SCC311 Coursework Part 1 Specification

Due: Friday Week 3 through Moodle
Marking: Your assigned Lab session Week 4
Total mark allocation: 20%

The goal of this exercise is to build a simple client-server setup that invokes a method using Java RMI. You will develop both the client and the server.

Level 1: Invocation (12%)

Build a **server** that offers this exact interface, listening on the default port 1099:

```
public AuctionItem getSpec(int itemId, int clientId) throws RemoteException;
```

This returns the details of an auctioned item that has the identifier `itemId`. The `clientId` is like a username to be used for authorisation in further versions of the coursework. The return type `AuctionItem` is for you to define, but below is a suggested structure.

```
int         itemId
String      itemTitle
String      itemDescription
```

You might want to extend this structure to add other variables, such as item condition (new / used), etc. especially in the remainder of the coursework. You need to be able to justify your decisions.

You also need to build a very simple **client** that invokes the above method on the server using RMI, and render the return values to be displayed on screen. A basic text-based client is all we ask for. You could develop a GUI if you wish, but there will be no extra credit awarded.

Tip: Do not forget to start the RMI registry when running your code.

Level 2: Encryption (8%) - *If achieved, no need to submit the interface of Level 1*

Implement logic to **encrypt communications**. You need to do this using AES keys, not passwords. To achieve this, use `SealedObject`, a mechanism that allows you to encrypt and decrypt objects.

```
public SealedObject getSpec(int itemId, SealedObject clientRequest) throws
RemoteException;
```

The modified interface (above) will take an item ID and a *sealed* (i.e. encrypted) "client request" (a class that just includes `clientId` and is `Serializable`) as its arguments, and returns a sealed server response that encapsulates an `AuctionItem`. The client needs to display the decrypted response. You will need to generate and manage your own keys; see KeyGenerator.

Marking Scheme

Level 1	
Client invocation and display	3 marks
Server interface	3 marks
Management of listing data on server	3 marks
Answer in-lab questions	3 marks
Level 2	
Key creation and use	2 marks
Create client request	3 marks
Decrypt sealed server response	3 marks