IPST-CNAM
Programmation JAVA
NFA 031
Mercredi 15 Avril 2015

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

2ème 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 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.

1ère PARTIE – 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:

Un langage de programmation orienté objet est :		Q 1.
1	un langage dont les traitements informatiques sont écrits dans une classe	**X**
2	un langage dont les traitements informatiques sont écrits en dehors de la classe	

Dans un langage orienté objet, un principe fort est que les attributs non statiques sont alloués dans une instance d'une classe appelé un objet.		Q 2.
1	OUI	**X**
2	NON	

Le paramètre **-classpath** ou la variable d'environnement **CLASSPATH** 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 3.
1	OUI	**X**
2	NON	

La compilation d'un fichier **.java** (usage de la commande javac)		Q 4.
1	crée un et un seul fichier .class	
2	crée un fichier .jar	
3	crée autant de fichier .class que de classes définies dans le fichier .java	**X**

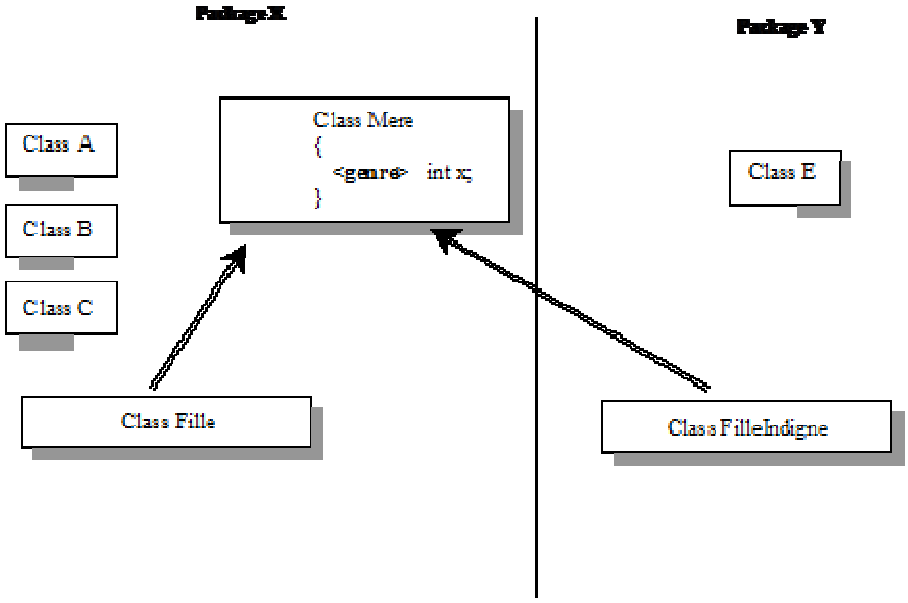
Soit le fichier suivant C:\CodeJava\exercices\fr\cnam\util\Terminal.java. Le fichier Terminal.java contient en 1ère ligne : **package exercices.fr.cnam.util;** Le répertoire C:\bin est vide. Dans C: on réalise la commande suivante : **javac -d C:\bin C:\CodeJava\exercices\fr\cnam\util\Terminal.java**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

Le fichier Exemple.java est dans le répertoire C:\Test\src L'en-tête du fichier Exemple.java est : **package fr.cnam.prog;** et contient la classe public Exemple. Le répertoire C:\Test\bin existe et est vide. On est dans le répertoire C:\Test et on réalise la commande de compilation suivante : javac –d bin src/Exemple.java		Q 6.
1	Dans le répertoire bin on obtient la création de l'arborescence de répertoires suivants : fr/cnam/prog et dans le répertoire prog la création du fichier Test.class	**X**
2	Dans le répertoire bin on obtient la création de : Test.class	

Quand on compile un fichier .java contenant une classe, de nom Prog, qui contient une méthode main, alors on obtient un fichier binaire Prog.exe que l'on peut ensuite exécuter sur le système d'exploitation.		Q 7.
1	OUI	
2	NON	**X**

