Epreuve Professionnelle

Cadre Supérieur en Etude et Développement Informatique

17/11/2013

Consignes

Cette épreuve comporte des questions théoriques et pratiques dont la nature est de deux types:

- **les questions à réponse directe**: dans ce cas, le candidat doit donner une solution ou compléter une proposée.
- **les questions à plusieurs propositions** dont lesquelles le candidat doit choisir une ou plusieurs bonnes réponses.

Si le candidat doit raturer une croix, il doit le faire correctement afin qu'il n'y ait aucune ambiguïté.

Partie I : Conception

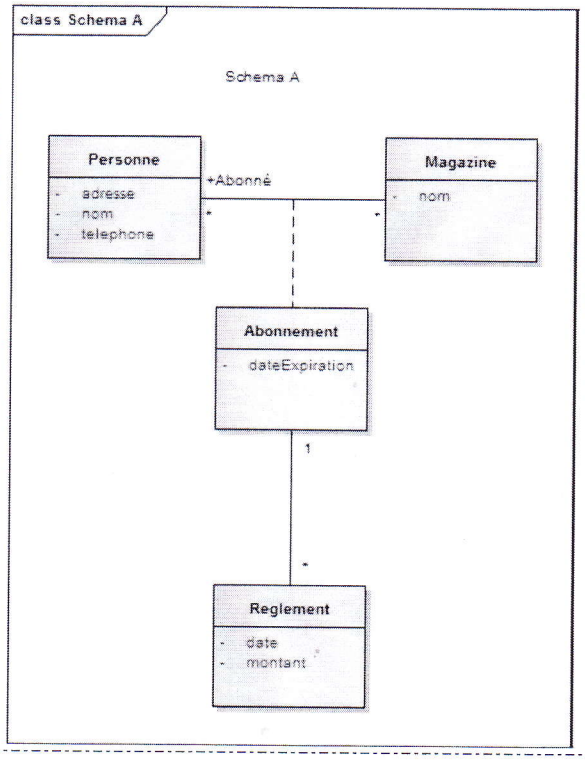
Question 1.

En trigonométrie, on a besoin de calculer le sinus, le cosinus, la tangente des angles et la valeur du nombre PI. La classe *Angle* existe déjà. Proposez une structure qui regroupe ces fonctions.

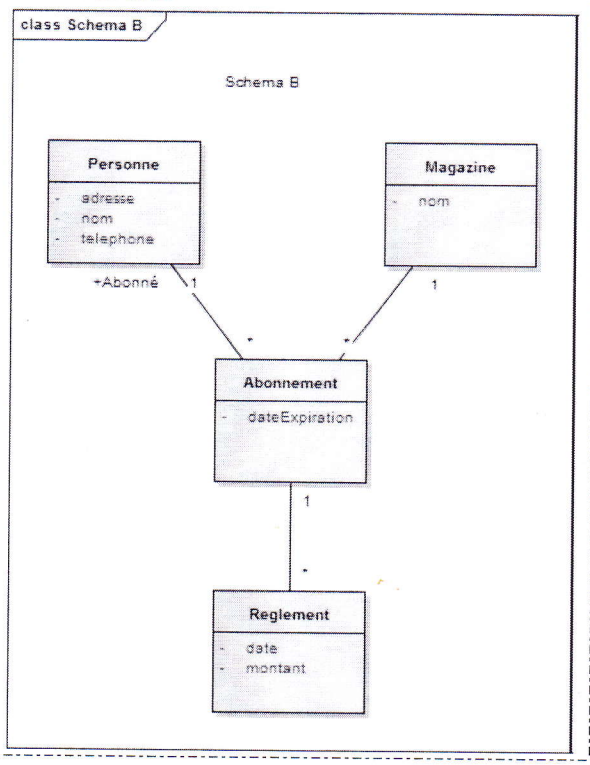
Question 2.

Une personne peut être abonnée à plusieurs magazines. Un magazine peut avoir de nombreux abonnés. Pour chaque abonnement, il est important de connaitre la date et le montant de chaque règlement ainsi que la date d'expiration de l'abonnement. Laquelle des 2 conceptions suivantes vous semble la meilleure ?

