UNIVERSITY OF COLOMBO, SRI LANKA

UNIVERSITY OF COLOMBO SCHOOL OF COMPUTING

DEGREE OF BACHELOR OF INFORMATION TECHNOLOGY (EXTERNAL)

Academic Year 2016 – 3rd Year Examination – Semester 6

IT6505: Middleware Architecture

09th October, 2016
(TWO HOURS)

To be completed by the candidate

BIT Examination Index No: .....

Important Instructions:

- The duration of the paper is **2 (two) hours**.
- The medium of instruction and questions is English.
- This paper has **4 questions** and **17 pages**.
- **Answer all questions.** All questions **do not** carry similar marks.
- **Write your answers** in English using the space provided **in this question paper**.
- Do not tear off any part of this answer book.
- Under no circumstances may this book, used or unused, be removed from the Examination Hall by a candidate.
- Note that questions appear on both sides of the paper.
  If a page is not printed, please inform the supervisor immediately.

Questions Answered

Indicate by a cross (×), (e.g. [X]) the numbers of the questions answered.

	Question numbers				
To be completed by the candidate by marking a cross (×).	1	2	3	4	
To be completed by the examiners:					

Index No …….

1) (a) (i). Briefly explain **two (2)** enabling benefits of middleware to distributed system designers and programmers, providing an example for each benefit.

(04 marks)

ANSWER IN THIS BOX

Any two of the following: (+2 marks for each bullet below = 4)

- Middleware can provide high-level abstractions that make it easier to develop distributed systems. These abstractions hide some of the details of the implementation of the system. For example, RPC hides marshalling and communication code behind a procedural interface to remote procedures.
- Middleware isolates the programmer from the operating system. Programs can be written to the middleware layer and can be (more easily) ported to other machines that support the same middleware (i.e., CORBA implementations are available on a variety of machines).
- Middleware can provide some forms of transparency to the programmer automatically. For example, middleware can handle the data representation problem, converting data in messages so they are appropriate for the architecture on which the receiving process is running.

(ii). State a fundamental difference between concepts conveyed by the terms '*Distributed Computing*' and '*Network Computing*'. Briefly state why management of *Distributed Computer Systems* is comparatively challenging.

(05 marks)

ANSWER IN THIS BOX

Network computing is a generic term in computing which refers to computers or nodes working together over a network. Concerns of network computing are generally described in terms of the Open Systems Interconnection (OSI). **➔ 2 marks**

The term distributed computing, in contrast with network computing, designates a set of tightly coupled programs executing on one or more computers and coordinating their actions. **➔ 2 marks**

Distributed systems are harder to manage because we have to address issues such as independent failures, unreliable communication, and insecure communication. **➔ 1 mark**

(b) (i). Consider a distributed system consisting of four identical servers. Each of the servers is available at any given instant with a probability of **0.8**. If the system is designed so that it can be operational if any one of the four servers is operational, what is the overall system availability? Alternatively, if the system is designed such that all four servers have to be available for the entire system to be available, what is the overall system availability?

(04 marks)

ANSWER IN THIS BOX

The probability of one server being down is 0.2. ➔ **1 mark**

The probability of all 4 servers being down simultaneously is $0.2^4 = 0.0016$. ➔ **1 mark**

So the probability of at least one being available is $1 - 0.0016 = 0.9984$. ➔ **1 mark**

On the other hand, the probability of all four serves being available at the same time is $0.8^4 = 0.4096$. ➔ **1 mark**

(ii). **Heterogeneity** is one of the major non-functional requirements of a distributed system. List and briefly describe two (2) other requirements.

(04 marks)

ANSWER IN THIS BOX

Any two of the following: (+2 marks for each bullet below = 4)

Scalability : Accommodate a growing load by adding hosts.

Openness : Easily extended and modified due to well defined interfaces.

Resource Sharing : H/W, S/W and Data.

