

IF1 : interrogation, correction

Groupe M3

Le 18 décembre 2008

Exercice 1

1. La valeur de `b` est toujours `true`, tandis que `c` dit si $x < y$. Ainsi, le `if(b)` est inutile et on exécute le `if(c)` si $x < y$. Dans ce cas, on renvoie $-x$ si $x < -x$, c'est-à-dire si $x < 0$, et on renvoie x sinon : en d'autres termes, on renvoie la valeur absolue de x . Si en revanche `c` vaut `false` (c'est-à-dire $y \leq x$), alors on renvoie la valeur absolue de y (puisque $y^3 < 0$ est équivalent à $y < 0$).

Ainsi, la fonction `f` renvoie la valeur absolue du plus petit de ses arguments.

2. On pourrait donc écrire `f` de cette façon :

```
public static int f(int x, int y) {
    int a = y;
    if (x < y)
        a = x;    // a vaut min(x,y)
    if (a < 0)
        return (-a);
    else
        return a;
}
```

Exercice 2

1.

```
public static int[] minmax(int[][] t) {
    int[] m = new int[2];
    m[0] = t[0][0]; // min
    m[1] = m;      // max
    for (int i=0; i < t.length; i++) {
        for (int j=0; j < t[i].length; j++) {
            if (t[i][j] < m[0])
                m[0] = t[i][j];
            if (t[i][j] > m[1])
                m[1] = t[i][j];
        }
    }
    return m;
}
```
2.

```
public static void valabs(int[][] t) {
    for (int i=0; i < t.length; i++) {
```

```

        for (int j=0; j < t[i].length; j++) {
            if (t[i][j] < 0)
                t[i][j] = -t[i][j];
        }
    }
}

3. // fonction auxiliaire : nombre de fois que t[i] apparait dans t[i..n]
public static int multiplicite(int[] t, int i) {
    int m = 0;
    for (int j = i; j < t.length; j++) {
        if (t[j] == t[i])
            m++;
    }
    return m;
}

public static int frequent(int[] t) {
    int f = t[0]; // élément le plus fréquent
    int m = multiplicite(t, 0); // nombre de fois qu'il apparaît dans t
    for (int i = 1; i < t.length; i++) {
        if (t[i] != f) { // inutile de recompter si t[i] == f
            int a = multiplicite(t, i);
            if (a > m) {
                m = a;
                f = t[i];
            }
        }
    }
    return f;
}

```

Exercice 3

```

1. public static int l(int n) {
    int x = 0;
    while (n > 1) {
        n = n/2;
        x++;
    }
    return x;
}

2. public static int letoile(int n) {
    int x = 0;
    while (n > 0) {
        n = l(n);
        x++;
    }
    return x;
}

```

Exercice 4

```
import fr.jussieu.script.Deug;
public class polynomes{

public static int[] demander() {
    int d = Deug.readInt();
    int[] p = new int[d+1];
    for (int i = 0; i <= d; i++){
        Deug.println("Entrer le coefficient de degré "+i);
        p[i] = Deug.readInt();
    }
    return p;
}

public static void afficher(int[] p) {
    boolean premier = true; // savoir si c'est le premier coefficient qu'on affiche
    for (int i = p.length-1; i >= 0; i--) {
        if (p[i] != 0) {
            if ((!premier) && (p[i]>0))
                Deug.print('+');
            premier = false;
            if (i > 1)
                Deug.print(p[i]+"x^"+i);
            if (i == 1)
                Deug.print(p[i]+'x');
            if (i == 0)
                Deug.println(p[i]);
        }
    }
}

public static int evaluer(int[] p, int a) {
    int s = 0;
    for (int i = p.length-1; i >= 0; i--) {
        s = s*a + p[i];
    }
    return s;
}

public static int[] somme(int[] p, int[] q) {
    int min = p.length; // taille du plus petit tableau
    int max = min;      // taille du plus grand tableau
    boolean b = true;   // repère le plus grand tableau (true pour p)
    if (q.length > max) {
        max = q.length;
        b = false;
    }
    else
```

```

        min = q.length;
    int[] r = new int[max];
    for (int i = 0; i < min; i++) { // pour ces coef. on fait la somme
        r[i] = p[i]+q[i];
    }
    for (int i = min; i < max; i++) { // pour ces coef. on prend soit p soit q
        if (b) // p est le plus grand tableau
            r[i] = p[i];
        else
            r[i] = q[i];
    }
    return r;
}

public static int[] produit(int[] p, int[] q) {
    int[] r = new int[p.length + q.length - 1];
    for (int i = 0; i < r.length; i++) { // calcul du i-ème coef.
        r[i] = 0;
        for (int j = 0; j < p.length; j++) {
            if ((i-j >= 0) && (i-j < q.length))
                r[i] += p[j] * q[i-j];
        }
    }
    return r;
}

public static void main(String[] s) {
    int[] p, q;
    int a;
    p = demander();
    q = demander();
    a = Deug.readInt();
    afficher(p);
    afficher(q);
    Deug.println(evaluer(p, a));
    Deug.println(evaluer(q, a));
    afficher(somme(p,q));
    afficher(produit(p,q));
}
}

```