Java 高级程序设计 复习题目

1. 线程

1) 用实现接口Runnable 的方式，实现两个并行运行的线程。

2) 用继承Thread的方式，实现两个并行运行的线程。

[参考代码段]

```
class Thread1 extends Thread{
   public void run(){
      try{
         while(true){
            sleep(800);
            System.out.println(" ----First Thread----");
         }
      }
      catch(InterruptedException e){}
   }
}

class Thread2 extends Thread{
   public void run(){
      try{
         while(true){
            sleep(1000);
            System.out.println(" ----Second Thread----");
         }
      }
      catch(InterruptedException e){}
   }
}
```

2. 见下面的一个字符串数组：

```
    private String animals[] =
        { "Tiger", "Lion", "Cat", "Dog" , "Deer", "Chicken", "Sheep", "Horse", "Rabit" ,
"Snake"};
```

编写代码段，把它转换为 List 对象，并用 List 的 iterator()方法得到一个迭代子，通过迭代子输出所有 List 分量。

[参考答案]

```
    ArrayList list;
    list = new ArrayList( Arrays.asList( animals ));
    Iterator it=list.iterator();
    while(it.hasNext()){
      String i=(String)it.next();
```

```
        System.out.println("Element in list is :      "+i);
```

3.编写程序，获取一个域名的 IP 地址。

[参考程序] 下面程序得到 www.baidu.com 的 IP

```
import java.net.*;
import java.io.*;
class Readaddr{
   public static void main(String args[])    {
      try {
         InetAddress inetadd;
         inetadd = InetAddress.getByName("www.baidu.com");
         System. out. println(inetadd.toString() );
      }
      catch(Exception e) {
         System.out.println(e);
      }
   }
}
```

4. Socket 通讯

1) 假设 displayArea 是一个 JTextArea 对象，input 是一个封装了 Socket 的对象输入流。编写方法 receiveData()不断从 input 读数据，显示在 displayArea()中，直到读到"TERMINATE"为止。

[参考代码]

```
private void receiveData() throws IOException{
      do{
            try   {
                  message = (String) input.readObject();
                  displayArea.append(message);
                  displayArea
                              .setCaretPosition(displayArea.getText().length());
            } catch (ClassNotFoundException classNotFoundException){
                  displayArea.append("\nUnknow object type received");
            }
      } while (!message.equals("TERMINATE"));
}
```

2) 假设 displayArea 是一个 JTextArea 对象，output 是一个封装了 Socket 的对象输出流。编写方法 sendData()不断用 output 发送字符串，同时显示在 displayArea()中。

[参考代码]

```
private void sendData(String message)   {
      try   {
                  output.writeObject(message);
                  output.flush();
```

```
                displayArea.append(message);
                displayArea
                        .setCaretPosition(displayArea.getText().length());
            }
        } catch (IOException ioException)        {
            displayArea.append("\nError writing object");
        }
    }
```

5. UDP 通讯

1) 假设 destid 是接收端 IP，编写方法 sender(Message msg)，采用 UDP 方式把 msg 通过 7000 端口发送到 destid 指定的客户端的 7002 端口。

[参考代码]

```
    public sender(Message msg) {
        DatagramPacket datagramPacket;
        DatagramSocket datagramSocket;
        try {
            ByteArrayOutputStream stream = new ByteArrayOutputStream(500);
            ObjectOutputStream objectOutputStream = new ObjectOutputStream(
                    new BufferedOutputStream(stream));

            objectOutputStream.writeObject(msg);
            objectOutputStream.flush();

            byte sendBuf[] = stream.toByteArray();
            datagramPacket = new DatagramPacket(sendBuf, sendBuf.length,
InetAddress.getByName(destid),7002);
            datagramSocket = new DatagramSocket(7000);
            objectOutputStream.close();
            datagramSocket.send(datagramPacket);
            datagramSocket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

2) 编写方法 receiver()，采用 UDP 方式从 7002 端口读取数据并显示。

[参考代码]

```
    public void receiver () {
        String msg;
        DatagramPacket datagramPacket;
        DatagramSocket datagramSocket;
        byte recvBuf[] = new byte[500];
        try {
```

