

JIPST-CNAM  
Programmation JAVA  
NFA 001  
Avril 2016

Avec document  
Durée : **2 h30**  
Enseignant : LAFORGUE Jacques

2<sup>ème</sup> Session NFA 031

## CORRECTION

**L'examen se déroule en deux parties.** Une première partie de 1h15mn, sans document, consacrée à des questions de cours, et une deuxième partie, avec document, consacrée à la réalisation de programmes Java.

Au bout de 1h15mn, les copies de la première partie seront ramassées avant de commencer la deuxième partie.

Pour la première partie, vous devez rendre le QCM rempli et les réponses aux questions libres écrites sur des copies vierges.

Pour la deuxième partie, vous écrivez vos programmes sur des copies vierges. Vous devez écrire le code en Java.

---

### 1<sup>ère</sup> PARTIE : COURS (sans document)

---

#### 1. QCM (35 points)

##### Mode d'emploi :

Ce sujet est un QCM dont les questions sont de 3 natures :

- **les questions à 2 propositions**: dans ce cas une seule des 2 propositions est bonne.
  - +1 pour la réponse bonne
  - -1 pour la réponse fausse
- **les questions à 3 propositions** dont 1 seule proposition est bonne
  - + 1 pour la réponse bonne
  - -1/2 pour chaque réponse fausse
- **les questions à 3 propositions** dont 1 seule proposition est fausse
  - + 1/2 pour chaque réponse bonne
  - -1 pour la réponse fausse

Il s'agit de faire une croix dans les cases de droite en face des propositions.

On peut remarquer que cocher toutes les propositions d'une question revient à ne rien cocher du tout (égal à 0).

Si vous devez raturer une croix, faites-le correctement afin qu'il n'y ait aucune ambiguïté.

N'oubliez pas d'inscrire en en-tête du QCM, votre nom et prénom.

Vous avez droit à **4 points** négatifs sans pénalité.

NOM:	PRENOM:
------	---------

Dans un langage orienté objet, un objet est :		Q 1.
1	une structure de données contenant des valeurs qui répond à un ensemble de « messages » (ou « méthodes »)	X
2	une zone mémoire du langage, allouée lors de la création de l'objet contenant les caractéristiques de l'objet	X
3	une vision virtuelle d'un paquet d'information qui est extérieur au programme	

Dans la programmation orientée objet, un objet est :		Q 2.
1	une instance d'une classe créée par exemple dans le programme principal	X
2	un processus informatique qui s'exécute indépendamment du programme principal	

Le langage JAVA est <b>portable</b> sur la plupart des plateformes (windows,unix,linux,...)		Q 3.
1	OUI	X
2	NON	

Dans un langage orienté objet, un principe fort est que les attributs (ou données) non statiques sont alloués dans une instance d'une classe appelée un objet. Ainsi,		Q 4.
1	les données d'une instance sont toujours directement (sans passer par une méthode) modifiées par toutes les autres instances du programme	
2	Une instance peut "protéger" ses données en ne les rendant pas directement accessibles par les autres instances du programme	X
3	les données du programme sont réparties dans toutes les instances du programme	X

Soit le fichier suivant C:\CodeJava\exercices\fr\cnam\util\Terminal.java. Le fichier Terminal.java contient en 1 <sup>ère</sup> ligne : <b>package exercices.fr.cnam.util;</b> Le répertoire C:\bin est vide. Dans C: on réalise la commande suivante : <b>javac -d C:\bin C:\CodeJava\exercices\fr\cnam\util\Terminal.java</b> On obtient la création du fichier suivant : C:\bin\exercices\fr\cnam\util\Terminal.class		Q 5.
1	OUI	X
2	NON	

Les fichiers .class générés par la compilation javac sont des fichiers intermédiaires qui contiennent un programme exprimé dans un autre langage (P-CODE) et qui est ensuite :		Q 6.
1	traduit en fichier binaire directement exécuté par le microprocesseur	
2	interprété par un interpréteur de code P-CODE	X

