



+10/1/6+

- 0 0
 1 1
 2 2
 3 3
 4 4
 5 5
 6 6
 7 7
 8 8
 9 9

← codez votre numéro d'étudiant ci-contre et inscrivez votre nom et prénom ci-dessous.

Nom et prénom :

LUU Nguyen Phuoc Loc

Email professionnel:

Nguyen-Phuoc-Loc.Luu@etu.univ-grenoble-alpes.fr

Dans le questionnaire suivant, pour chaque question, il vous est proposé plusieurs réponses. Une réponse possible pour une question est :

- soit une **proposition** (par exemple "vrai" ou "bleu").
- soit une *assertion sur les propositions* ("aucune n'est vraie", "toutes sont vraies", etc.) Les assertions sur les propositions seront en *italique*.

Pour chaque question, il existe une et une seule bonne réponse ; il s'agit soit d'une proposition, soit d'une assertion sur les propositions.

Attention, les assertions sur les propositions, par exemple *Toutes les propositions sont justes* ne concernent que les propositions, les autres assertions (du type *Aucune des propositions n'est juste*) ne sont pas concernées (sinon, ça n'a pas de sens...).

L'assertion *Données insuffisantes* signifie qu'étant donné les seuls éléments de la question, on ne peut pas déterminer pour chacune des propositions si elles sont vraies ou fausses. Par exemple à la question "Quel âge avait Kennedy ?", avec pour propositions 25, 43 et 46, il manque des éléments pour répondre : Quel Kennedy (John F. ? l'un de ses frères ? son père ?) ? et à quel moment (lorsqu'il est arrivé au pouvoir ? lorsqu'il est décédé ?) ? Bref, l'énoncé de la question ne donne pas assez d'éléments pour répondre.

Des points négatifs seront affectés en cas de réponse fausse ; une non-réponse, quant à elle, n'entraîne ni point positif, ni point négatif.

Le questionnaire suivant est corrigé par une machine. Veuillez noircir les cases des bonnes réponses. Barrer une réponse fausse ne sert à rien. Si vous souhaitez corriger l'une de vos réponses, utiliser du blanc.

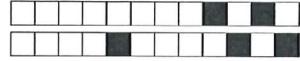
Diagrammes UML de classes

Question 1 En Java, le type de retour d'un constructeur est :

- 1/1 toujours String *tions n'est correcte.* le type de la Classe
 Aucune des proposi- toujours int void

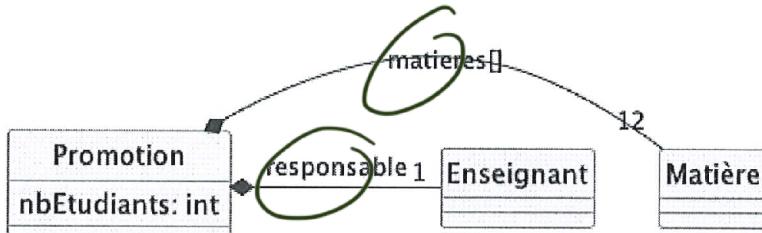
Question 2 À quoi reconnaît-on une méthode dans un diagramme UML de classes ?

- 0/1 Aucune des propositions n'est correcte.
 À son nom, qui commence par celui de la classe
 Au type de retour Au nombre de paramètres
 Aux parenthèses à la fin de son nom



+10/2/5+

Question 3 On considère le diagramme de classes suivant :



Écrire la classe **Promotion** en Java.

A B C D E Réservé au correcteur

0/1

```
class Promotion {
    int nbEtudiants
}
```

Types Prédéfinis en Java

Question 4 En Java, le type **String** est un type primitif.

-0.25/1 vrai

faux

Question 5 Pour le type **boolean**, l'opération **||** (ou bien) est prioritaire sur l'opération **&&** (et puis). C'est-à-dire que lorsque l'on a **bool res = a && b || c**, on calcule d'abord **d = b || c** et puis **a && d**.