Soit la classe C1 dont tous les attributs sont privés et statiques. Soit la classe C2 appartenant au même package que C1. Dans ce cas, les méthodes de C2 peuvent accéder directement aux attributs de C1		Q 8.
1	OUI	
2	NON	X

Soit un fichier Prog.java contenant une classe publique Prog et une classe privée Prog2. Chacun de ces classes contiennent une méthode public static void main(String... args)		Q 9.
1	On peut lancer une exécution avec la commande : java Prog	X
2	On peut lancer une exécution avec la commande : java Prog2	
3	La classe Prog2 ne peut pas contenir de méthode main	

Soit la répartition des classes suivantes (la classe Fille et la classe FilleIndigne hérite de la classe Mere) : 		Q 10.
1	si <genre> est private alors l'attribut x est accessible depuis la Class Fille	
2	si <genre> est protected alors l'attribut est accessible depuis Class FilleIndigne	X
3	si <genre> est vide alors l'attribut est accessible depuis la classe A	X

Un constructeur d'une classe C est une méthode de la classe dont la forme de déclaration est : public C C(*paramètres*). Exemple d'utilisation : C x = new C(12,"xx")		Q 11.
1	OUI	
2	NON	X

En JAVA, une classe peut contenir plusieurs constructeurs		Q 12.
1	OUI	X
2	NON	

En JAVA, un objet est un pointeur		Q 13.
1	OUI	X
2	NON	

La classe StringArrayList, de Java, permet de créer une collection de chaines de caractère. Chaque élément de cette collection (un tableau) est une String.		Q 14.
1	OUI	
2	NON	X

La classe StringBuffer est une classe qui permet de créer des chaines de caractères et contient des méthodes permettant de modifier les caractères de la chaine de caractères.		Q 15.
1	OUI	**X**
2	NON	

Dans la classe String la méthode d'objet ***int indexOf(char c)*** permet de rechercher dans une String la première occurrence du caractère c. Elle retourne la position de ce caractère.		Q 16.
1	OUI	**X**
2	NON	

En JAVA, le code suivant permet de remplacer un caractère d'une chaine de caractère par un autre :
`String str = new String("abcdefg");`
`str.replace(str,3,'x');`
`System.out.println(str);`
Ce code affiche : abcxefg
		Q 17.

1	OUI	
2	NON	**X**

Soit le code suivant :
`int[] tab_int;`
`tab_int[0]=12;`
`tab_int[1]=3;`
 	
Ce code s'exécute correctement.
		Q 18.

1	OUI	
2	NON	**X**

La méthode **equals** de la classe Object permet de tester l'égalité de deux objets. Cette méthode teste l'égalité des attributs de chacun des objets. Elle retourne VRAI si tous les attributs sont égaux deux à deux.		Q 19.
1	OUI	
2	NON	**X**

Soit le code suivant est :
`int[][] matrice = new Matrice[10][20];`
`matrice[0] = new int[3];`
On peut écrire ensuite les codes suivants :
		Q 20.

1	`System.out.println(matrice[0][0]);`	**X**
2	`System.out.println(matrice[0][3]);`	
3	`System.out.println(matrice[1][1]);`	**X**

Soit le code suivant :
`public class Exemple {`
`  public void main(String[] args) {`
`  System.out.println(args[0]);`
`}}`
Commande d'exécution : java Exemple toto
		Q 21.

1	Ce programme ne se compile pas car il y a une erreur de syntaxe	
2	L'exécution échoue car il y a une erreur d'exécution	**X**
3	Ce programme s'exécute correctement et affiche : "toto"	

Soit le code JAVA suivant : Q 22.

```
public class C {
   public int attribut;
   public C(){attribut=10;}
   public static void main(String[] args)
   {
      attribut = 10;
   }
}
```

Ce code est correct

1	OUI	
2	NON	X

Un attribut déclaré en **final static** est un attribut qui est protégé en écriture. Il est une constante du programme. Q 23.

