TP 1 : Récursivité

Objectif du TP

L'objectif de cette séance est de pratiquer la programmation récursive.

EXERCICES SUR MACHINES !

Exercice 1 Factorielle négative (20 min)

Modifier la fonction factorielle récursive du cours de telle sorte qu'elle puisse prendre en compte les nombres négatifs avec la règle suivante : $\forall$ n < 0 $(n)! = (-1)^{|n|} \times (|n|)!$.

Exercice 2 Palindrome récursif (40 min)

a) À l'aide de la classe `java.util.Scanner`, lisez une chaine de caractères au clavier. Écrivez une méthode récursive qui permet de savoir si la chaine est un palindrome. Vous n'utiliserez que les méthodes `charAt()`, `substring()` et `length()` sur la chaine de caractères.
b) Modifier la fonction récursive précédente pour ne pas tenir compte des espaces.
c) Pensez à écrire les méthodes de tests.

Exercice 3 Éponge de Menger (60 min)

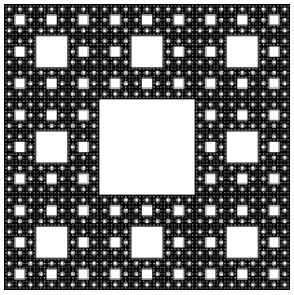
Vous trouverez ci-dessous le code source minimum pour afficher une fenêtre graphique en Java dans laquelle vous pourrez dessiner (méthode `paint`). Il est possible d'utiliser l'instruction `g.drawOval(x,y,1,1)` dans la méthode `paint` pour dessiner un point en `x, y`.

```
import java.awt.Graphics;
import javax.swing.JFrame;

public class FenetreG extends JFrame {
    public void paint(Graphics g) {

    }

    public static void main(String[] args) {
        FenetreG fenetre = new FenetreG() ;
        fenetre.setDefaultCloseOperation(EXIT_ON_CLOSE) ;
        fenetre.setExtendedState(MAXIMIZED_BOTH) ;
        fenetre.setVisible(true) ;
        fenetre.repaint();
    }
}
```



Afin de dessiner l'éponge de Menger (ci-dessus), on remarque que le grand carré est divisé en 9 petits carrés ; le carré central est blanc, mais les 8 petits carrés extérieurs représentent tous le même dessin qui n'est en fait que le grand carré réduit. Il en est de même des 8 petits carrés d'un carré extérieur et ainsi de suite. Ainsi, dessiner le grand carré, c'est dessiner 8 fois ce même carré avec un coté réduit d'un tiers. On vient d'identifier la récurrence du dessin. Par conséquent, on peut en déduire l'algorithme ci-contre.

```
void Menger(int taille) {
    Menger(taille/3) au Nord-Ouest
    Menger(taille/3) au Nord
    Menger(taille/3) au Nord-Est
    Menger(taille/3) à Est
    Menger(taille/3) à Ouest
    Menger(taille/3) au Sud-Ouest
    Menger(taille/3) au Sud
    Menger(taille/3) au Sud-Est
}
```

a) Identifier le cas d'arrêt.
b) Implémenter cet algorithme. La taille initiale de l'éponge de Menger doit être une puissance de 3 (3^5 donne un bon résultat).
c) Afin de mieux voir le tracé de l'éponge, ajoutez une temporisation après chaque appel récursif (`try`

`{Thread.sleep(nombre_de_millisecondes) ;} catch(Exception e) { } ;`).

d) Modifier l'ordre des appels récursifs (le sud avant le nord par exemple) et lancer en parallèle les deux programmes (le nord avant le sud ET le sud avant le nord) pour comparer visuellement ce que cela change.