Fault--- Tolerance : Ability to function correctly even if faults occur

(c) Consider the evaluation of a distributed system on a *soft deadline* property such as: '**there is a 90% chance that a HELLO message will be delivered within 5 time units**'. Assume that the system **does not** satisfy this property. Briefly explain the most appropriate type of *failure model* that one can identify for this system based on the property specified. List two other types of *failure models* that may be generally identified in distributed systems.

(04 marks)

ANSWER IN THIS BOX

This is temporal property. ➔ **1 mark**

Suitable failure model: Timing failures: these failures appear when a temporal property of the system is violated. ➔ **1 mark**

Any two of the following: (+1 marks for each bullet below = 2)

- Byzantine failures
- Halting failures
- Fail-stop failures
- Send-omission failures
- Receive-omission failures
- Network failures

(d) Consider the statement: "*Distribution Shall be Hidden from the Users*", about designing and implementing distributed systems. Do you think it is beneficial to impose the concept denoted by this statement as a general good practice when designing and implementing distributed systems? By employing a real world example for a system (other than a financial transaction handling system) that is likely to be a distributed system, briefly explain the reasons for your answer.

(04 Marks)

ANSWER IN THIS BOX

Yes. ➔ **1 mark**

Users shall rather perceive the system as a single computing facility. ➔ **1 mark**

It is also desirable to hide the complexity of distribution as much as possible from application programmers. This hiding is referred to as transparency. If we achieve this, application programmers can develop applications for distributed systems in very much the same way as they develop applications for centralized systems. ➔ **1 mark**

Providing a real world example for the reasoning ➔ 1 mark

2) (a) Suppose you are developing a server-side enterprise application. It must support a variety of different clients including desktop browsers, mobile browsers and native mobile applications. The application might also expose an Application Program Interface (API) for 3rd parties to consume. It might also integrate with other applications via either web services or a message broker. The application handles requests (HTTP requests and messages) by executing business logic; accessing a database; exchanging messages with other systems; and returning a HTML/JSON/XML response.

(i). Outline the importance of IT Architecture for designing and implementing distributed systems. Also, briefly explain the role of *Integration Infrastructure* in a typical IT Architecture.

(04 marks)

ANSWER IN THIS BOX

IT architecture identifies the components of a given problem, shows relationships among them and defines the terminology, rules and constraints on the relation and components. ➔ **1 mark**

IT Architecture makes a conducive environment for proper integration of components. Main purpose of defining an architecture is to try to impose order of chaos or potential chaos. Architecture is an essential first step to orderly solutions to problems. ➔ **1 mark**

Integration infrastructure describes the HW and SW required to make the connections. ➔ **1 mark**

Integration infrastructure makes no specific assumptions about the physical deployment. ➔ **1 mark**

(ii). Assume that you chose to build the enterprise application described in **2) (a)**, using services that are developed and deployed independently of one another. Also assume you decided that each such service should have its own database in order to be decoupled from other services and when necessary, consistency between databases is maintained using application-level events. By employing an appropriate diagram, explain in detail which type of a generic connection (*vertical* or *horizontal*) **and** a topological connection (*Bus* or *Hub*) would you employ at the *integration infrastructure layer* to support the implementation of the services.

(06 marks)

ANSWER IN THIS BOX

Employ vertical connections (➔ **1/2 mark**) and Bus topological connections (➔ **1/2 mark**).

