Final SR03 Printemps 2011

NOM et PRENOM:	
Durée 1 heure - 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)	
(2. Boillet les ciellents essentiers contenus dans du Hemer web.xim. (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)	
The state of the s	
signature 1	

Q6. Quelles sont les principales différences entre les	s servlets et les applets ? (1 point)
Q7. Le modèle bean : pour chacune des affirmations	s 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 argumec. Possède des propriétés qui peuvent être mo	
d. Peut interroger une base de données	contests for de son deprotement
e. Peut écouter des événements	
Q8. Faites un schéma des relations entre servlet, JSI	P et Bean. (2 points)
,	` ` `
Q9. Quel est l'intérêt d'une bibliothèque de tags JS	TL 2 (1 point)
\$7. Quei est i interet à une bibliothèque de digs 35	TE: (1 point)
Q10. Quelle est la différence entre les méthodes <i>jsp</i>	:forward et jsp:include'! (1 point)
Q 11. (3 points). Considérons la partie suivante d'u	nn fichier WSDL (calc.wsdl):
<definitions <="" name="calc" td=""><td><message name="subRequest"></message></td></definitions>	<message name="subRequest"></message>
····	<pre><pre><pre><pre><pre><pre><pre><pre></pre></pre></pre></pre></pre></pre></pre></pre>
<pre><types> <schema <="" pre="" targetnamespace="urn:calc"></schema></types></pre>	<pre><pre><pre><pre><pre><pre><pre><pre></pre></pre></pre></pre></pre></pre></pre></pre>
	<message name="subResponse"></message>
	<pre><part name="result" type="xsd:double"></part></pre>
<message name="addRequest"></message>	<pre><porttype name="calcPortType"> <operation name="add"></operation></porttype></pre>
<pre><pre><pre><pre><pre><pre><pre>part name="a" type="xsd:double"/></pre></pre></pre></pre></pre></pre></pre>	<pre><input message="(1)"/></pre>
<pre><part name="b" type="xsd:double"></part></pre>	<pre><output message="(2)"></output></pre>
<message name="addResponse"></message>	<pre><operation name="sub"> <input message="(3)"/></operation></pre>
<pre></pre> <pre><pre><pre><pre>change name= addresponse > <pre><pre><pre><pre><pre>cpart name="result" type="xsd:double"/></pre></pre></pre></pre></pre></pre></pre></pre></pre>	<pre><mput message="(3)"></mput> <output message="(4)"></output></pre>
<pre></pre>	<pre></pre>
signature	2

```
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
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
```

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' :