En Java, l'exécution d'un programme peut se faire d'autant de façons différentes qu'il existe de méthode main dans les classes du programme		Q 7.
1	OUI	X
2	NON	

La signature d'une méthode <b>main</b> de la class <b>C1</b> permettant l'exécution d'un programme JAVA est de la forme <b>public static void main(String... args)</b> . Cette méthode est static parce que la commande <b>java C1 a1 a2</b> consiste à demander à la JVM Java d'exécuter l'instruction JAVA suivante : <b>C1.main(a1,a2)</b>		Q 8.
1	OUI	X
2	NON	

La classe Exemple.java appartient au package "fr.cnam.prog". Soit l'arborescence de répertoires suivante : <pre> Exemple00/   bin/   fr/     cnam/       prog/         Exemple.java </pre> La commande de compilation est exécutée dans le répertoire Exemple00. Les fichiers compilés sont créés dans le répertoire bin. Cette commande peut être :		Q 9.
1	<code>javac -d bin Exemple.java</code>	
2	<code>javac -d bin fr/cnam/prog/Exemple.java</code>	X
3	<code>javac fr/cnam/prog/Exemple.java</code>	

Le paramètre <b>-classpath</b> ou la variable d'environnement <b>CLASSPATH</b> est utilisée pour désigner une liste de plusieurs path d'accès à des répertoires. Chacun de ces répertoires contient les fichiers .class ou les packages utilisés dans la compilation ou dans l'exécution d'un programme JAVA.		Q 10.
1	OUI	X
2	NON	

Soit une classe contenant les méthodes mstat1 et m2. mstat1 est une méthode statique et m2 n'est pas une méthode statique :		Q 11.
1	la méthode mstat1 peut utiliser les attributs statiques de la classe	X
2	la méthode m2 peut utiliser les attributs statiques de la classe	X
3	la méthode mstat1 peut utiliser les attributs non statiques de la classe	

Soit le code JAVA suivant : <pre> public class C {     private int attribut;     public C(){attribut=10;}     public static void main(String[] args)     {         C objet = new C();         objet.attribut = Integer.parseInt(args[0]);     } } </pre> Ce code est correct		Q 12.
1	OUI	
2	NON	X

Les méthodes public de la classe A peuvent utiliser directement les attributs privés de la classe A		Q 13.
1	OUI	X
2	NON	

Un attribut déclaré en protected est un attribut qui est protégé en écriture. Il est une constante du programme.		Q 14.
1	OUI	
2	NON	X

Soit la classe C1 dont tous les attributs sont <u>privés et statiques</u> . Soit la classe C2 appartenant à un autre package que C1. Dans ce cas, les méthodes de C2 peuvent accéder aux attributs de C1		Q 15.
1	OUI	
2	NON	X

La caractéristique "private" d'un attribut rend inaccessible l'attribut par toutes les autres classes		Q 16.
1	OUI	X
2	NON	

En JAVA, il est possible de définir deux méthodes de même nom dans la même classe		Q 17.
1	OUI	X
2	NON	

Le constructeur dans une classe d'objet permet :		Q 18.
1	de construire la classe de définition de l'objet (attributs et méthodes)	
2	d'initialiser les attributs de l'objet avec des valeurs bien particulières	X
3	d'initialiser les attributs en fonction des paramètres du constructeur	X

Soit la classe suivante :		Q 19.
<pre> public class C1{     private int x;     public C1(int x){ this.x = x; } } </pre>		
L'instruction suivante : C1 c1 = new C1(); est valide et la valeur de x est la valeur par défaut de Java 0		
1	OUI	
2	NON	X

Le code suivant est un exemple correct d'un constructeur :		Q 20.
<pre> public class Individu {     private String nom;     private String prenom;     private int age;      public Individu Individu(String nom, String prenom, int age)     {         this.nom=nom;  this.prenom=prenom;  this.age = age;         return this;     } } </pre>		
1	OUI	
2	NON	X

