

Dans ce TD nous  tudions une g n alogie invers e. Chaque personne conna t uniquement ses parents, ce qui peut  tre mod lis  par la classe `Personne`.

```

1 public class Personne {
    private final String prenom, nomDeFamille;
3     private Personne mere, pere;

5     public Personne (String prenom, String nomDeFamille,
        Personne pere, Personne mere) {
7         ...
    }

    public Personne (String prenom, String nomDeFamille) {
11         ...
    }
13 }
  
```

Exercice 1

1.  crire une m thode `boolean estFrereOuSoeur(Personne p)` qui teste si la personne courante (`this`) est un fr re ou une s eur de `p`.
2.  crire une m thode `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 fr re ou la s eur d'un parent de l'autre.

Exercice 2

3.  crire une m thode `int nbAscendantsVivants()` qui renvoie le nombre d'ascendants de `this` (c'est- -dire le nombre de n uds accessible depuis `this` dans l'arbre).
4.  crire une m thode `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 g n rations les s parant. Par exemple la distance entre un p re et son fils est 1, entre un grand-p re et sa petite-fille est 2, etc. La distance d'ascendant entre quelqu'un et lui m me vaut 0, et si aucun des deux n'est l'ascendant de l'autre, leur distance est -1.

5.  crire une m thode `int distanceAscendance(Personne p)` qui donne la distance d'ascendance entre `this` et `p`.
6.  crire une m thode `void afficheAscendantUn(Personne p)` qui si `p` est un ascendant direct de `this` affiche la lign e comme ceci :

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

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

Exercice 4

8.  crire une m thode `boolean verification()` qui renvoie `true` si l'arbre des ascendants de la personne courante (`this`) respecte les r gles de la famille traditionnelle :
 - Un enfant poss de le m me `nomDeFamille` que son p re.
 - Les deux parents d'un enfant ne peuvent  tre ni fr re et s eur, ni cousins germains.

Exercice 5 (Facultatif)

9. Écrire une méthode `ListePersonne getTousLesAscendants()` qui renvoie la liste de tous les ascendants de `this`. Pour faire cela, écrivez également les classes `ListePersonne` et `CellulePersonne`.
10. Écrire une méthode `boolean estDeMaFamille(Personne p)` qui teste si la personne courante (`this`) et `p` ont un ascendant commun.
11. Écrire une méthode `int distanceDHeritage(Personne p)` qui donne la distance d'héritage, c'est-à-dire la somme des distances d'ascendance 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.