TP 4 : Boucles et tableaux. Corrigé

Informatique Fondamentale (IF121)

10–21 octobre 2003

Exercice 1
Il n'y a pas de difficultés particulières. La boucle for sert à itérer n fois le produit par l'entier dont on souhaite calculer la puissance. Pour les puissance négatives, on remarque que $x^{-n} = \left(\frac{1}{x}\right)^n$. D'où le code suivant :

```
import fr.jussieu.script.Deug;
class Puissance{
    static double puissance (double nombre, int puissance){
        double p=1;
        if (puissance >= 0){
            for (int i=1; i <= puissance; i++){
                p = p*nombre;
            }
        }
        else{
            for (int i=1; i <= ((-1)*puissance); i++){
                p = p/nombre;
            }
        }
        return p;
    }
    public static void main (String[] args){
        Deug.print("Nombre? ");
        double nombre = Deug.readDouble();
        Deug.print("Exposant? ");
        int puiss = Deug.readInt();
        Deug.println(nombre+" ^ "+puiss+" vaut: "+puissance(nombre, puiss));
    }
}
```

Exercice 2
On a $n! = 1 \times 2 \times 3 \times \cdots \times n$. On utilise donc une boucle for et on fait le produit des valeurs successives prises par le compteur, à savoir $1, 2, \ldots, n$. D'où le code suivant :

```
import fr.jussieu.script.Deug;
class Factorielle{
    static int factorielle (int nombre){
        int fact = 1;
        for (int i=2; i <= nombre; i++){
            fact = fact*i;
        }
        return fact;
    }

    public static void main (String[] args){
        Deug.print("Nombre? ");
        int nombre = Deug.readInt();
        Deug.println(nombre+"!= "+factorielle(nombre));
    }
}
```

Exercice 3
On calcule les termes successifs de la suite (dans la variable `x` du code ci-dessous) :

```
import fr.jussieu.script.Deug;
```

```
class Suite{
    static void suite (int a, int n){
        int x = a;
        Deug.println("x(0)="+a);
        for (int i=1; i <= n; i++){
            x = 4*x*(1-x);
            Deug.println("x(" + i + ")=" + x);
        }
    }

    public static void main (String[] args){
        Deug.print("a? ");
        int a = Deug.readInt();
        Deug.print("n? ");
        int n = Deug.readInt();
        suite(a,n);
    }
}
```

Exercice 4

La seule difficulté est la gestion des nombres impairs. Le n-ième nombre impair étant $2n-1$, on a le code suivant :

```
import fr.jussieu.script.Deug;
class Serie2{
    static int sommeCarrésImpairs (int n){
        int somme = 0;
        for (int i=1; i <= n; i++){
            somme = somme + (2*i-1) * (2*i -1);
        }
        return somme;
    }

    public static void main (String[] args){
        Deug.print("n? ");
        int n = Deug.readInt();
        Deug.println("La somme des "+n+" premiers carrés impairs vaut: "+sommeCarrésImpairs(n));
    }
}
```

Exercice 5

```
import fr.jussieu.script.Deug;
class SuiteDouble{
    static double approximationDePi (int n){
        double u = 1.0;
        double v = 2.0;
        for (int i=1; i <= n; i++){
            u = (u+v) /2;
            v = Math.sqrt(u*v);
        }
        double pi = Math.sqrt(27.0) /v;
        return pi;
    }

    public static void main (String[] args){
        Deug.println("veuillez entrer un entier n d'après lequel calculer une valeur approchée de Pi");
        int n = Deug.readInt();
        Deug.println("une approximation de Pi à partir de v"+n+" est: "+approximationDePi(n));
    }
}
```

Exercice 6

Il faut penser à stocker les deux dernières valeurs calculées. Voici le code :

```
import fr.jussieu.script.Deug;
class SuiteFibonacci{
    static int fibonacci(int n){
```

```
        int f0=1;
        int f1=1;
        int tmp;
        for (int i=2; i <= n; i++){
            tmp=f1+f0;
            f0=f1;
            f1=tmp;
        }
        return f1;
    }
    public static void main (String[] args){
        Deug.print("n? ");
        int n = Deug.readInt();
        Deug.println("F"+n+" = "+fibonacci(n));
    }
}
```

Exercice 7
Voir ci dessous.

Exercice 8
Voir ci dessous.

Exercice 9
Pour la lecture du tableau il faut penser à initialiser le tableau, puis il s'agit d'une boucle for. Pour le produit scalaire on parcourt les deux tableaux en même temps. La norme est le produit scalaire du tableau par lui même. Quant à la norme sup, il faut prévoir une variable dans laquelle on stocke le plus grand élément déjà vu et l'actualiser en parcourant le tableau. Cela donne le code suivant :

```
import fr.jussieu.script.Deug;
class Tableau{
    static double[] tableau (int n){
        double[] t = new double[n];
        for (int i=0; i < n; i++){
            Deug.print("Réel n°"+(i+1)+" = ");
            t[i] = Deug.readDouble();
        }
        return t;
    }

    static double produitScalaire  (double[] x, double[] y){
        int n = x.length;
        double prod = 0.0;
        for (int i = 0; i<n; i++){
            prod = prod + x[i]*y[i];
        }
        return prod;
    }

    static double norme (double[] x){
        return produitScalaire(x,x);
    }

    static double abs (double x){
        if (x < 0){
            x = -x;
        }
        return x;
    }

    static double normeSup (double[] x){
        double max = abs(x[0]);
        int n = x.length;
        for (int i = 0; i<n; i++){
            if (max < abs(x[i])){
                max = abs(x[i]);
```

```
            }
        }
        return max;
    }

    public static void main (String[] args){
        Deug.print("Taille du tableau? ");
        int n = Deug.readInt();
        double[] t = tableau(n);
        Deug.println("Norme = " + norme(t));
        Deug.println("Norme sup = " + normeSup(t));
    }
}
```

