Exercice 1 (12 pts) : choisir la bonne réponse :

1 point pour chaque question

1. Quel est le rôle principal d'une servlet dans une application web Java utilisant le modèle MVC ?
   - ☐ Agir comme contrôleur pour traiter les requêtes
   - ☐ Contient la base de données et la couche Model
   - ☐ Représente la couche View de l'application.

2. L'instruction `request.setAttribute("username", "Mohamed");` permet de :
   - ☐ Ajouter une variable 'username' à la session de l'utilisateur.
   - ☐ Mettre une variable 'username' dans la requête pour l'envoyer à la JSP.
   - ☐ Lire la variable 'username' du formulaire envoyé par l'utilisateur.

3. Quelle est la fonction de la méthode request.getParameter("username") dans une page JSP ?
   - ☐ Récupérer une session utilisateur et de l'afficher.
   - ☐ Récupérer un attribut stocké dans l'objet session.
   - ☐ Récupérer la valeur d'un paramètre du formulaire envoyé par l'utilisateur.

4. Quelle est la portée d'un attribut stocké avec request.setAttribute("username", "Mohamed") ?
   - ☐ Toute la requête HTTP, c'est-à-dire dans le servlet et JSP qui traitent la requête.
   - ☐ Uniquement dans la page actuelle.
   - ☐ Dans toutes les pages de la session de l'utilisateur.

5. Comment le serveur attend-il les connexions entrantes avec un ServerSocket en Java ?
   - ☐ En appelant la méthode listen().
   - ☐ En appelant la méthode accept().
   - ☐ En appelant la méthode wait().

6. Quelle méthode permet d'envoyer des données via un socket en Java ?
   - ☐ send()
   - ☐ output()
   - ☐ write()

7. Que signifie RMI en Java ?
   - ☐ Un mécanisme permettant d'appeler des méthodes d'objets distants dans Java.
   - ☐ Une bibliothèque pour l'envoi de requêtes HTTP.
   - ☐ Un protocole pour l'échange de messages entre des machines distantes.

8. Quelle méthode est utilisée pour associer un objet distant à un nom dans le registre RMI ?
   - ☐ En utilisant la méthode register() de la classe Registry.
   - ☐ En utilisant la méthode rebind() de la classe Naming.
   - ☐ La méthode link() de la classe RemoteObject.

9. Comment un client peut-il récupérer une référence à un serveur distant à partir du registre RMI
   - ☐ En utilisant la méthode getReference() de la classe UnicastRemoteObject.
   - ☐ En utilisant la méthode findServer() de la classe Registry.
   - ☐ En utilisant la méthode lookup() de la classe Naming.

10. Pour permettre à un serveur d'accepter plusieurs connexions via des sockets :
    - ☐ On utilise les Threads.
    - ☐ On exécute plusieurs serveurs.
    - ☐ Chaque client utilise un port différent.

11. Dans une application web MVC, un JavaBean est principalement utilisé pour :
    - ☐ Encapsuler les données et fournir des méthodes d'accès dans la couche modèle.
    - ☐ Traiter les requêtes http transmises par le client.
    - ☐ Générer le contenu dynamique des pages web dans la couche view.

12. À quoi sert l'outil **Postman** dans le développement d'applications web ?
    - ☐ Agir comme un client HTTP.
    - ☐ Agir comme un serveur HTTP.
    - ☐ À concevoir et héberger des bases de données.

Exercice 2 (8 pts) :

Soit l'application RMI suivante:

```
public interface interface_calcul extends Remote {
public int calculer(int a, int b)throws RemoteException;}
```

```
public class A {
public A() {
interface_calcul x ;
try {
     x =(interface_calcul)Naming.lookup("AAA");
      int r=x.calculer(10,20);
      System.out.println("le résultat= "+r);
     } catch (Exception e) {System.out.println(e);}}
public static void main(String[] args) {
new A();   } }
```

```
public class B extends UnicastRemoteObject implements interface_calcul {
public B() throws RemoteException  {  }

public int calculer (int a, int b) throws RemoteException {
     while (b != 0) { int x = a % b; a = b; b = x; } return a; }

     public static void main(String[] args) {
          try {
               Registry registre = LocateRegistry.createRegistry(1099);
               B b=new B();
               Naming.rebind("BBB", b);
          } catch (Exception e) {System.out.println(e);}   } }
```

1- Déterminer le client et le serveur : le serveur est la classe …**B**….., le client est la classe …**A**… **1**

2- Que fait ce programme : ce programme calcule le **plus grand commun diviseur (PGCD)** de deux nombres entiers positifs a et b par la méthode de Euclide avec RMI **2**

……

3- Ce programme contient une erreur qui empêche son bon fonctionnement. Trouver et corriger cette erreur ? ……

Erreur : le nom du serveur inscrit dans le registre est différent à celui demandé par le client. **1**

Solution : Dans le client on doit changer le nom du serveur. On doit écrire :

x =(interface_calcul)Naming.lookup("AAA"); au lieu de BBB **1**

……

4- Comment Modifier ce programme si on suppose que l'adresse IP du client est 192.168.50.1 et celle du serveur est 192.168.50.5 ?

Dans le client : ……..……..

```
interface_calcul calc =
 (interface_calcul) Naming.lookup("rmi://192.168.5.50:1099/BBB");
```

1

……

Dans le Serveur : ……..……..

```
Naming.rebind("rmi://192.168.5.5:1099/BBB", b);
```

1

……

5- Modifier ce programme pur pouvoir gérer plusieurs clients simultanément :

Aucune modification à faire. Avec RMI on peut gérer plusieurs clients simultanément.

1