RMI QCM

* Required

Nom Prénom *
Une seule soumission possible.

Jimmy Deweer

Question 1

Soit le fichier text.txt contenant les ligne suivantes :

Ceci est la première ligne.
Ceci est la deuxième ligne.

Soit une classe java contenant la méthode main suivante :

```
public static void main(String[] args) throws IOException {
  FileInputStream in = new FileInputStream("text.txt");
  String line = in.readLine();
  System.out.println(line);
}
```

Une option possible

- [ ] Ce programme compile et affiche "Ceci est la première ligne".
- [ ] Ce programme compile mais lance une IOException.
- [x] Ce programme ne compile pas.
- [ ] Aucune de ces 3 options.

Question 2

Quels sont les assertions vraies concernant un BufferedReader :
Deux options possibles

- [ ] Un BufferedReader permet de lire des octets par l'utilisation d'un tampon.
- [ ] Un BufferedReader permet d'écrire des octets par l'utilisation d'un tampon.
- [x] Un BufferedReader permet de lire des caractères par l'utilisation d'un tampon
- [x] Un BufferedReader implémente la méthode readLine().

Question 3

Soit la classe Server.java suivante :

```
public class Server {

   public static void main(String... args) throws Exception {
      ServerSocket serverSocket = new ServerSocket(4444);
      Socket clientSocket = serverSocket.accept();
      PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
      BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
      String inputLine;
      while ((inputLine = in.readLine()) != null) {
         if ("Requete".equals(inputLine)) {
            out.println("Reponse");
         } else {
            out.println("Pas de reponse");
         }
      }
   }
}
```

Soit la classe Client.java suivante :

```
public class Client {

   public static void main(String[] args) throws IOException {
      Socket echoSocket = new Socket("localhost", 4444);
      PrintWriter out = new PrintWriter(echoSocket.getOutputStream(), true);
      BufferedReader in = new BufferedReader(new
InputStreamReader(echoSocket.getInputStream()));
      out.println("Requete");
      System.out.println(in.readLine());
      out.println("Pas de requete");
      System.out.println(in.readLine());
      out.println("Requete");
      System.out.println(in.readLine());
   }
}
```

Une fois le server et le client lancés, dans la console du client sera affiché :
Une option possible

- [ ] Rien la classe Server.java ne compile pas
- [ ] "Requete" "Pas de requete" "Requete"
- [ ] "Requete" "Reponse" "Pas de requete" "Pas de reponse" "Requete" "Reponse"
- [x] "Reponse" "Pas de reponse" "Reponse"

Question 4

Dans le cadre de la programmation Java et plus particulièrement dans le domaine de l'objet distribué, RMI signifie :
Une option possible

- [ ] Revenu Minimum d'Insertion
- [x] Remote Method Invocation
- [ ] Remote Procedure Call
- [ ] Aucune de ces 3 options

Question 5

Soit la classe Foo.java suivante :

```
public class Foo implements java.io.Serializable {

   private String foo;
   private transient int bar;

   public int getBar() {
      return bar;
   }

   public void setBar(int bar) {
      this.bar = bar;
   }

   public String getFoo() {
      return foo;
   }

   public void setFoo(String foo) {
      this.foo = foo;
   }
}
```

Soit la classe FooSerializer.java suivante :

```
public class FooSerializer {

   public static void main(String... args) throws Exception {
      Foo foo = new Foo();
      foo.setBar(99);
      foo.setFoo("FOO");
      FileOutputStream fos = new FileOutputStream("foo.ser");
      ObjectOutputStream oos = new ObjectOutputStream(fos);
      oos.writeObject(foo);
      FileInputStream fis = new FileInputStream("foo.ser");
      ObjectInputStream ois = new ObjectInputStream(fis);
      foo = (Foo) ois.readObject();
      System.out.println(foo.getBar());
      System.out.println(foo.getFoo());
   }
```

}

Quels sont les assertions vraies concernant ces deux classes
Deux options possibles

- [x] Ces deux classes compilent
- [ ] FooSerializer lance une NotSerializableException à l'exécution, car Foo n'implémente pas Externalizable.
- [x] FooSerializer s'exécute parfaitement et affiche "0" "FOO"
- [ ] FooSerializer s'exécute parfaitement et affiche "99" "FOO"

Question 6

Une application distribuée qui utilise un registre pour obtenir une référence à un objet distant doit contenir les actions suivantes :

1. Le client cherche un objet ("lookup") par son nom dans le registre
2. Le serveur appelle le registre afin de lier ("bind") un nom avec un objet distant.
3. Une fois la référence obtenue le client appelle la méthode souhaitée.