Exercice 10
Il faut tout d'abord récupérer les tailles `n1` et `n2` des deux tableaux à concaténer. Ensuite, on crée un tableau de taille égale à la somme des tailles précédentes. Puis on parcours le premier tableau en recopiant les éléments dans le nouveau tableau. Enfin on parcours le second tableau et on recopie les éléments dans le nouveau tableau à la suite des autres. Cela donne alors :

```
import fr.jussieu.script.Deug;
class Concatenation{
    static double[] concatene ( double[] t1, double[] t2){
        int n1 = t1.length;
        int n2 = t2.length;
        double[] concatene = new double[n1+n2];
        for (int i=0; i < n1; i++){
            concatene[i] = t1[i];
        }
        for (int i = 0; i < n2; i++){
            concatene[n1+i] = t2[i];
        }
        return concatene;
    }

    public static void main(String[] args){
        Deug.println("Premier tableau:");
        Deug.print("Dimension? ");
        int n1 = Deug.readInt();
        double[] t1 = Tableau.tableau(n1);
        Deug.println("Deuxième tableau:");
        Deug.print("Dimension? ");
        int n2 = Deug.readInt();
        double[] t2 = Tableau.tableau(n2);
        double[] t = concatene(t1,t2);
        Deug.print("Tableau concaténé= ");
        for (int i=0; i < t.length; i++){
            Deug.print(t[i] + "   ");
        }
    }
}
```

Exercice 11
Il suffit de comparer chaque élément du tableau à l'élément qui le suit jusqu'à détecter éventuellement une erreur. C'est pour cela que je préfère ici une boucle `while` (à quoi bon parcourir le tableau si dès les deux premiers éléments on détecte qu'il n'est pas trié ?). Il faut faire attention également à ne pas sortir du tableau.

```
import fr.jussieu.script.Deug;
class TableauTrie{
    static boolean verif (double[] t){
        boolean b = true;
        int n = t.length;
        int i = 0;
        while ((i<n-1)&& b){
            b = b && (t[i] < t[++i]);
        }
        return b;
```

```
    }

    public static void main (String[] args){
        Deug.print("Dimension? ");
        int n = Deug.readInt();
        double[] t = Tableau.tableau(n);
        if (verif(t)){
            Deug.println("Votre tableau est trié en ordre croissant");
        }
        else{
            Deug.println("Votre tableau n'est pas trié en ordre croissant");
        }
    }
}
```

Exercice 12

Ce programme est difficile. Le premier élément du tableau fusionné est le premier du premier ou du second tableau. Le second est le second élément du tableau d'où est issu le premier élément ou le premier élément de l'autre tableau. Et ainsi de suite. Ainsi, on maintient deux compteurs `i` et `j` qui parcourent respectivement le premier et le second tableau. Ils désignent le premier élément de chaque tableau non encore mis dans le tableau fusionné. C'est donc parmi ces deux éléments qu'il faudra chercher le nouvel élément à fusionner. Voici le code qui s'en suit (ne pas hésiter à me demander de plus amples détails) :

```
 static double[]
fusion (double[] x, double[] y) {
    int n = x.length + y.length;
    double[] t = new double[n];
    int i = 0; //compteur pour le premier tableau
    int j = 0; //compteur pour le second tableau
    for (int k = 0; k < n; k++) {
        if (i == x.length) { // on a épuisé le premier tableau
            t[k] = y[j];
             j++;
        }
        else if (j == y.length) { // on a épuisé le second tableau
            t[k] = x[i];
            i++;
        }
        else if (x[i] <= y[j]) { // le prochain élément est dans le premier tab.
        t[k] = x[i];
            i++; // on avance dans le premier tableau.
        }
        else { // le prochain élément est dans le second tab.
            t[k] = y[j];
            j++; //on avance dans le second tableau.
        }
    }
    return t;
}
```

Exercice 13

L'idée est de créer un tableau qui va contenir la répartition par note (la case `i` indiquera le nombre d'élèves ayant la note i) et de parcourir le tableau des notes en mettant à jour celui de la répartition. On a alors :

```
class Histogramme{
static int[] tableauNotes (int n){
        int[] t = new int[n];
        for (int i=0; i < n; i++){
            Deug.print("Note "+(i+1)+" = ");
            t[i] = Deug.readInt();
        }
        return t;
    }


    static int[] tableauStats (int[] notes){
        int[] t = new int[11];
```

```
        int note;
        for (int i=0; i < notes.length ; i++){
            note = notes[i];
            t[note] ++;
        }
        return t;
    }

    public static void main(String[] args){
        Deug.print("Veuillez entrer le nombre d'étudiants");
        int n = Deug.readInt();
        int[] notes = tableauNotes(n);
        int[] stats = tableauStats(notes);
        for (int i=0; i<=10; i++){
            Deug.println("Il y a eu "+stats[i]+" étudiant(s) ayant obtenu la note "+i);
        }
        Deug.readInt();
    }
}
```