

Nom :

Prénom :

Programmation Orientée Objets - Semestre d'automne 2024
Stéphane Malandain / Yassin Rekik

Test n°1
40 minutes

1. Typage (4 pts)

Indiquez, pour chaque affectation dans les lignes ci-dessous si l'affectation est possible, et le cas échéant, la valeur exacte qui sera affectée à la variable.

```
1 double a1 = 3;  
2 int a2 = 2.4;  
3 short a3 = (int) 24.7;  
4 int a4 = 1.2;  
5 float a5 = 4.44;  
6 double a6 = (double) a3;  
7 double a7 = (int) a6;  
8 float a8 = 6.4F;  
9 float a9 = a6 / a7;  
10 int a10 = 30 / 7;  
11
```

	Possible : oui – non	Valeur, si possible
1	Oui	3.0
2	Non	
3	Oui	24
4	Non	
5	Non	
6	Oui	24.0
7	Oui	24.0
8	Oui	6.4
9	Non	
10	Oui	4

2. Paramètres de Méthodes (4 pts)

On dispose des méthodes et des déclarations suivantes :

```
1 public void m1(short s, double d) {}  
2 public void m2 (char c) {}  
4 byte b;  
5 short s;  
6 float f;  
7 double d;  
8 char c;
```

Spécifiez si les appels suivants sont corrects ou non tout en justifiant vos réponses :

	Correct : oui / non ?	Pourquoi ?
22 m1(2,f)	non	incompatible types: possible lossy conversion from int to short
23 m1(f,b)	non	incompatible types: possible lossy conversion from float to short
24 m1(f,c)	non	incompatible types: possible lossy conversion from float to short
25 m1(f,i)	non	incompatible types: possible lossy conversion from float to short
26 m1(i+5,b)	non	incompatible types: possible lossy conversion from float to short
27 m2(c)	non	incompatible types: possible lossy conversion from int to short
28 m2(64)	oui	
29 m2(6.0)	non	incompatible types: possible lossy conversion from int to char
30 m2(b)	Non	incompatible types: possible lossy conversion from double to char
31 m2('a')	Non	incompatible types: possible lossy conversion from byte to char
	oui	

Nom :

Prénom :

3. Tableaux (4 pts)

Vous devez implémenter une méthode statique prenant en paramètre un tableau à 3 dimensions, pas nécessairement égales, et initialise toutes les cellules à la valeur d'un compteur que vous incrémentez de 1 à chaque nouvelle cellule.

Réponse :

```
1  public class Exo_tab {
2
3      public static void InitTab(int[][][] t) {
4          int cpt = 0;
5          for(int i = 0 ; i < t.length ; i++) {
6              for(int j = 0 ; j < t[i].length ; j++) {
7                  for(int k = 0 ; k < t[i][j].length ; k++)
8                      t[i][j][k] = cpt ++;
9              }
10         }
11     }
12
13     public static void main(String[] args) {
14         int[][][] tab = new int[5][2][3];
15         InitTab(tab);
16         for(int i = 0 ; i < tab.length ; i++) {
17             for(int j = 0 ; j < tab[i].length ; j++) {
18                 for(int k = 0 ; k < tab[i][j].length ; k++)
19                     System.out.print(tab[i][j][k]+" ");
20                 System.out.println();
21             }
22             System.out.println();
23         }
24     }
```

Nom :

Prénom :

4. String (8 pts)

En utilisant ces 2 méthodes offertes par la classe String :

String s = "ceci est un test"

public char charAt(int index)

Retourne le caractère à la position index de la String

s.charAt(1) retourne 'e'

public int length()

Retourne la taille du String

s.length() retourne 16

On donne l'algorithme de cryptage élémentaire (et très inefficace) suivant :

Données :

- une chaîne de caractères t : le texte à coder ;
- une chaîne de caractères c : la clé (de longueur strictement inférieure à celle de t).

Résultat : une chaîne-de-caractères cryptée.

1. créer une chaîne vide s.

2. pour chaque caractère x de t (dans l'ordre) :

- (a) soit j la position de x dans t. Calculer i comme le reste de la division de j par la longueur de c.
- (b) ajouter au code Unicode de x la valeur de i afin d'obtenir un nouveau caractère.
- (c) ajouter à la chaîne s ce caractère obtenu.

3. Résultat : la chaîne s.

Ci-après, le code du programme principal. Vous devez implémenter la méthode **Decrypte**

Nom :

Prénom :

```
1 public class CryptageElem
2 {
3     public static String Encrypte( String source, String cle) {
4         // à coder
5     }
6
7     public static String Decrypte( String source, String cle) {
8         // ne pas tenir compte
9     }
10    public static void main(String[] args)
11    {
12        String t = "mon texte a coder";
13        String c = "maCle";
14        String s = "";
15        String result = "";
16
17        s = Encrypte(t,c);
18        System.out.print(" La chaine t=* "+t+ " * donne // "+s+" // encryptée.");
19        result = Decrypte(s,c);
20        System.out.print(" La chaine s=* "+s+ " * donne // "+result+" // décryptée.");
21    }
22 }
```

Implémentez, dans la page suivante, la méthode Decrypte (indication ~ 15 lignes approx.)

```
1 public class CryptageElem {
2
3 > public static String Encrypte( String source, String cle) { ...
19
20 public static String Decrypte( String source, String cle) {
21     String result="";
22     int tailleCle = cle.length();
23     char c;
24     int unicode;
25     int reste;
26
27     for(int j = 0 ; j < source.length() ; j++) {
28         reste = j%tailleCle;
29         c = source.charAt(j);
30         unicode = c - reste;
31         c = (char) unicode;
32         result = result + c;
33     }
34     return result;
35 }
Run | Debug
36 public static void main(String[] args)
37 {
38
39     String t = "mon texte a coder";
40     String c = "maCle";
41     String s = "";
42     String result = "";
43
44     s = Encrypte(t,c);
45     System.out.print(" La chaine t=* "+t+ " * donne // "+s+" // encryptée.");
46     result = Decrypte(s,c);
47     System.out.print(" La chaine s=* "+s+ " * donne // "+result+" // décryptée.");
48 }
49 }
```