SFDV3006 Concurrent Programming
Assignment 2: Concurrent Programming using Java threads

Problem 1: [5 marks] Client and server - using sockets and serialization

Write a multithreaded server for the following **currency converter system**.

A client connects to the *CurrencyConverterServer* and sends a request string like "10 SAR". This means 10 Saudi Rials. The server will convert this currency (SAR) and amount (10) into Omani rials and return the value of 10 SAR as 1 OMR

The server should understand the following currencies and convert them into Omani rials at the following *exchange rates*
SAR (1 OMR = 10 SAR)
USD (1 OMR = 3 USD)
GBP (1 OMR = 1.5 GBP)
AUD (1 OMR = 3 AUD)
SGD (1 OMR = 4 SGD)

The server must be multi threaded so that it can handle multiple concurrent clients. The server must handle all the above currencies and any amounts which come as request from the client, for example:

```
Client sends: 100 USD     Server replies: 300 OMR
Client sends: 1000 SAR    Server replies: 100 OMR
Client sends: 60 GBP      Server replies: 40 OMR
Client sends: 80 SGD      Server replies: 20 OMR
Client sends: 80 XRP      Server replies: ERROR:unknown currency XRP

Client sends -1 GBP       Server replies: ERROR:incorrect currency value
Client sends: one USD     Server replies: Error:incorrect request format
Client sends: one         Server replies: Error:incorrect request format
Client sends: USD         Server replies: Error:incorrect request format
```

You must have the server, client, the request handler thread and other classes that you may need. You may use a Java *collection class* to store the currencies and their exchange rates however you are not allowed to use multiple conditionals such as *if*s and *switch/case* for the currency conversion logic

To get marks both the client and the server must work correctly including error handling of incorrect values and handling multiple concurrent requests.

Problem 2: [5 marks] ***Implement the currency converter using RMI***
Same Currency Converter software as above but using RMI
The server implements the following interface

```
import java.rmi.*;
import java.rmi.server.*;
interface ICurrencyConverter extends Remote {
      //This return value of the othercurrency in OMR as a string - "20 OMR"
       //the parameter is also a string of the format "80 SGD" for example
       public String convertToOMR(String otherCurrencyAndValue)
                                                      throws RemoteException;
}
```

You need to submit the client, the server class and the remote object interface.

You do not need to multithread the RMI version since RMI will automatically multithread the server (the remote) object for you.

You may use a Java *collection class* to store the currencies and their exchange rates however you are not allowed to use multiple conditionals such as *if*s and *switch/case* for the currency conversion logic

For both the parts the servers must handle improper request formats such as "-1 USD" or "USD one" or "10" or just "USD" or "INR" and give the correct error messages back to the client as given in the usage examples (on page 1) above.

Issue date: 14 May 2011

Due date: 24 May 2011 (see also section on late submission below)

Weightage: 10%.
To get full marks for a question - your classes and programs must behave correctly in all situations and you must implement the correct number of classes, threads, monitors and programs as required

Working in groups: Students can work in groups of up to a maximum of 3.

Copying: Copying whether in part or full, from other students or other sources will strictly not be tolerated leading to a loss of 100% (for this assignment) for all groups involved.

Late submission: 20% of weightage for this assignment is deducted per day after due date.

Submission guidelines: Do a demo in the labs and submit the zip file for each of the questions by email as attachments to ***assignments@sfdv3006.tk***