

IPST-CNAM  
Programmation JAVA  
NFA 032  
Mercredi 21 Juin 2017

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

1<sup>ère</sup> Session NFA 032

*L'examen se déroule en deux parties. Une première partie de 1h15mn, sans document, consacrée à des questions de cours, et une deuxième partie de 1h 15mn, avec document, consacrée en 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 les codes commentés en Java.*

---

**1<sup>ère</sup> PARTIE : COURS (sans document)**  
**Durée: 1h15**

---

**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
  - -½ pour chaque réponse fausse
- **les questions à 3 propositions** dont 1 seule proposition est fausse
  - + ½ 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.

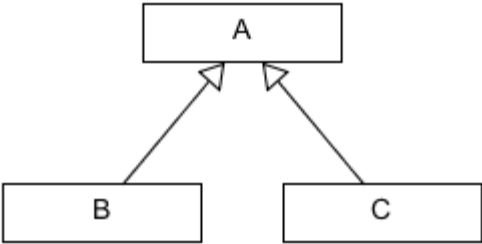
N'oubliez pas d'inscrire en en-tête du QCM, votre nom et prénom.

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

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

En programmation objet, le principe d'héritage permet de :		Q 1.
1	factoriser les attributs de plusieurs classes en une classe dont elles héritent	
2	spécialiser une classe en créant une classe dérivée qui surcharge certaines des méthodes de la classe d'héritage	
3	rendre plus performant en terme d'exécution le programme objet	

Par définition de l'héritage, des classes peuvent hériter d'une classe abstraite.		Q 2.
1	OUI	
2	NON	

Soit le symbolisme UML suivant :		Q 3.
 <pre> classDiagram     class A     class B     class C     B -- &gt; A     C -- &gt; A </pre>		
Cela signifie que B et C héritent de A.		
1	OUI	
2	NON	

Soit les classes A et B qui héritent de la classe C. A et B contiennent la déclaration d'un même attribut. (même type et même nom). Cela est une erreur de programmation et déclenche une erreur de compilation car cet attribut doit se trouver dans la classe C.		Q 4.
1	OUI	
2	NON	

Une classe A peut accéder à un attribut d'une autre classe B (sans utiliser de getteur) quand :		Q 5.
1	A hérite de B et que cet attribut est déclaré protected	
2	cet attribut est public	
3	A hérite de B et que cet attribut est déclaré private	

Soit la classe A qui hérite de B, alors A a accès à tous les constructeurs publics de B		Q 6.
1	OUI	
2	NON	

On a une classe A qui contient deux constructeurs :		Q 7.
<pre> public A(int x){ } public A(){ } </pre>		
On a la classe B qui hérite de A, qui contient le constructeur suivant :		
<pre> public B(){     attr = 100; } </pre>		
1	Le code de la classe B est correct	
2	Le code de la classe B n'est pas correct	

Soit une classe B qui hérite d'une classe A et, A et B n'ont pas de constructeur codé. Lors de la création d'un objet de B une erreur d'exécution se produit car A et B n'ont pas de constructeur		Q 8.
1	OUI	
2	NON	

En programmation objet, une collection polymorphe est une collection dont les éléments sont d'un type donné. En JAVA, ce type peut être :		Q 9.
1	une classe quelconque	
2	une classe abstraite	
3	une liste de classes qui héritent toutes d'une même classe abstraite	

Il est impossible d'instancier une classe abstraite.		Q 10.
1	OUI	
2	NON	

Une classe abstraite peut avoir un constructeur		Q 11.
1	OUI	
2	NON	

Tous les attributs d'une classe abstraite doivent être <b>protected</b> .		Q 12.
1	OUI	
2	NON	

Soit trois classes A, B et C qui héritent de la classe abstraite H et H implémente l'interface I. Pour créer une collection polymorphe, on peut faire :		Q 13.
1	ArrayList<I> elements;	
2	ArrayList<H> elements;	
3	ArrayList<A,B,C> elements	

Soit les classes quelconques A et B qui héritent de C. C n'est pas une classe abstraite. Soit la classe Stock contenant l'attribut : ArrayList<C> elements; On peut écrire :		Q 14.
<pre> elements.add( new A() ) elements.add( new B() ) elements.add( new C() ) </pre>		
1	OUI	
2	NON	

Soit une interface Java <b>InterfaceEx</b> et soit une méthode de la classe A dont la signature est la suivante :		Q 15.
<pre> public String unTraitement(InterfaceEx i) </pre>		
1	L'interface <b>InterfaceEx</b> doit implémenter la méthode <b>unTraitement</b>	
2	La classe A doit implémenter les méthodes de l'interface <b>InterfaceEx</b>	
3	Il est possible de passer en paramètre de la méthode <b>unTraitement</b> un objet dont la classe d'appartenance B implémente l'interface <b>InterfaceEx</b> .	