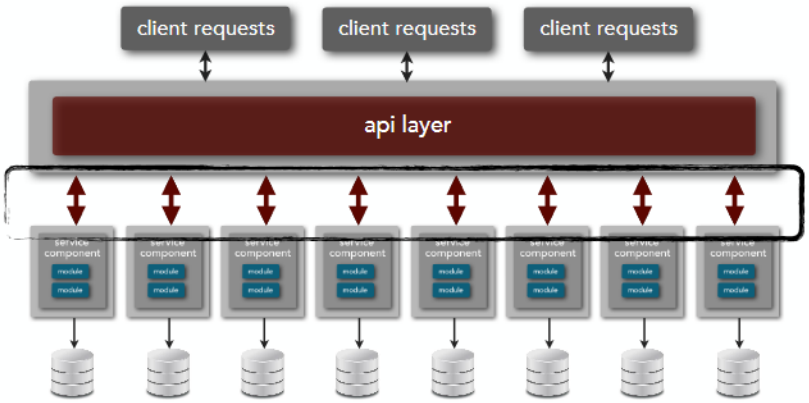
Vertical connection refers to the connection of the different layers in a multitiered architectures across multiple machines. ➔ **1 mark**

Since each service has its own database and consistency is maintained using application-level events there need not to make horizontal connections between the databases and we can go for vertical connections ➔ **1 mark**

Since Bus connection favours Vertical connections (Connecting Presentation, Application Layer and Database Layer as one silo) we use Bus connections ➔ **1 mark**

Continued…

ANSWER IN THIS BOX



Appropriate diagram ➔ **2 marks**

(iii). As described in **2) (a)**, the enterprise application will have to be integrated with other applications via a *message broker*. List **two (2)** possible actions which can be taken by a candidate *message broker* to facilitate messaging to and from the enterprise application.

(02 marks)

ANSWER IN THIS BOX

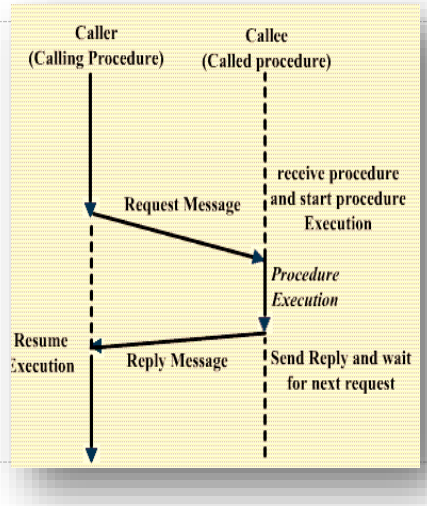
* Route messages to one or more of many destinations
* Transform messages to an alternative representation
* Perform message aggregation, decomposing messages into multiple messages and sending them to their destination, then recomposing the responses into one message to return to the user
* Invoke Web services to retrieve data
* Respond to events or errors

Any two of the above: (+1 mark for each bullet below = 2)

(b) (i). With the aid of an appropriate diagram, explain how RPCs (Remote Procedure Calls) achieve execution **transparency.**

(04 marks)

ANSWER IN THIS BOX



Appropriate diagram ➔ **2 marks**

Continued…

ANSWER IN THIS BOX

- Client sends a message to the server and server unpacks the message and performs its work
- Server return the results as a message
- When the client receives the message from server, the operating system of client sees that it is addressed to client process.
- Client has no idea that the work was done remotely instead of the local operating system.

For each bullet point above ➔ 1/2 mark ➔ total 2 marks

(ii). Consider following is the content of a RPC *Interface Definition Language (*IDL) file named *msg.x*.

```
const MAXLEN = 256;
/* msg.x: Remote msg printing protocol */
program MESSAGEPROG {
   version PRINTMESSAGEVERS {
          int PRINTMESSAGE(string) = 3;
   } = 3;
} = 20102344;
```

(A). Once complied, what would be the output artefacts generated by the **rpcgen** compiler for this IDL file.

(03 marks)

ANSWER IN THIS BOX

`rpcgen` is used to generate the header files (`msg.h`), **➔ 1 mark**

client stub (`msg_clnt.c`), and **➔ 1 mark**

server stub (`msg_svc.c`). **➔ 1 mark**

(B). State the values for the *program number*, *version number* and *procedure number* in the given IDL file. Briefly state their purpose in the execution of the relevant RPC program.

