

PROGRAMMATION ORIENTÉE OBJETS

EXAMEN 2ÈME ANNÉE

Cette épreuve serait à rendre sous forme électronique *et* papier. Sur papier, commentez vos œuvres. Sur ordinateur, ne mettez pas votre nom dans les fichiers. Vous placerez vos réponses dans un répertoire dont le nom est votre numéro de copie.

Une réponse sans commentaires/documentation ou sans code vaut 0. Une fonction qui ne marche pas *ou qui ne donne pas le même résultat que l'exemple*, même si elle compile, vaut 0 si vous n'expliquez pas dans quelles circonstances elle ne marche pas ou pourquoi. De même, précisez vos choix de conception. Si vos fonctions sont récursives terminales, mentionnez-le.

Les seuls documents autorisés sont vos cours et TDs et le *manuel* de Java. Vous avez 2h.

Fonctions entières

Pour ce qui suit, nous utiliserons l'interface `FonctionEntiere` pour représenter les fonctions :

```
package fr.univ_orleans.mi.exam.fun;

public interface FonctionEntiere
{
    /**
     * Calcule le résultat de la fonction avec un argument n.
     */
    public int appliquer(int n);
}
```

Ainsi, la fonction $f: \mathbf{Z} \rightarrow \mathbf{Z}$ définie par $\forall n \in \mathbf{Z}, f(n) = n + 1$ s'écrira

```
package fr.univ_orleans.mi.exam.fun;

public class FonctionF implements FonctionEntiere
{
    public int appliquer(int n)
    {
        return n+1;
    }
}
```

Exercice 1. (environ 2 points) Écrire une classe implantant `FonctionEntiere` construite à partir d'une `FonctionEntiere` représentant la fonction f et dont la méthode `appliquer` calcule associée à n la somme $\sum_{i=0}^{i=n} f(i)$.

Exemple

```
package fr.univ_orleans.mi.exam.fun;

//...
FonctionEntiere f      = new FonctionF();
FonctionEntiere somme = new Somme(f)  ;
System.out.println(somme.appliquer(5)); //affiche 21
//...
```

Exercice 2. (environ 5 points) Si $(u_n)_{n \in \mathbf{N}}$ et $(v_n)_{n \in \mathbf{N}}$ sont deux suites, le produit de convolution de u et v , noté $u \otimes v$, est la suite définie par $\forall n \in \mathbf{N}, (u \otimes v)_n = \sum_{i=0}^{i=n} u(i) \cdot v(n-i)$.

Écrire une fonction qui, à partir de deux fonctions u et v et d'un entier n , calcule $(u \otimes v)_n$.
Exemple

```
package fr.univ_orleans.mi.exam.fun;

public class FonctionV implements FonctionEntiere
{
    public int appliquer(int n)
    {
        return 2*n;
    }
}

//...
FonctionEntiere u    = new FonctionF()    ;
FonctionEntiere v    = new FonctionV()    ;
FonctionEntiere convo = new Convolution(u,v) ;
System.out.println(convo.appliquer(5))    ;    //affiche 70
//...
```

Quelle est la complexité de cette fonction, en nombre de multiplications $u(i) \cdot v(n - i)$ et en fonction de n ?

Tableaux

Exercice 3. (environ 3 points) Écrire une fonction qui enlève d'un tableau un élément sur 2
Exemple :

```
//...
int tableau[] = new tableau[] {1,2,3,4,5,6,7,8};
int tableau2[] = enlever(tableau);
for(int i = 0; i < tableau2.length; ++i)
{
    System.out.println(tableau2[i]);
} //affiche 1 3 5 7
```

Exercice 4. (environ 9 points) Écrire une fonction qui mélange *aléatoirement* un tableau. Vous pourrez par exemple utiliser les méthodes `nextInt` (qui tire au hasard un entier) ou `nextBoolean` (qui tire à pile ou face) de la classe `Boolean`.

Exemple :

```
//...
int tableau[] = new tableau[] {1,2,3,4,5,6,7,8};
int tableau2[] = melanger(tableau);
for(int i = 0; i < tableau2.length; ++i)
{
    System.out.println(tableau2[i]);
} //affiche 3 7 1 2 6 4 5 8
```

Exercice 5. (environ 6 points) Écrire une fonction qui transforme un tableau en son symétrique, sans modifier le tableau original.

Exemple :

```
//...
int tableau[] = new tableau[] {1,2,3,4,5,6,7,8};
```

```

int tableau2[] = retourner(tableau);
for(int i = 0; i < tableau.length; ++i)
{
    System.out.println(tableau[i]);
} //affiche 1 2 3 4 5 6 7 8
for(int i = 0; i < tableau2.length; ++i)
{
    System.out.println(tableau2[i]);
} //affiche 8 7 6 5 4 3 2 1

```

Exercice 6. Écrire une fonction qui transforme un tableau en son symétrique – en modifiant directement le tableau original.

```

//...
int tableau[] = new tableau[] {1,2,3,4,5,6,7,8};
retournerSurPlace(tableau);
for(int i = 0; i < tableau.length; ++i)
{
    System.out.println(tableau[i]);
} //affiche 8 7 6 5 4 3 2 1

```

Exercice 7. (environ 5 points) Écrire une fonction qui détermine si un tableau est un palindrome, c'est-à-dire si il est égal à son symétrique.

Exemple :

```

//...
int tableau[] = new tableau[] {1,2,3,4,5,6,7,8};
int tableau2[] = new tableau[] {1,2,3,4,5,4,3,2,1};
System.out.println(palindrome(tableau)); //affiche false
System.out.println(palindrome(tableau2)); //affiche true

```

Chaînes de caractères

Exercice 8. (environ 10 points) Écrire une fonction qui transforme un titre entièrement en majuscules en un titre à l'anglo-saxonne, c'est-à-dire où la première lettre de chaque mot est en majuscules.

Exemple :

```

System.out.println(a_l_anglo_saxonne("HARRY POTTER 7: HARRY POTTER AND THE
DEATHLY HALLOWS"));
//Affiche "Harry Potter 7: Harry Potter And The Deathly Hallows"

```

Théorie

Exercice 9. (environ 12 points) Est-il possible d'écrire une fonction `boolean compare(Object a, Object b)` qui renverra `true` si et seulement si `a` et `b` sont égaux, c'est-à-dire s'ils sont de la même classe et la valeur de chacun des champs de `a` est égale à la valeur du champ correspondant de `b` ?

Si oui, détaillez comment va marcher cette fonction. Sinon, expliquez pourquoi.

—

Bon courage.