Final Exam
Advanced Java - CIS 35B
Answer all questions

Question 1
a. Explain the term session in server side programming approach. What is the main benefit from implementing the session.

Session refers to an object created in the webserver that can be pinned for storing state of http transactions.

b. You have two servlets and JSP:
Main.java,
Route1.jsp
Route2.java.
Write code to do the following:
Main.java received a "Get" request.
Main.java transfers the control to Route1.jsp.
Route1.jsp forwards the request to Route2.java.
Route2.java prints "Finally There" in response.

```
public class Main extends HttpServlet
{
	public void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException
	{
		RequestDispatcher dispatcher = request.getRequestDispatcher(Route1.jsp);
		dispatcher.forward(request, response);
	}
}
```

```
Route1.jsp
<%
	RequestDispatcher
	dispatcher=getServletContext().getRequestDispatcher(Route2.jsp);
	dispatcher.forward(request,response);
%>
```

```
Route2.java:
public class Route2 extends HttpServlet
{
	public void doGet(HttpServletRequest req, HttpServletResponse res) throws
	IOException, ServletException
	{
		res.setContentType("text/plain");
		PrintWriter out = res.getWriter();
		out.println("Finally there: ");
	}
}
```

Question 2
Correct all errors in the following code snippet:

```
package questions.c5;
public class Debug5_5 extends Thread {
   private int samples;
   private double average;
   public Debug5_5( int s ) {
      samples = s;
   }
   public synchronized void run() {
      java.util.Random r = new java.util.Random();
      double sum = 0.0;
      for( int i=0; i < samples; i++ ) {
         sum += r.nextDouble();
      }
      average = sum/samples;
   }
   public double getAverage() {
      while ( average == 0.0 ) try {
        wait();
      } catch ( InterruptedException ix ) {
        System.out.println( ix );
      }
      return average;
   }
   public static void main( String[] args ) {
      Debug5_5 x = new Debug5_5( 500000 );
      x.start();
      System.out.println( "Average = "
                      + x.getAverage() );
   }
}
```

No Syntax errors. Since run method is synchronized when x starts and main tries to call x.getAverage it hangs on the call to read the average value is run is still computing.
This problem may not occur if samples value is small. Problem can be fixed by using wait and notify.

Question 3:
Correct all errors in the following program so that it outputs a message specifying the length of the line read for each line in the input file

```
package questions.c3;
import java.io.*;
public class Debug3_2 {
   public static void main( String[] args )
         throws IOException {
      if ( args.length >= 2 ) {
         Reader  input;
         Writer output;
         String inputLine;
         input = new FileReader( args[0] );
         output = new FileWriter( args[1] );
         inputLine = input.readLine();
         try {
```

```
                while ( inputLine != null ) {
                   output.println( "Line length = "
                                   + inputLine.length() );
                   inputLine = input.readLine();
                }
             }
             catch( IOException iox ) {
                System.out.println( iox );
             }
             finally {
                input.close();
                output.close();
             }
          } else {
             System.err.println( "Usage is <input_file> "
                                 + "<output_file>" );
          }
       }
    }
```

API for reading/writing data is misplaced in general. No significant errors. Corrected code below:

```
import java.io.*;
public class Debug3_2 {
  public static void main( String[] args )
    {
       if ( args.length >= 2 ) {
       BufferedReader  input;
       FileWriter output;
       String inputLine;
       try {
              input = new BufferedReader(new FileReader( args[0] ));
              output = new FileWriter( args[1] );
              inputLine = input.readLine();
              while ( inputLine != null ) {
                     output.write( "Line length = "
                            + inputLine.length() );
                     inputLine = input.readLine();
              }
       }
       catch( IOException iox ) {
              System.out.println( iox );
       }
       finally {
              input.close();
              output.close();
       }
       } else {
       System.err.println( "Usage is <input_file> "
                           + "<output_file>" );
       }
  }
}
```

Question 4:

What tasks are involved in initializing a servlet?

Write code snippet (parts of servlet) to demonstrate how servlets are initialized.
Servlets do not have constructor and are initialized the first time a servlet is loaded using the init() method. A ServletConfig object can be passed into the init method that allows values to be read from a properties file.
Here is a typical example of using init method:

```
public void init(ServletConfig config)
{
	msg = "Hello from Java servlets!";
}
```

In the example above an instance variable msg has been initialized.

Question 5

List all circumstances in which a session object is no longer usable by Servlets and JSP's.

1. Session object is expired after the user has either logged out and a reference to session object has not been obtained by a servlet or jsp
2. Session time out has reached.
3. Session.invalidate has been called on the session object.

Question 6

Create a pair of classes called ImmediateMessage and ImmediateMessageServer/ Objects of the ImmediateMessageServer class should listen on specified port for a connection from other systems and then diplay the information received on the connection. Objects of the ImmediateMessage class will connect to these server objects and send a single message to be displayed at the server system.

```
class ImmediateMessageServer {
	private ServerSocket s1;
	Socket s1;
	PSocket ss1;
	ImmediateMessageServer() {
		ss1 = new ServerSocket(2234);
		ss1 = new PSocket();
	}
	void runServer()
	{
		while(true)
		{
			s1 = s1.accept()
			ss1.SetPSocketStreams(s1);
			ss1.handleConnection();
		}
	}
}

class PSocket {
	private PrintWriter out;
```

```
	private BufferedReader in

	void SetPSocketStreams(Socket s1) {
		try {
			out = new PrintWriter(clientSocket.getOutputStream(), true);
			in = new BufferedReader(new InputStreamReader(
				s1.getInputStream()));
		}
		catch (IOException e){

		}
	}
	void handleConnection {
		  String strInput = "";
		  try {
			while ( (strInput = in.readLine()) != null)
				System.out.println(strInput);
		  }
		  catch (IOException e){

		}
	}
}

class ImmediateMessage {
	private BufferedReader reader;
	private BufferedWriter writer;
	private Socket sock;
	private String strHost;
	private int iPort;

	ImmediateMessage (String strHost, int iPort){
		this.strHost = strHost;
		this.iPort = iPort;
		reader = new BufferedReader
		(new InputStreamReader(sock.getInputStream()));
		writer = new BufferedWriter
		(new OutputStreamWriter (sock.getOutputStream()));
	}

	public void sendOutput(String strOutput){
		try {
			writer.write(strOutput, 0, strOutput.length());
		}
		 catch (IOException e){                }
	  }
	}

public class StartServer {
	public static void main(String [] args)
	{
		ImmediateMessageServer a1 = new ImmediateMessageServer();
		a1.runServer();
```

```
	}
}

public class StartClient{
	public static void main(String [] args)
	{
		ImmediateMessage a1 = new ImmediateMessage (“127.0.0.1”, 2234);
		a1.sendOutput(“Server - Client “ + arg[0] + ”  talking”);
	}
}
```

Question 7

a. What is the significance of declaring a variable in following way in JSP?

<%! int x = 5 %>

x is an instance variable in a servlet created by the JSP. The value of x changes for all requests accessing it. A variable like this might be used to track the # of times a page has been accessed since start.

b.

Under what circumstance should you consider declaring method or code blocks synchronized?

Synchronization should be used to avoid data corruption in data for shared resources. It can also be used where the code block should only be executed by one thread at a time.