(03 marks)

ANSWER IN THIS BOX

Program number: `20102344` **➔ 1/2 mark**

Version number: 3 **➔ 1/2 mark**

Procedure number: 3 **➔ 1/2 mark**

Continued…

ANSWER IN THIS BOX

The program number identifies a group of related remote procedures, each of which has a unique procedure number. ➔ **1/2 mark**

Version numbers enable multiple versions of an RPC protocol to be available simultaneously.

➔ **1/2 mark**

Each version consists of a collection of procedures which are available to be called remotely, each of which are identified by a procedure number. ➔ **1/2 mark**

(c) Consider the statement: "*Unlike in RPCs, in Message Queuing, a receiver needs to be programmed explicitly to understand the message layout to process messages received from a sender*". Using a single sentence, explain why this statement must be true. In Amazon Simple Queue Service (SQS), how does the concept of '**Visibility Timeout**' help the receiver components process messages reliably?

(03 marks)

ANSWER IN THIS BOX

A disadvantage of message queueing is that there is no IDL and this means that there is no mrshalling, therefore, it is up to designer/developer to ensure that the sender and the receiver knows the message layout. ➔ **1 mark**

While a Message is being processed, it remains in the queue and is not returned to subsequent receive requests for the duration of the visibility timeout. ➔ **1 mark**

Immediately after the component receives the message you don't want other components in the system receiving and processing the message again. ➔ **1 mark**

3) (a) (i). Briefly explain how the concept of an *Interface Definition Language (IDL)* is employed in an object-oriented middleware such as *Common Object Request Broker Architecture* (CORBA)**.**

(02 marks)

ANSWER IN THIS BOX

- IDLs in OOM support the concept of object types as parameters; failure handling; and inheritance.
- IDLs separates object interfaces from their implementations.
- IDLs definitions focus on object interfaces, the operations supported by those interfaces, and exceptions that may be raised by operations.
- IDLs should support OOM's operations to be called through an interpretative interface such as a macro language.
- IDLs should support to define reflexivity in OOM.

Any two of the above: (+1 mark for each bullet below = 2)

(ii). Consider a hypothetical banking system designed to manage bank accounts. A user of this system wishes to make deposits and withdrawals. An account also needs to hold the balance of the account and the name of the account's owner. Write a simple **CORBA IDL code segment** to fulfil the requirements of this system. Define a module, an interface, attributes and relevant operations with an example of failure handling.

(08 marks)

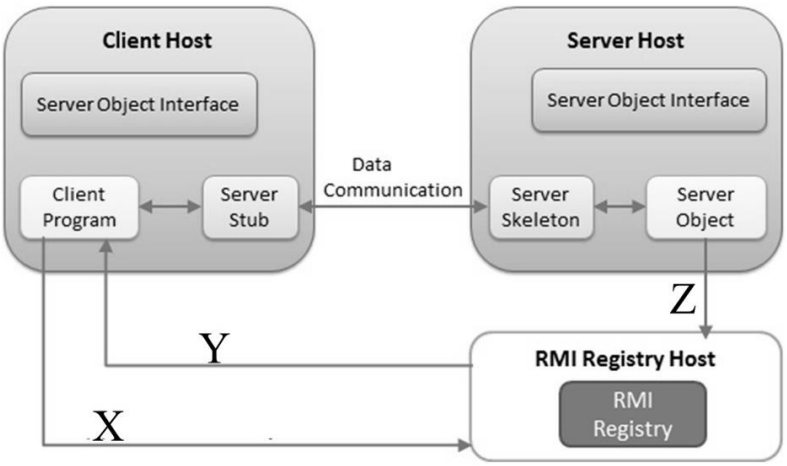
ANSWER IN THIS BOX

```
// IDL
  module Finance {➔ 1 mark
      interface Account {➔ 1 mark
            // Attributes to hold the balance and the name
            // of the account's owner.
            attribute float balance; ➔ 1 mark
            readonly attribute string owner; ➔ 1 mark
            exception SystemDown {➔ 1 mark
                  string reason;
            };
            // The operations defined on the interface.
            void makeDeposit(in float amount,
                             out float newBalance) raises
(SystemDown); ➔ 1 mark (for the operation)
            void makeWithdrawal(in float amount,
                             out float newBalance) raises
(SystemDown); ➔ 2 mark (1 for exception handling, 1 the operation)
      };
 };
```

Continued…

ANSWER IN THIS BOX

(b) (i). The following diagram illustrates the Java Remote Method Invocation (RMI) architecture.