☐ Schéma A

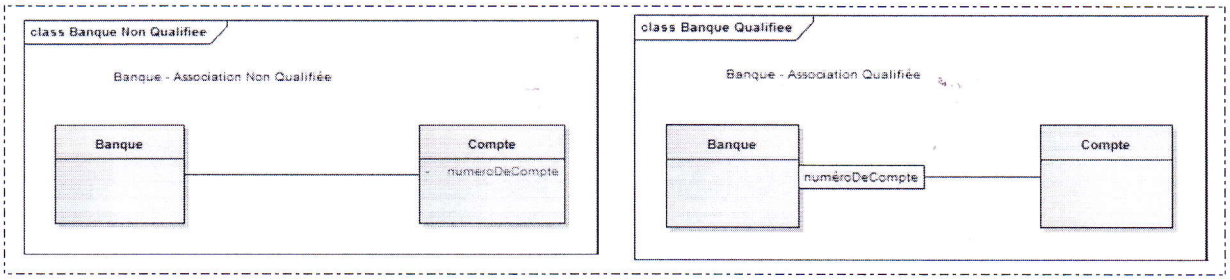


☐ Schéma B



Question 3.

Une banque gère de nombreux comptes. Un compte appartient à une seule banque. Un numéro de compte permet d'identifier un compte unique dans une banque donnée. Un numéro de compte est relatif à une banque. Indiquez la multiplicité selon que l'association soit qualifiée ou qu'elle ne le soit pas.



Question 4.

Les chaînes de caractères du langage C sont codées comme un tableau de caractères non nuls, terminé par un caractère '\0'. Par exemple, la chaîne s="hello!" est codée comme suit :

s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]
'h'	'e'	'l'	'l'	'o'	'!'	'\0'

Décrivez une activité implémentant la fonction strlen, qui prend en entrée un tableau de caractères et rend un entier correspondant à la taille de la chaîne. Exemple : strlen("hello!")=6.

Partie II : Développement

Question 5.

Dans le modèle MVC, Hibernate correspond à la couche

☐ C
☐ M
☐ V
☐ Aucune

Question 6.

Quel est le scope par défaut d'un bean Spring ?

☐ session
☐ singleton
☐ prototype
☐ request

Question 7.

Quelle implémentation de Map doit-on utiliser si on veut garder l'ordre d'insertion des clés ?

☐ HashMao
☐ TreeTable
☐ HashTable
☐ LinkedHashMap

Question 8.

Pourquoi appelle-t-on Spring un conteneur léger

☐ en opposition avec EJB
☐ pour la taille des jars réduite
☐ la faible charge de développement nécessaire
☐ la possibilité de déployer une application sur un conteneur de servlet (comme tomcat)

Question 9.

Quelle est une utilisation typique des méthodes ejbSelect()?

☐ Utiliser les méthodes ejbSelect() dans des "session beans" pour déterminer les critères de sélection pour une liste
☐ Utiliser les méthodes ejbSelect() dans les méthodes ejbHome afin de renvoyer des "entity beans"
☐ Utiliser des méthodes ejbSelect() afin d'accéder à des champs d'une instance d'un bean
☐ Utiliser des méthodes ejbSelect() pour exécuter des opérations qui ne sont pas spécifiques à une instance d'un "entity bean"

Question 10.

Le "EJB Query Language" est utilisé

☐ pour compléter le standard SQL96 avec des options supplémentaires pour les relations complexes
☐ à la place de la méthode findByPrimaryKey(), pour définir des critères de sélection alternatifs
☐ pour spécifier une implémentation concrète pour les méthodes "find" et "select" des "CMP entity beans"
☐ pour pouvoir utiliser les types de données définis par les utilisateurs ou les procédures stockées dans une base de données

Question 11.

Un Middleware est :

☐ dans les architectures web, un framework, comme eclipse, d'aide au développement, à la mise au point et au déploiement des logiciels basés sur une architecture répartie
☐ dans une architecture client-serveur, une couche logicielle, utilisée par le client et le serveur pour communiquer par exemple par envoi/réception de message
☐ dans une architecture répartie, un ORB (Object Request Broker) assurant la communication entre les différentes entités du réseau

Question 12.
Pour la conception d'une architecture logicielle Intranet, la technologie CORBA n'est pas bien adaptée
☐ Oui
☐ Non

Question 13.
Les composants d'un ORB (Object Request broker) sont :
☐ Une interface Java, la classe UnicastRemoteObject, la classe LocateRegistry
☐ Eclipse, JDK, Apache
☐ Une API (fonctions de base de l'ORB), un service de nommage, un compilateur IDL

Question 14.
En RMI de Java,
☐ la classe d'appartenance d'un objet distribué, hérite de UnicastRemoteObject et implémente une interface qui décrit les méthodes distantes
☐ la classe d'appartenance d'un objet distribué, hérite de RemoteObject et implémente l'interface Remote