1/1 faux

vrai

Question 6 Pour le type **boolean**, lors que l'on a l'expression **boolean res = a && b**, si **a** est faux (*false*), alors on n'évalue pas **b**

1/1 vrai

faux



Question 7 On considère le programme suivant:

```
int a = 5;  
int b = 2;  
int c = a / b;  
System.out.println("c = " + c);
```

Ce programme écrit :

- 1/1 c = 2.5 c = 2 c = 2.0

Question 8 On considère le programme suivant :

```
System.out.println(3 + 5);
```

À l'écran est écrit :

- 0.25/1 3 + 5 35 8

Question 9 On peut convertir un int en double

- 0.25/1 de manière explicite (il faut utiliser le cast) pas du tout (on ne peut pas le faire) de manière implicite (on n'a rien à faire)

Question 10 On peut convertir un boolean en char

- 0/1 pas du tout (on ne peut pas le faire) de manière explicite (il faut utiliser le cast) de manière implicite (on n'a rien à faire)

Les Exceptions

Question 11 En Java, les exceptions sont :

- 0/1 des messages des classes statiques
 des types primitifs des objets
 Aucune des propositions n'est correcte.

Question 12 Lorsqu'on a un message d'erreur contenant une exception en Java, il s'agit d'une erreur

- 1/1 de compilation d'exécution

Question 13 Que se passe-t-il si une *checked exception* n'est ni déclarée, ni attrapée dans une méthode susceptible de lancer ce type d'exception ? Exemple :

```
1 void f() {  
2     // code ...  
3     throw new Exception("Attention, rien ne va plus !");  
4     // code ...  
5 }
```

- 0.25/1 Rien, il n'est pas obligatoire de faire quoi que ce soit pour ce type d'exception. Il y a une erreur à l'exécution (le programme plante).
 Il y a une erreur à la compilation.

Question 14 L'exception reçue en cas de division par zéro est :

 A E Réserve au correcteur

Anithmétique Exception



+10/4/3+

Question 15 L'exception reçue lorsqu'un indice sort d'un tableau est :

A E Réservé au correcteur

0/1

BubbleSortException ?? D'où sort cette exception?

Question 16 L'exception reçue lorsque l'on essaie de lire un attribut ou appliquer une méthode à une instance qui n'existe pas est :

A E Réservé au correcteur

1/1

NullPointerException

On considère la classe Pangolin suivante :

```
1 class Pangolin {  
2     String nom;  
3     int nbEcailles;  
4     Pangolin(String nom, int nbEcailles) throws Exception{  
5         this.nom = nom;  
6         setNbEcailles(nbEcailles);  
7     }  
8     void setNbEcailles(int nbEcailles) throws Exception {  
9         if (nbEcailles < 0) {  
10             throw new Exception("un nombre d'écaillles est forcément positif !");  
11         }  
12         this.nbEcailles = nbEcailles;  
13     }  
14 }
```



Question 17 Complétez le code de la classe TestPangolin ci-dessous de sorte que :

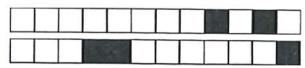
- Un Pangolin soit créé avec le nom et le nombre d'écailles entrés par l'utilisateur
- L'exception envoyée dans la classe Pangolin soit gérée
- Quoi qu'il arrive, l'utilisateur lit le message "Merci d'avoir participé" à la fin du programme (même s'il n'a pas entré une valeur correcte pour le nombre d'écailles).

Note : on ne demande pas de gérer les éventuelles exceptions générées par les méthodes de Scanner.

A B C D E Réservé au correcteur

0/4

```
1 import java.util.Scanner;
2
3 public class TestPangolin {
4     public static void main(String[] args) {
5         String nom;
6         int nbEcailles;
7
8         // Un scanner va nous permettre de lire des paramètres entrés sur la console par l'
9         // 'utilisateur.
10        Scanner scanner = new Scanner ( System . in ) ;
11        System.out.println("Veuillez entrer le nom du pangolin");
12        nom = scanner.next();
13        System.out.println("Veuillez entrer le nombre d'écailles");
14        nbEcailles = scanner.nextInt();
15
16        -----
17
18        Pangolin maturin = new Pangolin(nom, nbEcailles);
19        setNbEcailles(nbEcailles);
20
21        -----
22
23
24        System.out.println(" Merci d'avoir participé " );
25
26    }
27
28 }
```



+10/6/1+