According to the order of events that occur in a typical RMI application, **name** the events X, Y and Z in the above illustration and write the **order** of invocation by assigning them numbers chosen from 1 through 3 where number 1 denotes the first invocation and number 3 denotes the last. Also, briefly describe events X, Y and Z. Write your answers in the appropriate spaces provided below.

(06 marks)

ANSWER IN THIS BOX

X – Look for Server Object ➔ **1/2 mark**, Number 2 ➔ **1/2 mark**

Y – Return Server Stub ➔ **1/2 mark**, Number 3 ➔ **1/2 mark**

Z – Register Server Object ➔ **1/2 mark**, Number 1 ➔ **1/2 mark**

X - Clients use the name to find server objects. ➔ **1 mark**

Y – Once found, clients can obtain a remote reference to those objects from the RMI Registry. ➔ **1 mark**

Z - Servers name and register their objects to be accessed remotely with the RMI Registry. ➔ **1 mark**

(ii). Consider the code segment given below of a Java Interface which is going to be used to instantiate a remote server object that is to be deployed as part of a RMI application.

```
package com.ucsc.exam;
import java.rmi.RemoteException;

public interface Calculator{

     public long add(long a, long b) throws RemoteException;
     public long sub(long a, long b) throws RemoteException;
     public long mul(long a, long b) throws RemoteException;
     public long div(long a, long b) throws RemoteException;
}
```

When the above code segment is compiled with the other relevant source files, a **compile error** is generated. Assuming the other source files are free of any compile errors, identify in the code segment, the cause of the compile error. Correct and rewrite the erroneous line(s) of code so that the code segment is compiled without any compile errors. With respect to RMI architecture, identify the role of the above Java Interface: *Calculator*.

(03 marks)

ANSWER IN THIS BOX

Compile Error: interface has not extended java.rmi.Remote. ➔ **1 mark**

```
public interface Calculator extends java.rmi.Remote {
```
➔ **1 mark**

Calculator is a ***remote interface*** ➔ **1 mark**

(iii). Briefly explain why there are different semantics in Java RMI for passing remote and non-remote objects in performing *remote method invocation*.

(03 marks)

ANSWER IN THIS BOX

First, a remote object that is passed to a method on some other machine has the capability of being accessed remotely and therefore it can be passed by reference; the semantics of parameter-passing can be the same as in the case of a local method invocation. ➔ **1.5 marks**

Secondly, non-remote objects are, by definition, not accessible from remote objects. Hence, it would not be possible to pass non-remote objects by reference to a method that might be executed on a different machine; that method could never invoke any operations from the object passed. ➔ **1.5 marks**

Continued…

ANSWER IN THIS BOX

(c) With reference to Enterprise JavaBeans (EJBs), briefly state **two (2)** major differences between stateless and stateful session beans.

(03 marks)

ANSWER IN THIS BOX

Stateful session beans encapsulate business logic and state specific to a client. ➔ **1 mark**

Stateful beans are called "stateful" because they maintain conversational state between method invocations. ➔ **1/2 mark**

Stateless session beans are made up of business methods that behave like functions: they operate only on the arguments passed to them when they are invoked (but can lookup state in a database or file). ➔ **1 mark**

Stateless beans are called "stateless" because they are transient - they do not maintain a conversational state between method invocations. ➔ **1/2 mark**

4) (a) (i). Briefly explain in your own words, the evolution of middleware from RPC to Web services along the path: RPC ➔ CORBA ➔ EJB ➔ Web services, while commenting on the significant **improvements** you perceived/understood to have happened along this path.