signature 3

Beta bachir=new Beta(); bachir putLabel("bachir"); Beta barharan=new Beta(); barbara.putLabel("barbara"); Beta betty-new Beta(); betty.putLabel("betty"); bachir.linkBeta(barbara); bachir.linkBeta(barbara); bachir.linkBeta(barbara); barbara.linkBeta(barbara); barba	EXTED ATE 0.1	DAZED A KE 0.0	
bachir putLabel("bachir"): Beta barbara=new Beta(); barbara_putLabel("antara=new Beta(); bachir_linkBeta(barbara"); Beta betty=new Beta(); betty.putLabel("betty"); bachir_linkBeta(barbara); barbara_linkBeta(barbara); barbara_linkBeta(bachir); barbara_linkBeta(bachir); anatole_linkBeta(bachir); anatole_linkBeta(bachir); anatole_linkBeta(bachir); anatole_linkBeta(bachir); anatole_printBetaChain("four"); anatole_printBetaChain("four"); anatole_printBetaChain("four"); anatole_printBetaChain("four"); anatole_printBetaChain("four"); anatole_printBetaChain("barbara"); backir_linkAppla(maiole); aziz_printAlphaChain("barbara"); backir_linkAppla(maiole); aziz_printAlphaChain("barbara"); backir_linkAppla(maiole); aziz_printAlphaChain("barbara"); backir_linkAppla(maiole); aziz_printAlphaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("four"); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("four"); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir);	EXTRAIT n° 1	EXTRAIT n° 2	
bachir putLabel("bachir"): Beta barbara=new Beta(); barbara_putLabel("antara=new Beta(); bachir_linkBeta(barbara"); Beta betty=new Beta(); betty.putLabel("betty"); bachir_linkBeta(barbara); barbara_linkBeta(barbara); barbara_linkBeta(bachir); barbara_linkBeta(bachir); anatole_linkBeta(bachir); anatole_linkBeta(bachir); anatole_linkBeta(bachir); anatole_linkBeta(bachir); anatole_printBetaChain("four"); anatole_printBetaChain("four"); anatole_printBetaChain("four"); anatole_printBetaChain("four"); anatole_printBetaChain("four"); anatole_printBetaChain("barbara"); backir_linkAppla(maiole); aziz_printAlphaChain("barbara"); backir_linkAppla(maiole); aziz_printAlphaChain("barbara"); backir_linkAppla(maiole); aziz_printAlphaChain("barbara"); backir_linkAppla(maiole); aziz_printAlphaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("four"); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("four"); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir); anatole_printBetaChain("barbara"); barbara_linkBeta(bachir);	Reta hachir-new Reta():	AlphaInt anatole –	
Beta barbara—new Beta(); barbara putLabel("barbara"); Beta betty=new Beta(); betty.putLabel("betty"); bachir.linkBeta(barbara); barbara linkBeta(barbara); anatole.linkBeta(barbara); anatole.linkBeta(barbir); anatole.printBeta(Chain("four"); anatole.linkBeta(bachir); anatole.printBeta(Chain("four"); anatole.linkBeta(bachir); anatole.printBeta(Chain("four"); anatole.printBeta(Chai			
barbara putLabel("barbara"); Beta betty=new Beta(); betty.putLabel("betty"); bachir.linkBeta(barbara); barbara.linkBeta(barbara); barbara.linkBeta(barbir); anatole.linkBeta(barbir); anatole.linkBeta(barbir)			
Beta betty=new Beta(); betty.putLabel("betty"); bachir.linkBeta(barbara); barbara.linkBeta(betry); bachir.printBeta(Chain("bachir") EXTRAIT n° 3 Alpha anissa=new Alpha(); anissa.putLabel("aziz"); aziz.printAlpha(anisol.); anissa.putLabel("asisa"); anatole.linkAlpha(anisol.); aziz.printAlpha(anisol.); anatole.linkAlpha(anisol.); aziz.printAlpha(anisol.); anatole.linkAlpha(anisol.); aziz.printAlpha(anisol.); anatole.linkBeta(bachir); anatole.printBetaChain("four"); anatole.	v ·	,, maroto, parameter, maroto /,	
bachir. linkBeta(barbara); barbara. linkBeta(betty); barbara. linkBeta(betty); barbara. linkBeta(betty); barbara. linkBeta(betty); barbara. linkBeta(bity); anatole.linkBeta(bachir); anat		Alpha aziz=new Alpha(); aziz.putLabel("aziz"):	
bachir. Jink Beta(bachy); bachir. printBetaChain("bachir") EXTRAIT n° 3 ===================================	, , , , , , , , , , , , , , , , , , , ,		
barbara, link Beta(betty); bachit, printBetaChain("bachir") EXTRAIT n° 3 ———————————————————————————————————	bachir.linkBeta(barbara);		
EXTRAIT n° 3 Alpha anissa=new Alpha(); anissa_putLabel("anissa"); anatole.linkAlpha(anissa"); anatole.linkAlpha(anissa); aziz.printAlphaChain("sr03"); 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,&rrang); if (rang == 0) {	barbara.linkBeta(betty);		
Alpha anissa=new Alpha(); anissa.putl.abel("anissa"); anatole.linkAlpha(anissa); aziz.printAlphaChain("sr03"); 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,⟩); if (rang == 0) {	bachir.printBetaChain("bachir")		
Alpha anissa=new Alpha(); anissa_putLabel("anissa"); anatole.linkAlpha(anissa"); anatole.linkAlpha(anissa"); 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,⟩); if (rang == 0) { MPI_Send(&val, count, MPI_INT, 1, etiquette, MPI_COMM_WORLD); MPI_Recv(&valeur, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD); MPI_Recv(&valeur, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD); MPI_Recv(&valeur, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD,&statut); } MPI_Finalize(); } MPI_Finalize(); } Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.	EXTRAIT n° 3	EXTRAIT n° 4	
anatole.linkAlpha(anissa"); anatole.linkAlpha(anissa"); anatole.linkAlpha(anissa); aziz.printAlphaChain("srt03"); 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,⟩); if (rang == 0) {		========	
anatole.linkAlpha(anissa); aziz.printAlphaChain("sr03"); 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,⟩); if (rang == 0) {			
Réponse :		anatole.printBetaChain("four");	
Q13 (1 point): Q14 (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,⟩); 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); MPI_Send(&val, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD); MPI_Send(&val, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD); MPI_Finalize(); } MPI_Finalize(); } Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.			
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,⟩); 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); MPI_Send(&val, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD); MPI_Send(&val, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD); MPI_Send(&val, count, MPI_INT, 0, etiquette, MPI_COMM_WORLD); MPI_Finalize(); } MPI_Finalize(); } Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.			
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,⟩); if (rang == 0) {	keponse:	.	
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {		,	
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {			
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,⟩); if (rang == 0) {	L		
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,⟩); if (rang == 0) {	Q13 (1 point):		
Q14 (1 point): Soit le bout de code suivant: MPI_Comm_rank(MPI_COMM_WORLD,⟩); 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(); } MPI_Finalize(); } Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il? Expliquez.			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {			
MPI_Comm_rank(MPI_COMM_WORLD,⟩); if (rang == 0) {	O14 (1 maint) . Sait la hout de code quivert :		
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(); } MPI_Finalize(); } Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.			
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_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(); } MPI_Finalize(); } Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.			
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.			
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.	}		
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.	else if $(rang == 1)$ {		
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.			
MPI_Finalize(); } Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.			
Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.	}		
	MPI_Finalize();		
	}		
signature 4	Nous considérons que l'envoi de message standard est non bufferisé. Que se passe-t-il ? Expliquez.		
signature 4			
	signature	4	