En JAVA, on déclare une exception en créant une classe qui hérite de la classe <b>Exception</b>		Q 16.
1	OUI	
2	NON	

En JAVA, une exception est un objet		Q 17.
1	OUI	
2	NON	

Soit le code suivant :		Q 18.
<pre> public void action(int parametre) {     if (parametre==0)         throw new RuntimeException("Erreur");     else         faireUnTraitement(); } </pre>		
Ce code est correct.		
1	OUI	
2	NON	

En JAVA, l'instruction suivante permet de déclencher une exception		Q 19.
<pre> throw new MyException("Impossible de faire l'action"); </pre>		
avec MyException une classe qui hérite de Exception		
1	OUI	
2	NON	

La déclaration de la méthode suivante :		Q 20.
<pre> public void traitement(String s) throws MyException </pre>		
précise que la méthode doit capturer l'exception MyException dans le corps de sa méthode.		
1	OUI	
2	NON	

Soit le code suivant :		Q 21.
<pre> try{     System.out.println("AAA");     call();     System.out.println("BBB"); } catch(Exception ex) {     System.out.println("CCC"); } catch(MyException ex) {     System.out.println("DDD"); } </pre>		
avec la méthode call qui déclenche l'exception MyException.		
Ce code affiche :		
1	AAA CCC	
2	AAA CCC DDD	
3	AAA DDD	

En Java, la classe Collections permet de trier les éléments de n'importe quelle collection (classe qui implémente l'interface List) grâce à la méthode <b>Collections.sort(List&lt;T&gt; list)</b> Cette méthode fonctionne si la classe d'appartenance, ici T, des éléments de la collection implémente l'interface :		Q 22.
1	Comparator	
2	Comparable	
3	Comparer	
En JAVA, une collection est une classe qui implémente, directement ou indirectement, l'interface Collection<E>		Q 23.
1	OUI	
2	NON	
En JAVA, toutes les collections ont la propriété d'itération. C'est-à-dire qu'il existe un moyen de parcourir les éléments de la collection.		Q 24.
1	OUI	
2	NON	
La classe ArrayList hérite de la classe abstraite AbstractList. La classe HashSet hérite de la classe abstraite AbstractSet. La classe AbstractList gère des collections dont les éléments sont ordonnés suivant un index d'insertion. La classe AbstractSet gère des collections dont les éléments ne sont pas ordonnés et n'ont pas d'index d'insertion.		Q 25.
1	OUI	
2	NON	
Une collection polymorphe est une collection qui hérite des 3 interfaces List, Set et Map		Q 26.
1	OUI	
2	NON	
En Java, une collection polymorphe de contact est une collection qui est créée de la manière suivante :		Q 27.
1	ArrayList< ?> collection = new ArrayList< ?>()	
2	ArrayList<ContactAbstract> collection = new ArrayList< ContactAbstract >() où ContactAbstract est une classe abstraite dont héritent toutes les classes d'éléments que l'on veut ajouter	
En Java, une liste chaînée (ex: classe LinkedList) est une liste dont les éléments ne sont pas continus dans la mémoire de la JVM..		Q 28.
1	OUI	
2	NON	
En Java, pour écrire dans un fichier ou pour écrire dans un Socket, on peut utiliser les mêmes méthodes d'écriture et de lecture.		Q 29.
1	OUI	
2	NON	
En Java, la classe File permet, entre autre, de parcourir les fichiers contenus dans un répertoire.		Q 30.
1	OUI	
2	NON	

Le code suivant crée un fichier de nom "exemple.txt" dans le répertoire courant d'exécution. Le fichier est créé, vide de toute information		Q 31.
<pre> FileInputStream fis = new FileInputStream(new File("exemple.txt")); DataInputStream dis = new DataInputStream(fis); dis.close() </pre>		
1	OUI	
2	NON	

Soit un serveur qui réalise le traitement suivant :		Q 32.
<pre> ServerSocket ssoc = new ServerSocket(9100); Socket soc = ssoc.accept(); DataInputStream dis= new DataInputStream(soc.getInputStream()); System.out.println(dis.readUTF()); </pre>		
Pour afficher sur le serveur la valeur "HELLO", le client doit écrire le code suivant :		
1	<pre> Socket soc = new Socket("localhost",9100); OutputStream os=soc.getOutputStream(); PrintStream ps=new PrintStream(os); ps.print("HELLO"); soc.close(); </pre>	
2	<pre> Socket soc = new Socket("localhost",9100); OutputStream os=soc.getOutputStream(); DataOutputStream dos=new DataOutputStream(os); dos.writeUTF("HELLO"); soc.close(); </pre>	

En JAVA, le socket est une technologie qui permet, sur une même machine, d'échanger des informations entre deux programmes (JVM) Java.		Q 33.
1	OUI	
2	NON	

