

IPST-CNAM
Programmation JAVA
NFA 032
Mercredi 22 Juin 2016

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

1^{ère} 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^{ère} 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
 - -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:
------	---------

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

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 B :		Q 2.
1	une erreur d'exécution se produit car A et B n'ont pas de constructeur	
2	l'objet B est créé, les attributs de A et B sont alloués en mémoire et initialisés par défaut	

Soit une classe B qui hérite d'une classe A. A a un seul constructeur codé avec des paramètres. B n'a pas de constructeur codé. Cela provoque une erreur de compilation.		Q 3.
1	OUI	
2	NON	

En programmation objet, une collection est polymorphe si la classe d'appartenance des éléments de la collection hérite de plusieurs classes		Q 4.
1	OUI	
2	NON	

Une classe abstraite est une classe dans laquelle au moins une méthode est abstraite		Q 5.
1	OUI	
2	NON	

Une classe abstraite ne peut pas avoir de constructeur codé.		Q 6.
1	OUI	
2	NON	

Tous les attributs d'une classe abstraite doivent être publics sinon les classes qui héritent de cette classe abstraite ne peuvent pas accéder directement à ses attributs.		Q 7.
1	OUI	
2	NON	

La caractéristique "protected" d'un attribut de la classe C est utilisée pour que les classes qui héritent de C puissent accéder directement à cet attribut		Q 8.
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 9.
1	ArrayList<I> elements;	
2	ArrayList<H> elements;	
3	ArrayList<A,B,C> elements	

Soit le code suivant :		Q 10.
<pre>public class A extends B implements C { private int attr_A; public A() { attr_A = 10; attr_B = "TOTO"; C.attr_C = 100; } }</pre>		
1	attr_B est un attribut protected de B	
2	attr_B est un attribut privé de B	
3	attr_C est un attribut static public de l'interface C	

Soit le code suivant :		Q 11.
<pre>abstract public class H { public H(){ } } public class A extends H { static public getInstance(){ return new A(); } }</pre>		
Ce code est correct		
1	OUI	
2	NON	

En JAVA, une exception est un objet dont la classe d'appartenance peut être :		Q 12.
1	la classe prédéfinie Exception	
2	une classe qui hérite de la classe RuntimeException	
3	une classe qui hérite de la classe Object car Object hérite de la classe Exception	

Le code suivant est correct :		Q 13.
<pre>public void action(int parametre) throws Exception { if (parametre==0) throw new Exception("Erreur"); else faireLeTraitement(); }</pre>		
1	OUI	
2	NON	

En JAVA, l'instruction suivante permet de déclencher une exception <pre>throw new MyException("Impossible de faire l'action");</pre> avec MyException une classe qui hérite de Exception		Q 14.
1	OUI	
2	NON	

La déclaration de la méthode suivante : <pre>public void traitement(String s) throws MyException</pre> précise que la méthode peut déclencher l'exception MyException dans le corps de sa méthode.		Q 15.
1	OUI	
2	NON	

Soit le code suivant :		Q 16.
<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 <i>call</i> qui déclenche l'exception <i>MyException</i> .		
Ce code affiche :		
1	AAA CCC DDD	
2	AAA CCC	
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 statique : Collection.sort(List<T> list) Cette méthode fonctionne si la classe T implémente l'interface Comparable		Q 17.
1	OUI	
2	NON	

En Java, la classe Collection est une classe abstraite dont hérite la classe ArrayList		Q 18.
1	OUI	
2	NON	

En Java, la différence entre l'interface List et l'interface Collection est que l'interface List définit des méthodes supplémentaires basées sur la notion d' « index ».		Q 19.
1	OUI	
2	NON	

Soit la déclaration suivante : public ArrayList<AbstractH> liste = new ArrayList<AbstractH>(); avec AbstractH qui est une classe abstraite et les classes A et B qui héritent de AbstractH. Nous pouvons écrire le code suivant :		Q 20.
1	liste.add(new A());	
2	liste.add(new B());	
3	liste.add(new AbstractH());	

Soit la déclaration suivante : public ArrayList<H> liste = new ArrayList<H>(); avec H qui n'est pas une classe abstraite et les classes A et B qui héritent de H. Nous pouvons écrire le code suivant :		Q 21.
<pre> liste.add(new A()); liste.add(new B()); </pre>		
1	OUI	
2	NON	

Soit une interface Java InterfaceEx et soit une méthode de la classe A dont la signature est la suivante :		Q 22.
<pre>public String unTraitement(InterfaceEx i)</pre>		
1	L'interface InterfaceEx doit implémenter la méthode unTraitement	
2	La classe A doit implémenter les méthodes de l'interface InterfaceEx	
3	Il est possible de passer en paramètre de la méthode unTraitement un objet dont la classe d'appartenance B implémente l'interface InterfaceEx .	