Question 15.
CORBA (Common Object Request Broker Architecture) est une norme de Middleware
☐ OUI
☐ NON

Question 16.
Un Design Pattern (DP) ou Patron est une norme de description des interfaces entre les composants d'une architecture logicielle orientée objet
☐ OUI
☐ NON

Question 17.
Un DP définit des principes de conception, et non des implémentations spécifiques de ces principes
☐ OUI
☐ NON

Question 18.
Quelle affirmation est vraie à propos des fichiers .class ?
☐ Un fichier .class n'est pas un fichier géré par java
☐ Un fichier .class contient du binaire indépendant de la plateforme
☐ Un fichier .class contient du code assemblé plateforme dépendant
☐ Un fichier .class contient le code source

Question 19.

Quelle affirmation est fausse à propos des interfaces ?

☐ Une classe peut implémenter plusieurs interfaces

☐ Une interface peut contenir des déclarations de méthodes et de constantes

☐ Une variable peut être déclarée du type d'une interface : MonInterface m_i ;

☐ Une interface peut contenir des déclarations de méthodes static et non static

Question 20.

Quelle affirmation est fausse à propos des classes abstraites ?

☐ Une classe peut étendre plusieurs classes abstraites

☐ Une classe abstraite est définie grâce au mot réservé abstract

☐ Une variable peut être du type d'une classe abstraite

☐ Une classe abstraite peut implémenter des méthodes

Question 21.

Quelle affirmation est fausse à propos du Garbage Collector

☐ Le programmeur peut demander le lancement du Garbage Collector

☐ Une instance d'un objet est éligible pour le Garbage Collector quand plus personne ne le référence

☐ Le Garbage Collector évite automatiquement toute perte de mémoire

☐ Le Garbage Collector démarre automatiquement quand il décide que c'est le bon moment

Question 22.

Code

```
public class A{
}
public class B extends A{
}
public static void main(String[] args){
B anObject=new B();
Boolean o1 = (anObject instanceof A);
Boolean o2 = (anObject instanceof B);
Boolean o3 = (anObject instanceof Object);
}
```

Quels booléens ont comme valeur true ?

☐ Seul o2 a comme valeur true

☐ o1, o2 et o3 ont comme valeur true

☐ Seuls o1 et o2 ont comme valeur true

☐ Seuls o2 et o3 ont comme valeur true

Question 23.

Code

```
public static void main(String[] args){
int i;
for(i=0;i<10;i++){
```

```
if(i%5==4){
i*=3;
i--;
}
}
System.out.println("i="+i);
}
```

Quelle est la valeur affichée de i ?

☐ i=11

☐ i=10

☐ i=9

☐ i=12

Question 24.

Code

```
public class Point{
private int mx=1;
private int my=1;

public Point(){
}

public void dump(){
system.out.println("x="+mx+",y="+my);
}
public static void main(String[] args){
Point p;
p.dump();
}
}
```

Quel est le résultat obtenu si vous compilez et exécutez ce code?

☐ x=0, y=0

☐ Erreur à la compilation

☐ x=1, y=1

☐ Null Pointer Exception à l'exécution

Question 25.

Code

```
public class Point{

private static Point centerPoint = new Point();

private int mx=0;
private int my=0;

public Point(){
}
```

```
public Point(int x, int y){
mx=x ;
my=y ;
centerPoint = new Point() ;
}

public static void main(String[] args){
Point p1 = new Point(1,1);
Point p2 = new Point(2,2);
}
}
```

Combien d'instances de la classe Point sont créées ?

☐ 2 instances
☐ 4 instances
☐ 5 instances
☐ 3 instances

Question 26.

Code

```
package test;

public class A{

protected int mx=0;
private int my=0;

}
```

Quelle affirmation est vraie ?

☐ mx et my sont accessibles par les sous classes de A
☐ mx est accessible par toutes les classes et my est accessible que par la classe A
☐ mx est accessible par les classes du package test et my est accessible par les sous classes de A
☐ mx est accessible des sous classes de A et my n'est accessible que par la classe A

Question 27.

Code

```
package test;

public class A{
int mx=0;
}
```

Quelle affirmation est vraie ?

☐ mx est accessible par toutes les classes du package test
☐ mx est accessible par toutes les classes
☐ mx est accessible par n'importe quelle sous classe de A

Question 28.