Réordonner ces 3 actions.
Une option possible

- [ ] 1 - 2 - 3
- [x] 2 - 1 - 3
- [ ] 2 - 3 - 1
- [ ] Other:

Question 7

Dans RMI, une interface qui définit un "contrat de service" accessible depuis le réseau doit :
Deux options possibles

- [x] Étendre l'interface java.rmi.Remote
- [ ] Surcharger le constructeur par défaut d'une classe.
- [x] Avoir des méthodes dont la signature lance une java.rmi.RemoteException
- [ ] Etendre l'interface java.io.Serializable

Question 8

Soit la classe DateService.java suivante :

```
import java.rmi.Remote ;
import java.rmi.RemoteException ;
import java.util.Date ;

public interface DateService extends Remote {

   public Date getOfficialDate() throws RemoteException;
}
```

Soit l'implémentation suivante :

```
public class DateServiceRemoteImpl implements DateService {

   public Date getOfficialDate() throws RemoteException {
      return new Date();
   }
}
```

Que manque-t-il à l'implémentation DateServiceRemoteImpl
Plusieurs réponses possibles

Il manque un "extends UnicastRemoteObject" dans la déclaration de l'implémantation.

Il manque aussi les "import" nécessaires.

Et pour finir constructeur ^^

Question 9

Soit la classe suivante :

```
public class DateServerMain {

   public static void main(String... args) throws Exception {
      Registry registry = LocateRegistry.createRegistry(Registry.REGISTRY_PORT);
      registry.lookup("rmi://dateService");
      System.out.println("rmi://dateService bound ...");
   }
}
```

Quelles sont les assertions vraies concernant la classe DateServerMain
Une option possible

- [ ] Cette classe compile, crée un registre RMI et affiche "rmi://dateService bound ..."
- [ ] Cette classe ne compile pas.

- [x] Cette classe compile mais lance une java.rmi.NotBoundException à l'exécution si il n'y a pas de service "bindé" auprès du registre.
- [ ] Cette classe compile, obtient un accès au registre RMI et recherche un objet Remote qui est finalement "casté" en un DateService.

Question 10

Soit le diagramme UML suivant : http://tpsoflfo.googlecode.com/files/Selection_057.png

Quelles sont les assertions vraies concernant ce diagramme UML
Deux options possibles

- [x] BookStoreServiceRemoteImpl étend la classe UnicastRemoteObject
- [ ] Ce diagramme concerne le client vu dans les différents TPs
- [ ] BookStoreServiceRemoteImpl est Serializable
- [x] BookStoreServiceRemoteImpl implémente BookStoreService

Question 11

Soit l'interface suivante :

```
public interface BookStoreService extends Remote {
   public Collection<Book> findAllBooks() throws Exception;
}
```

Quelles sont les assertions vraies pour fournir une implémentation correcte de ce service RMI,
Deux options possibles

- [ ] L'implémentation étend l'interface Remote
- [x] Book doit être Serializable
- [x] La Collection doit être Serializable
- [ ] L'implémentation doit être une final class

Question 12

Soit la "stack" suivante :

```
Exception in thread "main" java.rmi.NotBoundException: rmi://localhost/heureDeMoscou
        at sun.rmi.registry.RegistryImpl.lookup(RegistryImpl.java:136)
```

```
        at sun.rmi.registry.RegistryImpl_Skel.dispatch(Unknown Source)
        at sun.rmi.server.UnicastServerRef.oldDispatch(UnicastServerRef.java:409)
        at sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:267)
        at sun.rmi.transport.Transport$1.run(Transport.java:177)
        at sun.rmi.transport.Transport$1.run(Transport.java:174)
        at java.security.AccessController.doPrivileged(Native Method)
        at sun.rmi.transport.Transport.serviceCall(Transport.java:173)
        at sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:553)
        at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run0(TCPTransport.java:808)
        at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(TCPTransport.java:667)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1110)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:603)
        at java.lang.Thread.run(Thread.java:722)
        at
sun.rmi.transport.StreamRemoteCall.exceptionReceivedFromServer(StreamRemoteCall.java:273)
        at sun.rmi.transport.StreamRemoteCall.executeCall(StreamRemoteCall.java:251)
        at sun.rmi.server.UnicastRef.invoke(UnicastRef.java:377)
        at sun.rmi.registry.RegistryImpl_Stub.lookup(Unknown Source)
        at com.bissy.distrib.rmi.client.DateClientMain.main(DateClientMain.java:15)
```

Quelles sont les assertions vraies

Deux options possibles

- [ ] La classe DateClientMain a lancé une Exception
- [ ] L'exception lancée est une ClassCastException.
- [x] L'exception lancée est une NotBoundException.
- [x] L'exception a été lancée lors d'un lookup sur le registre.

Question 13

Justifier votre dernière réponse

Exception in thread "main" java.rmi.NotBoundException: rmi://localhost/heureDeMoscou at sun.rmi.registry."""""""""RegistryImpl""""""."""""""""lookup""""""""(RegistryImpl.java:136)

voilà, ligne 136 de la classe RegistryImpl sur le lookup...

Submit