Final SR03
Printemps 2011

NOM et PRENOM : _____

Durée 1 heure **21 JUIN 2011**

- Aucun document n'est autorisé.
- Il ne sera répondu à aucune question en cours d'examen. Si vous estimez une question ambiguë, expliquez pourquoi sur votre copie et répondez en indiquant l'hypothèse choisie. Ecrivez lisiblement.
- Répondez aux questions directement sur les copies, dans l'emplacement prévu pour la réponse.

Q1. Expliquer en quoi consistent la portabilité et la sécurité des applets. (1 point)

Q2. Donner les éléments essentiels contenus dans un fichier web.xml. (1 point)

Q3. Comment les événements sont-ils traités en java ? Détailler la réponse. (1 point)

Q4. Expliquer le processus de sérialisation en java. (1 point)

Q5. Décrivez comment les servlets récupèrent les paramètres envoyés par un formulaire. (1 point)

signature

Q6. Quelles sont les principales différences entre les servlets et les applets ? (1 point)

Q7. Le modèle bean : pour chacune des affirmations suivantes précisez à côté si elle est correcte ou non. (1 point)

a. Implémente l'interface java.io.Serializable
b. Fournit un constructeur public sans argument
c. Possède des propriétés qui peuvent être modifiées lors de son déploiement
d. Peut interroger une base de données
e. Peut écouter des événements

Q8. Faites un schéma des relations entre servlet, JSP et Bean. (2 points)

Q9. Quel est l'intérêt d'une bibliothèque de tags JSTL ? (1 point)

Q10. Quelle est la différence entre les méthodes *jsp:forward* et *jsp:include*? (1 point)

Q 11. (3 points). Considérons la partie suivante d'un fichier WSDL (calc.wsdl):

```
<definitions name="calc"
 ....
<types>
 <schema targetNamespace="urn:calc"
  ...
 </schema>
</types>

<message name="addRequest">
 <part name="a" type="xsd:double"/>
 <part name="b" type="xsd:double"/>
</message>

<message name="addResponse">
 <part name="result" type="xsd:double"/>
</message>
```

```
<message name="subRequest">
 <part name="a" type="xsd:double"/>
 <part name="b" type="xsd:double"/>
</message>
<message name="subResponse">
 <part name="result" type="xsd:double"/>
</message>
<portType name="calcPortType">
 <operation name="add">
  <input message="(1)"/>
  <output message="(2)"/>
 </operation>
 <operation name="sub">
  <input message="(3)"/>
  <output message="(4)"/>
 </operation>
</portType>
```

signature

a- Expliquer le rôle de l'élément <types>. Quel langage doit être utilisé pour définir cet élément ?
b- Compléter les éléments manquant numérotés de (1) à (4).
c- Donner le prototype (entête) des opérations définies par ce WSDL. Justifiez votre réponse en utilisant le WSDL.

Q12. (4 points) Soit deux classes java Alpha et Beta :

```
public class Alpha extends UnicastRemoteObject implements AlphaInt {
    private String label;
    private AlphaInt nextA;
    private Beta nextB;

    public Alpha() throws RemoteException { label = null; nextA = null; nextB = null; }
    public void putLabel(String s) throws RemoteException { label = s; }
    public void linkAlpha(AlphaInt a) throws RemoteException { nextA = a; }
    public void linkBeta(Beta b) throws RemoteException { nextB = b; }
    public void printAlphaChain(String prefix) throws RemoteException {
                        System.out.println(prefix + " : " + label);
                        if (nextA != null) nextA.printAlphaChain(prefix);
    }
    public void printBetaChain(String prefix) throws RemoteException {
                        if (nextB != null) nextB.printBetaChain(prefix);
    }
} //class Alpha

public class Beta implements Serializable {
    private String label;
    private Beta nextB;
    private Alpha nextA;

    public Beta() { label = null; nextB = null; nextA = null; }
    public void putLabel(String s) { label = s; }
    public void linkBeta(Beta b) { nextB = b; }
    public void linkAlpha(Alpha a) { nextA = a; }
    public void printBetaChain(String prefix) {
                        System.out.println(prefix + " : " + label);
                        if (nextB != null) nextB.printBetaChain(prefix);
    }

    public void printAlphaChain(String prefix) {
        try {
                          if (nextA != null) nextA.printAlphaChain(prefix);
        } catch(RemoteException e) {System.out.println("Erreur: " + e);}
    }
} //class Beta
```

Dans une JVM 'distante' un objet de type Alpha est instancié et déclaré au niveau d'un rmiregistry sous le nom de 'rmi://serveur/alpha'.
Dans une JVM 'locale' on compile et exécute une application java dont voici quelques extraits de code de la méthode main().
Pour chaque extrait (les extraits s'exécutant dans l'ordre : 1, ensuite 2, ensuite 3, ensuite 4), indiquer le contenu correspondant de la sortie standard au niveau de cette JVM 'locale' :

EXTRAIT n° 1	EXTRAIT n° 2

==========
Beta bachir=new Beta();
bachir.putLabel("bachir");
Beta barbara=new Beta();
barbara.putLabel("barbara");
Beta betty=new Beta(); betty.putLabel("betty");
 	
bachir.linkBeta(barbara);
barbara.linkBeta(betty);
bachir.printBetaChain("bachir")
	==========
	AlphaInt anatole =
	(AlphaInt)Naming.lookup("rmi://serveur/alpha");
	anatole.putLabel("anatole");
	
	Alpha aziz=new Alpha(); aziz.putLabel("aziz");
	aziz.linkAlpha(anatole);
	aziz.printAlphaChain("barbara");

EXTRAIT n° 3	EXTRAIT n° 4

==========
Alpha anissa=new Alpha();
anissa.putLabel("anissa");
anatole.linkAlpha(anissa);
aziz.printAlphaChain("sr03");
	==========
	anatole.linkBeta(bachir);
	anatole.printBetaChain("four");

Réponse :

Q13 (1 point):
Pour des applications parallèles telles que nous les avons vues en cours, est-ce le même programme MPI qui fonctionne au sein des différents process ? Quelle procédure de la bibliothèque est essentielle ?

Q14 (1 point) : Soit le bout de code suivant :

```
MPI_Comm_rank(MPI_COMM_WORLD,&rang);
if (rang == 0) {
        MPI_Send(&val, count, MPI_INT , 1 , etiquette, MPI_COMM_WORLD) ;
        MPI_Recv(&valeur, count, MPI_INT, 1 , etiquette, MPI_COMM_WORLD, &statut) ;
        }
else  if (rang == 1){
        MPI_Send(&val, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD) ;
        MPI_Recv(&valeur, count, MPI_INT, 0,etiquette,MPI_COMM_WORLD,&statut) ;
        }
MPI_Finalize() ;
}
```

Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.

signature