1	OUI	X
2	NON	

En JAVA, pour créer un objet, on peut créer une méthode static de la classe dont le rôle est de créer un objet et de le retourner. Exemple : Q 24.

```
public class Individu
{
  public String nom;
  public String prenom;
  public int age;

  static public Individu Individu(String nom, String prenom, int
age)
  {
    Individu ind = new Individu();
    ind.nom=nom;  ind.prenom=prenom;  ind.age = age;
    return ind;
  }
}
```

Utilisation de cette méthode dans un programme Java :

```
Individu ind = Individu.Individu("LAFONT","Pierre",30);
```

1	OUI	X
2	NON	

Soit le code suivant : Q 25.

```
Livre l1 = new Livre();
Livre l2 = l1;
l1.nom = "Face aux feux du soleil";
ArrayList<Livre> livres = new ArrayList<Livre>();
livres.add(l1);
l2.nom="Les cavernes d'acier";
System.out.println(livres.get(0).nom);
```

Ce code affiche :

1	Les cavernes d'acier	X
2	Face aux feux du soleil	

Soit le code suivant : Q 26.

```
Livre l1 = new Livre("LAFONT","Pierre",20);
Exemple.changerNom(l1,"DUPONT");
System.out.println(l1.getNom());
```

Avec

```
public class Exemple{
   public static changerNom(Livre l, String nom)
     {  l.setNom(nom);  }
```

Ce code affiche :

1	DUPONT	X
2	LAFONT	

En JAVA, le type de retour d'une méthode peut être un type primitif (int, double, char, ...) ou un type référence (objet, tableau) ou void		Q 27.
1	OUI	**X**
2	NON	

En JAVA, un tableau (tab[]) ne peut contenir que des éléments de type primitif		Q 28.
1	OUI	
2	NON	**X**

Soit le code JAVA suivant : Q 29.

```
String slue = "un/deux/trois///quatre/cinq//six/sept/huit";
StringTokenizer str = new StringTokenizer(slue,"/");
String res="";
while (str.hasMoreTokens())
{
   String s = str.nextToken();
   res=res+s+">";
}
Terminal.ecrireString (res);
```

Ce code :affiche :

1	un>deux>trois>>>quatre>cinq>>six>sept>huit>	
2	un>deux>trois>quatre>cinq>six>sept>huit>	**X**

En Java, il est possible de modifier le contenu d'un tableau passé en paramètre d'une méthode		Q 30.
1	OUI	**X**
2	NON	

Soit le code JAVA suivant : Q 31.

```
ArrayList<String> tab1 = new ArrayList<String>(4);
String[]          tab2 = new String[4];
tab1.add("UN"); tab1.add("DEUX");
tab2[0]="UN"; tab2[1]="DEUX";
boolean egal=true;
for(int i=0;i<tab1.size();i++)
    if ( ! (tab1.get(i).equals(tab2[i])) egal=false;
System.out.println(egal);
```

1	Ce code affiche: true	**X**
2	Ce code affiche : false	

En JAVA, faire la déclaration : Individu ind; consiste à créer un objet dont la classe d'appartenance est Individu. On peut alors utiliser les attributs de l'objet ind.		Q 32.
1	OUI	
2	NON	**X**

S'il existe au moins un constructeur avec des paramètres et pas de constructeur sans paramètres alors le constructeur par défaut est toujours accessible		Q 33.
1	OUI	
2	NON	**X**

En JAVA, il est possible d'augmenter la taille d'un tableau.		Q 34.
1	OUI	
2	NON	**X**

Le package qu'il faut importer pour utiliser la classe ArrayList est : java.lang		Q 35.
1	OUI	
2	NON	**X**

(Tourner 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

Précisez, avec détail, quelle utilité a la classe prédéfinie du langage Java : ArrayList<E>
Que représente <E> ?