Le code JAVA suivant, liste les noms des fichiers et les noms des répertoires qui se trouvent dans le répertoire "/home/jl"		Q 23.
<pre>File varfile; varfile = new File("/home/jl"); for(String nom : varfile.list()) System.out.println(nom);</pre>		
1	OUI	
2	NON	

Le code JAVA suivant, crée physiquement un fichier de nom "toto.txt" dans le répertoire "data" :		Q 24.
<pre>File varfile = new File("./data/toto.txt");</pre>		
1	OUI	
2	NON	

Le code suivant crée un fichier de nom "exemple.bin" dans le répertoire courant d'exécution. Le fichier est créé contenant des informations binaires (une chaîne, un double):		Q 25.
<pre>FileOutputStream fos = new FileOutputStream("exemple.bin"); DataOutputStream dos = new DataOutputStream(fos); dos.writeUTF("EXEMPLE"); dos.writeDouble(123.456); dos.close();</pre>		
1	OUI	
2	NON	

Le code suivant crée un fichier texte (qui peut être lu avec un éditeur de texte) de nom "exemple.txt" dans le répertoire courant d'exécution :		Q 26.
<pre>FileOutputStream fos = new FileOutputStream(new File("exemple.txt")); PrintStream ps = new PrintStream(fos); ps.println("EXEMPLE"); ps.close();</pre>		
1	OUI	
2	NON	

En JAVA, tout objet dont la classe d'appartenance hérite de la classe ObjectOutputStream, peut être sérialisé.		Q 27.
1	OUI	
2	NON	

En JAVA, pour créer un thread T, on peut :		Q 28.
1	implémenter la méthode void run() dans la classe T qui implémente l'interface Runnable	
2	implémenter la méthode void run() dans la classe Thread qui hérite de T	
3	implémenter la méthode void run() dans la classe T qui hérite de Thread	

La méthode "start" est la méthode qui permet de démarrer un thread. Cette méthode appartient à la classe Thread. On y accède directement ou par héritage de la classe Thread.		Q 29.
1	OUI	
2	NON	

Soit une classe MyThread qui hérite de Thread et qui contient un attribut défini comme suit : <code> private static int tempo;</code>		Q 30.
Alors tous les threads (instances de MyThread), partage la même valeur tempo (exemple: temps de temporisation des threads)		
1	OUI	
2	NON	

En JAVA, la classe ServerSocket est une classe prédéfinie qui hérite de Thread. Ainsi, elle permet de traiter les requêtes de plusieurs clients en parallèle		Q 31.
1	OUI	
2	NON	

Le socket est un canal de communication permettant de faire communiquer deux logiciels distants se trouvant sur deux machines à travers un réseau TCP/IP		Q 32.
1	OUI	
2	NON	

Soit le code du serveur suivant :		Q 33.
<pre> ServerSocket ssoc = new ServerSocket(9100); Socket soc = ssoc.accept(); DataInputStream dis= new DataInputStream(soc.getInputStream()); String requete = dos.readUTF(); </pre>		
1	dès son exécution la méthode <i>accept</i> retourne un socket. Le serveur se met alors en attente de lecture avec l'instruction <code>dos.readUTF()</code> , il attend qu'un client écrit sur le socket	
2	dès son exécution la méthode <i>accept</i> se met en attente qu'un client crée un socket sur le port 9100.	

En JAVA, sur un socket, il est possible d'écrire n'importe quel type d'information (type primitifs, types référence)		Q 34.
1	OUI	
2	NON	

Dans l'architecture client-serveur classique, le serveur est le "maître". C'est-à-dire que c'est à son initiative qu'il traite les requêtes des clients en demandant ces requêtes aux différents clients.		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 le rôle d'une classe abstraite dans la conception d'une collection polymorphe.

QUESTION 2 :

Expliquez avec précision les différences qui existent entre les classes **Exception** et **RuntimeException**

(principes, syntaxe de code, rôle, ...)

QUESTION 3 :

Expliquer ce qu'est la notion d'héritage dans le cadre des langages de programmation orientée objet.

FIN DE LA 1^{ère} PARTIE

2^{ème} PARTIE : PROGRAMMATION (avec document)

Durée: 1h15

PROBLEME [50 points]

Pour une banque (classe **Banque**), on se propose de gérer les comptes bancaires des clients de la banque.

Une classe abstraite **AbstractCompte** est caractérisée par :

- le nom du propriétaire du compte (string)
- le numéro du compte (string)
- le solde du compte (double)

Il y a 2 types de comptes : les comptes courants (classe **CompteCourant** qui hérite de **AbstractCompte**) et les comptes d'épargne (**CompteEpargne** qui hérite de **AbstractCompte**).

Un compte courant est caractérisé par :

- le montant d'autorisation de découvert (double de 0 à 1500 euros)

Un compte épargne est caractérisé par :

- le taux d'épargne (double)

La classe **Banque** est caractérisée par :

- une collection (ArrayList) de **AbstractCompte** qui contient tous les comptes de la banque.

1/ Ecrire les classes **Banque**, **AbstractCompte**, **CompteCourant**, **CompteEpargne**.

Vous écrivez que les attributs, le constructeur, et la méthode `toString` de chacune des classes.

2/ Un fichier texte contient tous les comptes de la banque, 1 ligne par compte.

La ligne d'un compte courant est :

COURANT;nom du client;numero compte;solde;autorisation découvert

La ligne d'un compte épargne :

EPARGNE; nom du client;numero compte;solde;taux

Exemple :

```
COURANT;LAFORGUE Jacques;8912Z;1230.50;800.00
```

```
COURANT;LAFORGUE Jacques;8913X;-330.50;400.00
```

```
EPARGNE;LAFORGUE Jacques;7856D;34500.00;4.5
```

```
COURANT;LAFONT Paul;78FGR;2300.00;0.0
```

Ecrire la méthode de la classe **Banque** :

public void importer(String nomFichier)

qui lit un fichier texte et ajoute les comptes à la collection de la classe **Banque**.

3/ Ecrire le code qui permet de trier les comptes de la collection de la classe **Banque**, ordonné par ordre alphabétique croissant sur le nom du propriétaire puis par solde décroissant.

4/ Ecrire la méthode de la classe **Banque** qui permet de retourner tous les numéros des comptes courant dont le solde est négatif et qui a dépassé le montant d'autorisation de découvert.

(Fin du sujet)