```
            datagramPacket = new DatagramPacket(recvBuf, recvBuf.length);
            datagramSocket = new DatagramSocket(7002);
        } catch (SocketException e) {
            e.printStackTrace();
        }
        while (true) {
            try {
                datagramSocket.receive(datagramPacket);
                ByteArrayInputStream stream = new ByteArrayInputStream(recvBuf);
                ObjectInputStream is = new ObjectInputStream(
                        new BufferedInputStream(stream));
                Object object = is.readObject();
                msg = (String) object;
                is.close();
                System.out.println(msg);
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
```

6. 消息摘要

假设 `Conversion.byteArrayToHexString(byte[] b)` 得到字节数组对应的 16 进制字符串，编写方法 `md(String f)` 以 16 进制字符串的形式返回字符串的 MD5 消息摘要。

[参考代码]

```
    public static String md(String f){
        MessageDigest md;
        BufferedInputStream file;
        String digestString;
        DigestInputStream in;
        try{
            // Open the file
            file = new BufferedInputStream(new FileInputStream(f));
            // Create an MD5 message digest
            md = MessageDigest.getInstance("MD5");
            //Filter the file as a DigestInputStream object
            in = new DigestInputStream(file,md);
            int i;
            byte[] buffer = new byte[BUFFER_SIZE];
            //Read the file and compute the digest
            do{
                i = in.read(buffer,0,BUFFER_SIZE);
            }while(i==BUFFER_SIZE);
            //Get the final digest and convert it ti a String.
            md = in.getMessageDigest();
```

```
            in.close();
            byte[] digest = md.digest();
            digestString = Conversion.byteArrayToHexString(digest);
        }catch (Exception e){
            e.printStackTrace();
        }
        return digestString;
    }
```

7. 私钥加密

1)已知 key 是一个私钥，写出用 Cipher 对明文 plainText(字节数组)进行加密的代码段。

[参考代码]

```
Cipher cipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
cipher.init(Cipher.ENCRYPT_MODE, key);
byte[] cipherText = cipher.doFinal(plainText);
```

2)已知 key 是一个私钥，写出用 Cipher 对密文 cipherText(字节数组)进行解密的代码段。

[参考代码]

```
Cipher cipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
cipher.init(Cipher.DECRYPT_MODE, key);
byte[] newPlainText = cipher.doFinal(cipherText);
```

8. 公钥加密

编写代码，利用 RSA 算法，产生一个 1024 长度的非对称密钥。

[参考代码]

```
KeyPairGenerator keyGen = KeyPairGenerator.getInstance("RSA");
keyGen.initialize(1024);
KeyPair key = keyGen.generateKeyPair();
```

9. 通讯安全

1) 写出代码段，利用 DES 算法生成一个私钥，并保存到文件 key.key 中，以便通讯双方取出。

[参考代码]

```
//生成私钥
KeyGenerator keyGen = KeyGenerator.getInstance("DES");
keyGen.init(56);
Key key = keyGen.generateKey();
System.out.println(key);
//用输出流输出到文件
ObjectOutputStream o = new ObjectOutputStream ( new FileOutputStream ( "key.key" ) );
o.writeObject(key);
o.close();
```

2)写出 sendData(String message)方法，利用对象输出流 output 实现对 message 的发送。发送前需要从 key.key 文件中取出私钥对 message 加密，加密后转换为 BASE64 码发

送。假设已存在 byteArrayToBase64String (byte[] b)方法可以对字节数组转为 BASE64 码。

[参考代码]

```
void sendData(String message) throws Exception {
    ObjectInputStream keyin = new ObjectInputStream(new FileInputStream("key.key"));
    Key key = (Key) keyin.readObject();
    byte[] plainText = message.getBytes("UTF8");
    Cipher cipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
    cipher.init(Cipher.ENCRYPT_MODE, key);
    byte[] cipherText = cipher.doFinal(plainText);
    message = byteArrayToBase64String (cipherText);//将字节数组转为字符串

    output.writeObject(message);
    output.flush();
}
```

3)编写receiveData()方法，不断用对象输入流input获取密文的BASE64字符串到message，从文件key.key取出私钥解密并显示，直到读到字符串"TERMINATE"为止。假设已存在base64StringToByteArray (String str)方法，实现BASE64字符到字节数组的转换。

[参考代码]

```
void receiveData() throws Exception {
    do {
        message = (String) input.readObject();

        ObjectInputStream keyin = new ObjectInputStream(new FileInputStream("key.key"));
        Key key = (Key) keyin.readObject();
        byte[] cipherText = base64StringToByteArray (message);
        Cipher cipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
        cipher.init(Cipher.DECRYPT_MODE, key);
        byte[] newPlainText = cipher.doFinal(cipherText);
        message=new String(newPlainText, "UTF8");
        System.out.println(message);

    } while (!message.equals("TERMINATE"));//未读到结束信息

}
```