(04 marks)

ANSWER IN THIS BOX

- RPC are neither reflexive nor did they deal with the concept of OO. ➔ **1 mark**
- CORBA was an OOM, support reflexivity and due to its IDL, CORBA can be used in a language agnostic way. ➔ **1 mark**
- However, programmers of CORBA had to handle non-functional requirements such as security, transactions, etc, explicitly by themselves. With EJB, the container provided services to handle these requirements by its own. ➔ **1 mark**
- Web services on the other hand are designed to be agnostic of the underlying device technology, operating platform, programming language to support small, medium and very large scale enterprise applications. Concepts of IDLs were replaced by WSDL, SOAP and UDDI. ➔ **1 mark**

(ii). Consider the following content of a Web Service Definition Language (WSDL) file:

```
<?xml version="1.0"?>
<definitions name="StockQuote"
             targetNamespace="http://example.com/stockquote.wsdl"
             xmlns:tns="http://example.com/stockquote.wsdl"
             xmlns:xsd1="http://example.com/stockquote.xsd"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns="http://schemas.xmlsoap.org/wsdl/">

  <types>
    <schema targetNamespace="http://example.com/stockquote.xsd"
            xmlns="http://www.w3.org/2000/10/XMLSchema">
      <element name="TradePriceRequest">
        <complexType>
          <all>
            <element name="tickerSymbol" type="string"/>
          </all>
        </complexType>
      </element>
      <element name="TradePrice">
         <complexType>
           <all>
             <element name="price" type="float"/>
           </all>
         </complexType>
      </element>
    </schema>
  </types>

  <message name="GetLastTradePriceInput">
    <part name="body" element="xsd1:TradePriceRequest"/>
  </message>
  <message name="GetLastTradePriceOutput">
    <part name="body" element="xsd1:TradePrice"/>
  </message>
```

Continued…

```
  <portType name="StockQuotePortType">
①  <operation name="GetLastTradePrice">
      <input message="tns:GetLastTradePriceInput"/>
      <output message="tns:GetLastTradePriceOutput"/>
    </operation>
  </portType>

  <binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
    <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetLastTradePrice">
      <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </binding>

  <service name="StockQuoteService">
    <documentation>My first service</documentation>
    <port name="StockQuotePort" binding="tns:StockQuoteSoapBinding">
      <soap:address location="http://example.com/stockquote"/>
    </port>
  </service>
</definitions>
```

(A). What is the **service name**, **URL**, **binding protocol** and **allowed operations** of the Web service defined in the above WSDL file?

(02 marks)

ANSWER IN THIS BOX
Service Name: `StockQuoteService` **➔ 1/2 mark**
URL: `http://example.com/stockquote` **➔ 1/2 mark**
Binding protocol: **soap ➔ 1/2 mark**
Allowed operations: `GetLastTradePrice` **➔ 1/2 mark**

(B). Briefly describe the use of <*portType*> element in **this** WSDL document. What is the relationship between the <*binding*> element and the <*portType*> element?

(02 marks)

ANSWER IN THIS BOX
The <portType> element combines multiple message elements to form a complete one-way or round-trip operation. Here a complete round-trip operation `GetLastTradePrice is` composed using an input and an output message. **➔ 1 mark**

Continued…

ANSWER IN THIS BOX

The <binding> element provides specific details on how a portType operation will actually be transmitted over the wire. ➔ **1 mark**

(C). Using the namespace `xmlns:stoc="http://example.com/stockquote"`, complete the Simple Object Access Protocol (SOAP) Body part of a possible SOAP message communicated over the network related to the **response** returned from the Web service defined in the above WSDL file, for a web service call made to execute the operation "`GetLastTradePrice`" defined at ①.