Cette classe permet de gérer une collection dont les éléments sont gérés sous la forme d'un tableau dynamique. On appelle cela une Liste.
Les éléments sont accessibles par un indice et sont ajoutés. Ils sont rangés dans l'ordre d'ajout.
Il existe de nombreuses méthodes permettant d'insérer, supprimer, rechercher un élément.
Cette collection n'est pas limité en taille physique des éléments.

E désigne le type des éléments de la collection. Ce type est une classe ou une interface.

Q 2

Dans la programmation JAVA, quels sont les rôles d'un constructeur ?
Quelle est l'utilité de créer plusieurs constructeurs ?

Les rôles d'un constructeur sont :
- allouer l'objet en mémoire
- initialiser les attributs de l'objet
- appeler des méthodes qui sont exécutées lors de la création de l'objet

Q 3

Expliquez, avec détail, comment on crée, on compile et on exécute un programme Java (Vous avez le choix de la solution).

On crée un programme Java en écrivant différentes classes dans des fichiers textes. Au moins une classe contient la méthode 'main'.
On utilise la commande javac pour compiler la classe contenant la méthode 'main' en précisant si besoin l'accès aux packages. Les fichiers .class sont générés par la compilation dans un répertoire.
Ensuite, pour exécuter le programme, dans ce répertoire, on utilise la commande 'java' (exécution d'une JVM) qui prend en entrée le nom de la classe contenant la méthode main et si besoin l'accès aux packages.

Fin de la 1ère partie

Vous devez rendre votre copie vierge double et le QCM dans la copie double avant de commencer la 2ème partie.

2ème PARTIE : PROGRAMMATION (avec document)

Probleme 1 [25 points]

Soit une classe RendezVous qui contient les attributs suivants :

String dateDebut
String heureDebut
int duree;
String texte:

La date de début est au format JJ/MM/AAAA (ex: 03/01/2015)
L'heure de début est au format HHhMMmn (ex 10h30mn)
Tous les getteurs de ces attributs sont définis.

1/ Ecrire le code **complet** de la méthode statique qui prend en entrée une collection de rendez-vous. Cette méthode recherche le rendez vous le plus récent, le supprime de la collection et retourne le rendez-vous supprimé.
La signature de cette méthode est :

```
public static RendezVous plusRecent(ArrayList<RendezVous> rdvs)
```

NB: Vous utilisez la class Calendar pour convertir la date String JJ/MM/AA en secondes.

2/ En utilisant la méthode précédente, écrire la méthode static qui permet de trier par ordre croissant les rendez vous d'une collection.
La signature de cette méthode est :

```
public static ArrayList<RendezVous> trier(ArrayList<RendezVous>
rdvs)
```

Cette méthode retourne la collection triée.

Probleme 2 [25 points]

On veut gérer des réservations de créneaux horaires de plusieurs salles de réunion.
On a donc une classe **SallesReunion** qui contient une collection de **Salle**.
La classe **Salle** contient les informations suivantes :
- le nom de la salle (String)
- une collection de créneaux de réservation **Creneau**

Un Creneau est caractérisé par une date, une heure de début, une heure de fin et l'objet de la réservation.

On a un fichier qui contient toutes les créneaux de réservation de toutes les salles.
Chaque ligne du fichier est de la forme :
nom salle;date debut;heure début;heure fin;objet

Ecrire les classes **SallesReunion**, **Salle**, et **Creneau**, de telle manière que le constructeur de la classe SallesReunion lit le fichier et initialise ses données en fonction du contenu du fichier.

Pour cela, vous avez la méthode **public static String[] lireFichierTexte(String nomFichier)** de la classe Terminal qui retourne toutes les lignes du fichier dans le tableau de retour.

NB: N'écrivez pas les getteurs et setteurs des attributs de vos classes.

(Fin du sujet)