Quand une classe définit un constructeur avec des paramètres alors le constructeur par défaut (sans paramètre) n'est plus accessible		Q 21.
1	OUI	X
2	NON	

En JAVA, une classe peut contenir plusieurs constructeurs		Q 22.
1	OUI	X
2	NON	

Soit le code JAVA suivant		Q 23.
<pre> public class Constructeur {     public static void main(String args[])     {         Exemple1 ex = new Exemple1();         ex.tab[0] = 22;     } }  class Exemple1 {     public int[] tab;      public void Exemple1()     {         tab = new int[5];     } } </pre>		
Ce code s'exécute correctement :		
1	OUI	X
2	NON	

Soit la classe suivante :		Q 24.
<pre> public class Individu {     private String nom;     private String prenom;     private int age;      public Individu(String nom, String prenom, int age)     {         this.nom=nom;  this.prenom=prenom;  this.age = age;     }      public Individu(String nom, String prenom)     {         this.nom=nom;  this.prenom=prenom;  this.age = 0;     } } </pre>		
On peut écrire les codes suivants :		
1	Individu ind = new Individu("LAFONT","Pierre",45);	X
2	Individu ind = new Individu();	
3	Individu ind = new Individu("LAFONT","Pierre");	X

Soit le code JAVA suivant :		Q 25.
<pre> Livre l1 = new Livre(); l1.nom = "UN"; Livre l2 = l1 ; l2.nom = "DEUX"; System.out.println(l1.nom); </pre>		
Ce code affiche :		
1	UN	
2	DEUX	X

En Java, il est possible de modifier le contenu d'un objet passé en paramètre d'une méthode		Q 26.
1	OUI	X
2	NON	

Soit le code suivant :		Q 27.
<pre> public class Test2 {     public static void main(String a_args[])     {         int v = 10;         proc(v);         Terminal.ecrireIntln(v);     }      static void proc(int p)     {         p = 100;     } } </pre>		
Le résultat est :		
1	10	X
2	100	

Soit le code suivant :		Q 28.
<pre> int v=4; boolean premier=true; for(int k=2;k&lt;v;k=k+1)     if (v%k == 0) premier = false;     else premier = true; if (premier) System.out.println("PREMIER"); else System.out.println("NON PREMIER"); </pre>		
Ce code :		
2	affiche "NON PREMIER"	
3	affiche "PREMIER"	X

La boucle <b>for</b> dite "énumérative" permet :		Q 29.
1	d'incrémenter une valeur scalaire de type enum et de réaliser un traitement à chaque valeur	
2	de parcourir tous les éléments d'une instance de ArrayList	X
3	de parcourir tous les éléments d'un tableau java ( tab[] )	X

Soit le code suivant :		Q 30.
<pre>int i=0; int[] tab = new int[3]; for(int v : tab) {     tab[i] = v*10;     i = i+1; } for(int j=0;j&lt;tab.length;j++)     System.out.print(tab[j]+" ");</pre>		
Ce code affiche :		
1	0 10 100	
2	0 0 0	X

En JAVA, le type de retour d'une méthode est soit void, soit un type primitif, soit la référence d'un objet		Q 31.
1	OUI	X
2	NON	

En JAVA, un tableau ( tab[] )		Q 32.
1	peut contenir des éléments de type primitif	X
2	peut contenir des références d'objet JAVA	X
3	ne peut pas contenir de références d'objet JAVA	

Soit le code suivant :		Q 33.
<pre>String str1[] = new StringTokenizer("AA;BB;CC",";"); for(String s: str1)     System.out.println(s);</pre>		
Ce code ::		
1	affiche : AA BB CC	
2	n'affiche rien	
3	ne se compile pas correctement	X

Dans la classe String la méthode d'objet <i>void set(int i, char c)</i> change le i-ième caractère d'une chaîne de caractère avec la valeur c.		Q 34.
1	OUI	
2	NON	X

En JAVA, la déclaration d'un tableau comme suit :		Q 35.
<pre>Individu[] tab_ind = new Individu[10]</pre>		
contient :		
1	10 éléments dont les valeurs sont égales au résultat de l'exécution du constructeur <i>Individu()</i>	
2	10 éléments dont les valeurs sont toutes à <i>null</i>	X

(Tournez la page)

## 2. Questions libres (15 points)

Chaque question est notée sur 5 points.

