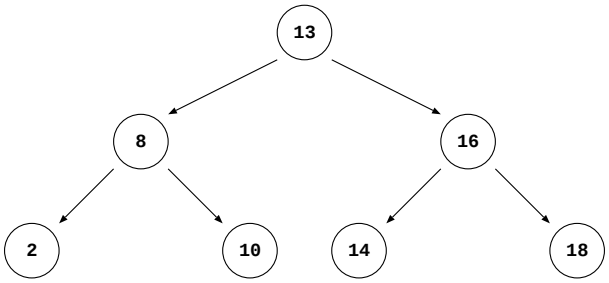
Programmation Orientée Objet
L2S4 2007–2008
hoenen@icps.u-strasbg.fr

Université Louis Pasteur
UFR de math-info
18 mars 2008

TD4 : récursivité et polymorphisme

Exercice 1 : *arbre binaire*



Les arbres binaires sont des structures permettant d'effectuer des recherche par valeur de manière efficace. Chaque nœud possède une donnée (ou clé) et peut avoir deux nœuds fils. La figure ci-dessus montre un tel arbre où les clés sont des entiers. Les noeuds sont rangés en fonction de la valeur de leur clé : le fils gauche d'un nœud a une clé plus petite que son père, et inversemment le fils droit a une clé plus grande que son père. Le nœud le plus haut et qui n'a pas de père est appelé la *racine*, et les nœuds les plus bas qui n'ont pas de fils sont appelés les *feuilles*.

On veut écrire une classe permettant de construire et de manipuler un arbre binaire de recherche qui stocke des chaînes de caractères. La comparaison entre les chaînes de caractères pourra s'effectuer avec la méthode `int compareTo(String)` qui prend en argument une chaîne de caractère et la compare lexicographiquement à la chaîne depuis laquelle la méthode a été appelée. L'entier renvoyé vaut 0 si les deux chaînes sont identiques, est négatif si la chaîne courante est plus petite que la chaîne en paramètre, et est positif sinon.

1. Écrivez la classe `Arbre` avec la méthode récursive `ajouter` qui prend en paramètre une chaîne de caractère et ajoute la chaîne au bon endroit dans l'arbre.
2. Ajoutez la méthode récursive `rechercher` qui prend une chaîne de caractère en paramètre et renvoie le sous-arbre dont la racine stocke la chaîne ou la feuille sous laquelle insérer la chaîne si celle-ci n'est pas présente dans l'arbre.
3. Écrivez une méthode affichant les éléments de l'arbre avec un parcours en profondeur d'abord.
4. Écrivez une méthode qui renvoie la plus grande clé de l'arbre et une autre qui renvoie la plus petite clé de l'arbre.
5. Écrivez une méthode pour tester si un nœd de l'arbre est une feuille, une autre pour comptre le nombre de feuilles, une pour compter le nombre de nœuds et enfin une dernière pour déterminer la *hauteur* de l'arbre. La hauteur de l'arbre est défini comme valant 0 à la racine de l'arbre, 1 au niveau des fils de la racine, 2 pour les fils des fils, etc...
6. Trouver les différents cas pouvant arriver lors de la suppression d'un nœud de l'arbre. Écrivez une méthode qui permet de supprimer de l'arbre une chaîne de caractère passée en paramètre.

Exercice 2 : *polymorphisme*

Étudiez les classes suivantes en faisant bien attention aux relations entre les *types déclarés* et les *types courants* :

```
class Animal {
  String nom;
  int age = 0;

  public Animal(String s) {
    nom = s;
  }

  public void vieillir() {
    age++;
  }

  public void dire() {
    System.out.println("...");
  }
}
```

```
public Test {
  public static void main(String[] args) {
    Animal a = new Animal("l'animal");
    Oiseau o = new Oiseau("le piaf");
    Perroquet p = new Perroquet("Coco");
    Animal[] tab = new Animal[5];
    tab[0] = a;
    tab[1] = o;
    tab[2] = p;
    tab[3] = new Oiseau("Birdy");
    tab[4] = new Perroquet("Roger");

    for (int i=0; i<5; i++) {
      tab[i].dire();
    }
  }
}
```

```
class Perroquet extends Oiseau {
  private boolean parle = false;

  public Perroquet(String s) {
    super(s);
  }

  public void apprendParole() {
    parle = true;
  }

  public void dire() {
    if (parle) {
      System.out.println(name);
    }
    else {
      super.dire();
    }
  }
}
```

```
class Oiseau extends Animal {
  protected boolean vole = false;

  public Oiseau(String s) {
    super(s);
  }

  public void apprendVole() {
    vole = true;
  }

  public void dire() {
    if (vole) {
      System.out.println("cui-cui");
    }
    else {
      System.out.println("piou-piou");
    }
  }
}
```

1. Donnez l'affichage du programme `Test`.
2. Que se passe-t-il si on rajoute `o.apprendVole()` et `p.apprendParole()` avant la boucle `for` ?
3. De même, comment faire voler `Birdy` et apprendre à parler `Roger`.
4. Si un oiseau est toujours un animal, comment vérifier qu'un animal est un oiseau ?
5. Modifiez le programme `Test` pour que chaque animal possède toutes ses facultés.