Arbres (fin)

Aujourd'hui nous terminons nos manipulations d'arbres.
(Si celles-ci sont finies, on considère une petite introduction aux graphes facultative, mais nous en entreprendrons la prochaine fois un traitement systématique ne présupposant rien).

1. Reprendre les exercices en cours du TD 7, les terminer, jusqu'au 5 inclus (ne pas faire le 6, sauf si vous l'aviez déjà commencé, et dans ce cas nous consulter avant).

Facultatif

Une petite introduction aux graphes Le but des exercices suivants est de représenter des graphes, puis de calculer s'il existe un chemin entre deux sommets. Ici les sommets seront simplement des entiers. (On utilise les notions de PC, qui sont aussi dans le poly. Appelez-nous si besoin pour des précisions.)
On n'utilisera pas la récursivité. On choisira judicieusement entre les boucles *for* ou *while*.

2. Représenter un graphe par sa matrice d'adjacence, codée en Java par un classe contenant en particuleir un tableau à deux dimensions (pour les tableaux, voir exemple restreint ci-dessous). Saisir et afficher un graphe (lors des tests, se limiter à trois sommets pour raccourcir la saisie).

3. Rappelons-nous ensuite qu'il y a k chemins de longueur $p = 2$ entre les sommets i et j si et seulement si le produit de la matrice d'adjacence par elle-même contient k en i, j.

   Calculer ce produit, puis, étant donné deux sommets, l'utiliser pour dire s'ils vérifient cette propriété. (On calculera le produit en entier, même si ceci n'est pas optimal, et on l'utilisera dans l'exercice suivant.)

4. Calculer si deux sommets sont connectés en utilisant la généralisation de la propriété précédente à p quelconque.

Corrigé : `http://www.lri.fr/~waller/COURS/POLYTECHNIQUE/TC/TD8`

```java
import java.io.*;

public class CodeTd8 {

    public static String Lire () {
        ... comme d'habitude ...
    }

    public static void main (String[] args) {
        int a[][] = new int[7][7];

        a[0][6] = Integer.parseInt(Lire()); // on ne remplit que cette case
                                            // autres valent zero par defaut
        for (int i=0; i<7; i++)
            System.out.println(a[i][6]);    // affichage de la 7eme colonne
    }
}
```