Code

```
public class Test{

private int mx=0;
private static int my=0;
protected int mz=0;

public static class A{
public void increase(){
my++;
}
}

public static class B{
public void increase(){
mx++;
}
}

public static class C{
public void increase(){
mz++;
}
}
}
```

Combien obtient-on d'erreurs de compilation avec ce code ?

☐ 3 erreurs de compilation

☐ 2 erreurs de compilation

☐ 4 erreurs de compilation

☐ 1 erreur de compilation

Question 29.

Code

```
public static void main(String[] args){
try{
int toto=1/0;
}catch(java.lang.NullPointerException e1){
System.out.print("NullPointer Exception,");
return;
}catch(java.lang.ArithmeticException e1){
System.out.print("Arithmetic Exception,");
return;
}finally{
System.out.print("Finally Clause,");
}
}
```

Quel est le résultat en sortie standard ?

☐ Arithmetic Exception,

☐ NullPointer Exception, Finally Clause,
☐ Finally Clause,
☐ Arithmetic Exception, Finally Clause,

Question 30.

Code

```
public static void main(String[] args){
final Thread secondThread = new Thread(){
public void run(){
for(int i=0;i<3;i++){
System.out.print("2");
}
}
};

final Thread firstThread = new Thread(){
public vois run(){
secondThread.start();
try{
secondThread.join();
}catch(java.lang.InterrupteException e){
}
for(int i=0;i<3;i++){
System.out.print("1 ");
}
}
};
firstThread.start();
}
```

Quel est le résultat en sortie standard ?

☐ 1 1 1 2 2 2
☐ 2 2 2 1 1 1
☐ Résultat aléatoire en sortie standard
☐ 2 1 2 1 2 1

Question 31.

Code

```
public static class TestThread{
private long v1=0;
private long v2=0;
private Object lock1 = new Object();
private Object lock2 = new Object();

public long getV1(){
synchronized(lock1){
return v1;
}
}
public long getV2(){
synchronized(lock2){
```

```
return v2;
}
}

public void incV1V2(){
synchronized(lock1){
synchronized(lock2){
v1++;
v2++;
}
}
}

public void mulV1V2(int mult){
synchronized(lock2){
synchronized(lock1){
v1*=mult;
v2*=mult;
}
}
}

}
```

Dans un contexte multi-thread, quelle est l'affirmation vraie ?

☐ Cette classe peut provoquer un blocage : « dead lock »

☐ Les variables v1 et v2 ne peuvent pas être lues en même temps par deux threads

☐ La variable v1 peut être modifiée par deux threads en même temps

☐ Il manque des catch sur les expressions de type InterruptedException

Question 32.

Code

```
public static boolean checkArray(Object[] list, Object o){
final int len = list.length;
for(int i=0;i<len;i++){
if(list[i].equals(o)){
return true;
}
}
return false;
}

public static boolean checkVector(Vector list, Object o){
final int len = list.size();
for(int i=0;i<len;i++){
if(list.get(i).equals(o)){
return true;
}
}
return false;
}

public static boolean checkArrayList(ArrayList list, Object o){
final int len = list.size();
```

```
for(int i=0;i<len;i++){
if(list.get(i).equals(o)){
return true;
}
}
return false;
}
```

Quelle est l'affirmation exacte ?

☐ Généralement, checkArrayList() est la méthode la plus rapide et checkArray() la plus lente

☐ Généralement, checkArray() est la méthode la plus rapide et checkArrayList() la plus lente

☐ Généralement, checkArray() est la méthode la plus rapide et checkVector() la plus lente

☐ Généralement, checkVector() est la méthode la plus rapide et checkArray() la plus lente

Question 33.

Code

```
<?php echo htmlspecialchars($nom);?>
```

Quelle est l'utilité de la fonction `htmlspecialchars()` ?

☐ Cette fonction permet d'enlever les possibles caractères anti-slash de la string

☐ Cette fonction encode les caractères spéciaux HTML afin d'éviter toutes injection de balise (X-) HTML

☐ Cette fonction permet d'interpréter les variables contenues par la chaine $nom

☐ Cette fonction permet un affichage correct des caractères étrangers quelle que soit la langue du navigateur utilisé

Question 34.

Code

```
<form name='nomformulaire' action='traitement.php' method='post'>
<input name='nom' type='text' size=50><br>
<input name='go' type='submit' value='inscription'><br>
</form>
```

Comment récupérer la variable nom dans le fichier traitement.php ?