Vous répondez à ces questions sur une copie vierge en mettant bien le numéro de la question, sans oublier votre nom et prénom.

### Q 1

Expliquez le rôle de la classe Java prédéfinie **ArrayList<E>**.  
Dans votre explication précisez ce que signifie la syntaxe **<E>**.

**Le rôle de la classe ArrayList<E>.est de gérer une collection dont les éléments sont de type E. E dénote n'importe quelle classe C, par exemple ArrayList<Individu>.**

**La gestion d'une telle collection permet par exemple de:**

- **créer une collection de type E. La collection est vide.**
- **ajouter un élément de type E dans une collection**
- **supprimer un élément de la collection se trouvant à un rang donné**
- **rechercher un élément de type E dans la collection**

### Q 2

Expliquez de quoi est composée une classe en JAVA (Détaillez, ne restez pas trop général).

**Une classe JAVA est composé de :**

- **une entête qui permet de préciser l'héritage de la classe (par défaut hérite de Object) et les interfaces qu'elle implémente (optionnel)**
- **les attributs de la classe. Ces attributs peuvent être soit statiques ou non statiques. S'ils sont statiques cela veut dire qu'ils sont communs à toutes les instances de la classe, sinon ils sont propres à une instance donnée**
- **les méthodes de la classe. Les méthodes peuvent être soit statiques ou non statiques. Si elles sont statiques alors on appelle ces méthodes à travers le nom de la classe, sinon à travers un objet ou this.**

### Q 3

Expliquez ce qu'est un constructeur en JAVA.

**Un constructeur en JAVA est une méthode particulière d'une classe qui permet de créer un objet, on parle de création d'une instance de la classe. Il alloue les attributs de la classe en mémoire et initialise les valeurs de ces attributs en fonction des paramètres de la méthode ou en appelant des traitements (méthodes internes de la classe ou méthodes appartenant à d'autres classes).**

**Toute classe a un constructeur, au moins le constructeur par défaut de JAVA (pas de paramètres et les attributs ont les valeurs par défaut de JAVA).**

**Fin de la 1<sup>ère</sup> partie**

---

**2<sup>ème</sup> PARTIE : PROGRAMMATION (avec document)**

---

**Problème 1 [15 points]**

```
public ArrayList<Operation> getOperationsType(String type)
{
    ArrayList<Operation> liste = new ArrayList<Operation>();
    for(Operation ope : elements)
        if(type.equals(ope.getType()))
            liste.add(ope);
    return liste;
}
```

**Problème 2 [35 points]**

```
public class ComptesEnAnomalie
{
    //Format des noms de fichier des comptes
    static private String formatCompte = "data/Compte%s.txt";

    private ArrayList<String> comptes; //Comptes en anomalie

    private double seuil; // seuil de découvert (valeur negative)

    //Constructeur
    public ComptesEnAnomalie(double seuil)
    {
        this.comptes = new ArrayList<String>();
        this.seuil = seuil;
    }

    // Ajouter un compte si en anomalie
    //
    public void ajouter(String nomDuCompte)
    {
        String nomFichier = String.format(formatCompte,
                                           nomDuCompte);
        Compte c = new Compte(nomFichier);
        if (c.solde()<seuil)
            comptes.add(nomDuCompte);
    }
}
```

Dans la classe Compte.java :

```
public double solde()
{
    double solde=0.0;
    for(Operation ope:elements)
        if (! ope.getBudgete())
            solde=solde+ope.getMontant();
    return solde;
}
```

***(Fin du sujet)***