(04 marks)

ANSWER IN THIS BOX

<soapenv:Body>

```
    <stoc:TradePrice ➔ 1 mark
xmlns:stoc="http://example.com/stockquote" ➔ 1 mark >

    <stoc:price>3.75</stoc:price> ➔ 1 mark

    </stoc:TradePrice>  ➔ 1 mark
```

</soapenv:Body>

(b) (i). Briefly describe a **disadvantage** of having *statelessness* in Representational State Transfer (REST)-ful Web services.

(02 marks)

ANSWER IN THIS BOX

Web services need to get extra information in each request. ➔ **1 mark**

They also need to interpret these information to get the client's state in case client interactions are to be taken care of. ➔ **1 mark**

(ii). Your boss wants you to write a Web service that automatically calculates the VAT (value added tax – 11%) value for any restaurant **net** (*without* VAT) bill value and subsequently return the **total/gross** (*net + tax*) bill value. Consider the following piece of Java API for RESTful Web Services (JAX-RS) code segment you wrote:

```
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.QueryParam;
import javax.ws.rs.core.Response;

① @Path("/genbill")

public class VATService {

② @GET

   @Path("/data")

③ ……

④ public Response calGrossValue(@QueryParam("nval") float netVal,
@QueryParam("cusid") String customerId) {

            float tempGross = netVal;

⑤           tempGross *= 1.11;

            String result = "{\"customer\":\"" + customerId + "\",
\"Total\":" + tempGross + "}";

            return Response.status(200).entity(result).build();
      }
}
```

(A). Explain the function of the lines of code ①, ②, ④ and ⑤. What code segment can be inserted into the empty line ③ so that the **response** returned will be of content type '*application/json*'?

(05 marks)

ANSWER IN THIS BOX

① specifies the relative path for the web service resource class `VATService` ➔ **1 mark**

② Identifies which HTTP method is used for the operation: **calGrossValue** ➔ **1 mark**

④ Web service resource method with two @QueryParam parameter annotations to access to specific parameter URI query string and with a return type of a Response object used to customize the HTTP response. ➔ **1 mark**

⑤ Calculation of total bill value: `tempGross = tempGross + tempGross` *11/100➔ **1 mark**

③ `@Produces("application/json")` ➔ **1 mark**

Continued…

ANSWER IN THIS BOX

(B). Assume that the Web service written using the code segment in **4) (b) (ii)** belongs to an enterprise application that contains a suite of many other Web services. If all such Web services are deployed at the base URL '**http://ws.exam.ucsc.lk/rest'**, provide a complete URL that a client application may produce to retrieve a **total/gross** bill value (*inclusive of* VAT) for a **net** bill value of **Rs. 24,500** for a customer with id: '**CS0023**'. What modifications need to be done to the code segment and to the complete URL you provided, if you were only allowed to use the JAX-RS parameter annotation: '@PathParam' (but **not** '@QueryParam')?

(04 marks)

ANSWER IN THIS BOX

http://ws.exam.ucsc.lk/rest/genbill/data?nval=24500&cusid=CS0023➔ 1 mark

```
    @GET
    @Path("/data/{nval}/{cusid}")➔ 1 mark
    @Produces("application/json")

    public Response calGrossValue(@PathParam("nval")➔ 1/2 mark
float netVal, @PathParam("cusid")➔ 1/2 mark String customerId) {
```

http://ws.exam.ucsc.lk/rest/genbill/data/24500/CS0023 ➔ 1 mark

(c) "JAX-RS services can either be **singletons** or **per-request** objects." Briefly explain the difference between JAX-RS services designated as **singletons** and JAX-RS services designated as with **per-request** objects.

(02 marks)

ANSWER IN THIS BOX

A singleton means that one and only one Java object services HTTP requests➔ **1 mark**.
Per-request means that a Java object is created to process each incoming request and is thrown away at the end of that request. ➔ **1 mark**

*****