercice 5 (Facultatif)

8. Écrire une méthode `ArrayList<Personne> getTousLesAscendants()` qui renvoie la liste de tous les ascendants de this.
9. Écrire une méthode `boolean estDeMaFamille(Personne p)` qui teste si la personne courante (`this`) et `p` ont un ascendant commun.
10. Écrire une méthode `int distanceDHeritage(Personne p)` qui donne la distance d'héritage, c'est-à-dire la somme des distances directe de `this` et de `p` à leur ascendant commun. Par exemple un oncle et son neveu sont à une distance de 3, deux cousins germains à une distance de 4 et un grand-père et son petit-fils à une distance de 2.