☐ On peut récupérer la valeur de nom par un appel à $_GET['nom'] ou à $_POST['nom']

☐ On peut récupérer la valeur de nom par un appel à $_GET['nom'] ou à $_REQUEST['nom']

☐ On peut récupérer la valeur de nom par un appel à $_REQUEST['nom']

☐ On peut récupérer la valeur de nom par un appel à $_POST['nom'] ou à $_REQUEST['nom']

Question 35.

Code

```
$site='emploi.ma'
//expression 1
echo "3000 offres d'emploi en informatique et électronique sur $site" ;
//expression 2
```

```
echo "Annuaire de 1200 entreprises en informatique sur".$site ;
//expression 3
echo '3000 offres d'emploi en informatique et électronique sur $site' ;
//expression 4
echo 'Annuaire de 1200 entreprises en informatique sur'.$site ;
```

Quelle expression ne va pas afficher emploi.ma dans la phrase ?

☐ L'expression 2
☐ L'expression 3
☐ L'expression 4
☐ L'expression 1

Question 36.

Code

```
$site='recrutement.org'
//expression 1
echo "$site : Annuaire des cabinets de recrutement " ;
//expression 2
echo $site.': Annuaire des cabinets de recrutement ';
```

Quelle expression est exécutée la plus rapidement ?

☐ Cela va dépendre de la version de l'interpréteur PHP
☐ L'expression 2 est exécutée la plus rapidement
☐ L'expression 1 est exécutée la plus rapidement
☐ Les expressions 1 et 2 sont exécutées à la même vitesse

Question 37.

Code

```
//Note : le serveur a le magic_quotes_gpc à on
//On cherche à faire une requête SQL à la fois correcte et sécurisée

//expression 1
$myVariable=stripslashes($_POST['myVariable']) ;
$querysprintf("INSERT INTO products('name') VALUES('%s')",
              Mysql_real_escape_string($myVariable,$link));

//expression 2
$myVariable=$_POST['myVariable'];
$query=sprintf("INSERT INTO products('name') VALUES('%s')",
              $myVariable);

//expression 3
$myVariable=stripslashes($_POST['myVariable']) ;
$query=sprintf("INSERT INTO products('name') VALUES('%s')",
              Addslashes($myVariable));

//expression 4
$myVariable=$_POST['myVariable'] ;
$querysprintf("INSERT INTO products('name') VALUES('%s')",
              Mysql_real_escape_string($myVariable,$link));
```

Quelle expression faut-il privilégier pour construire une requête SQL juste et sécurisée ?

☐ L'expression 4
☐ L'expression 1
☐ L'expression 2
☐ L'expression 3

Question 38.

Code

```
//Définition de fonction n°1
function addtionne($b){
  $a=$a+$b ;
}

//Définition de fonction n°2
function addtionne($b){
  static $a ;
  $a=$a+$b ;
}

//Définition de fonction n°3
function addtionne($b){
  global $a,$b ;
  $a=$a+$b ;
}

//Définition de fonction n°4
function addtionne($b){
  global $a ;
  $a=$a+$b ;
}

<?php
// code utilisant la fonction
$a=1;
additionne(2);
echo $a;
?>
```

Quelle définition de la fonction additionne() donne comme résultat "3" lors de l'appel additionne(2) ?

☐ Définition 4
☐ Définition 3
☐ Définition 1
☐ Définition 3

Question 39.

Code

```
<?php
$tableau=array(1,2,3) ;
$tableau[]=0;
$tableau[]=4;
```

```
array_pop($tableau);
array_shift($tableau);
for($i=0;$i<count($tableau);$i++){
    echo $tableau[$i].'';
}
?>
```

Quel est le résultat du code ci-dessus ?

☐ Ce code donne comme affichage : 0 2 3

☐ Ce code donne comme affichage : 4 3 2

☐ Ce code donne comme affichage : 2 3 0

☐ Ce code donne comme affichage : 2 3 4

Question 40.

Code

```
<?php
class classA{
var $i=0 ;
function classA($value){
$this->i=$value;
}
function afficher(){
echo $this->i.'';
}
}

class classB extends classA{
var $j=0;
function classB($value1,$value2){
$this->i=$value1;
$this->j=$value2;
}
function afficher(){
echo $this->i.' '.$this->j.' ';
}
}

$obj1=new classA(1);
$obj2=new classB(2,3);

echo $obj1->afficher();
echo $obj2->afficher();
?>
```

Quelle affirmation est vraie ?

☐ Ce code donne comme affichage : 1 1

☐ Ce code donne comme affichage : 0 2 3

☐ Ce code donne comme affichage : 1 2 3

☐ L'héritage est interdit en PHP, ce code génère une erreur