Soit un programme Java P1 (JVM) qui crée un serveur de socket sur le port 9100. Soit un programme Java P2 (JVM) qui crée un serveur de socket sur le port 9100.		Q 34.
1	Les deux programmes peuvent s'exécuter sur la même machine	
2	Le programme P1 peut s'exécuter sur la machine A, et le programme P2 peut s'exécuter sur une autre machine B	

En JAVA, sur un socket, il est possible d'écrire n'importe quel type d'information (type primitifs, types référence)		Q 35.
1	OUI	
2	NON	

## 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.

### **QUESTION 1 :**

Expliquez les principes importants qui permettent de sauvegarder une collection polymorphe dans un fichier.

### **QUESTION 2 :**

Soit la méthode suivante de la classe prédéfinie java Collection :

```
public static void sort(List<T> list, Comparator<? super T> c)
```

Expliquez le fonctionnement de cette méthode.

### **QUESTION 3 :**

Quel est le rôle de la classe ServerSocket de Java ? Expliquez son fonctionnement.

***FIN DE LA 1<sup>ère</sup> PARTIE***

---

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

### Durée: 1h15

---

#### **PROBLEME 2 [50 points]**

On se propose de créer une classe d'IHM (**IHMStock**) (*ce n'est pas une ihm distante*) qui utilise la classe Site et la classe Produit décrites en Annexe.

Le rôle de cette IHM est de gérer le stock des produits.

**1/** Cette IHM permet de saisir la référence du produit, son nom, son prix et sa quantité. Si le produit existe déjà dans le stock alors on met à jour le nom et le prix (si non vides), et on ajoute la quantité à celle existante, sinon on ajoute le nouveau produit avec le nom, le prix et la quantité saisie.

Une fois l'action réalisée, l'IHM affiche tout le contenu du stock.

Si la quantité ou le prix saisis sont incorrectes alors l'action ne peut pas se faire et l'IHM affiche un texte d'erreur.

Ecrire la classe **IHMStock**.

Ecrire le code d'impact de la classe **Site** donnée en annexe.  
(IHMStock appelle les méthodes de la classe Site).

*NB : Pour créer les éléments graphiques de l'IHM, vous utilisez la classe Formulaire, vue en cours et en TP.*

**2/** On veut ajouter un bouton dans l'IHM permettant, de trier les produits du stock par ordre croissant sur la quantité du produit.

Ecrire la méthode de la classe Site, qui sera appelée depuis l'IHM, qui permet de réaliser cette action. Cette méthode retourne, sous la forme d'une chaîne de caractère, la liste des produits du stock.

*Ne pas écrire le code d'impact sur l'IHM.*

*Si d'autres modifications sont nécessaires dans la classe Site et dans la classe Produit, écrire le code de ces modifications.*

**3/** On veut écrire dans un fichier texte la liste des produits qu'il faudra recommander aux fournisseurs si leur quantité est en-dessous d'un certain seuil saisi dans l'IHM.

Le nom de ce fichier texte est en dur, par exemple : ComFournisseurs.txt.

Chaque ligne du fichier texte est de la forme : <référence> <quantité>

Ecrire la méthode de la classe Site, qui sera appelée depuis l'IHM, qui permet de réaliser cette action.



**ANNEXE 1 :**

```
public class Site
{
    private ArrayList<Produit> stock;    // Les produits du stock

    public Site()
    {
        stock = new ArrayList<Produit>();
    }

    public Produit chercherProduit(String reference)
    {
        for(Produit p:stock)
            if (p.getReference().equals(reference)) return p;
        return null;
    }

    public String listerTousProduits()
    {
        String res="";
        for(Produit prod : stock)
            res=res+prod.toString()+"\n";

        return res;
    }
}

public class Produit
{
    // Les caracteristiques d'un Produit
    //
    private String  reference;    // reference du produit
    private String  nom;         // nom du produit
    private double  prix;        // prix du produit
    private int     quantite;    // quantité du produit

    // Constructeur
    //
    public Produit(){ }

    public Produit(String reference,
                    String nom,
                    double prix,
                    int quantite)
    {
        this.reference = reference;
        this.nom = nom;
        this.prix = prix;
        this.quantite = quantite;
    }

    public String toString()
    {
        return String.format("%-15s %-50s %3.2f   %3d",reference,nom,prix,quantite);
    }

    public String getReference(){return reference;}
    public double getPrix(){ return prix;}
    public String getNom(){return nom;}
    public int getQuantite(){return quantite;}
    public void setQuantite(int quantite){this.quantite=quantite;}
    public void setNom(String nom){this.nom=nom;}
    public void setPrix(double prix){this.prix=prix;}
}
}
```

**(Fin du sujet)**