Testkonzept

Prinzipiell werde ich innerhalb der Testserien zwei Konzepte verwenden.

1. **eine Art inkrementelles Testmodell**
2. **der Einsatz von JUnit um die statischen Elemente des Programms zu testen.**

Zu 1:

Da das Programm in 4 bis 5 Stufen programmiert werden soll, ist es möglich nach jeder Scheibe einige Testläufe durchzuführen. Wie schon in der Aufgabe vermerkt müssen alle Tests, der vorherigen Programme, immer wieder das gleiche Ergebnis liefern. Darunter fallen besonders die dynamischen Teile des Programms. Diese Vergehensweise beschreibt eine Art inkrementelles Testmodell.

Das Testkonzept für den dynamischen Teil soll folgende Gestalt besitzen. Es müssen als erstes immer, für mehrere Benutzer, die Passwörter und Namen getestet werden, wobei sowohl eine erfolgreiche Anmeldung als auch eine falsche Anmeldung getestet werden soll. Dabei muss der Ablauf der Eingabe sowie die Weiterleitung an die einzelnen Punkte des Programms immer wieder auf dieselbe Weise geschehen, bzw. dieselbe Ausgabe erfolgen. Letzteres bedeutet, dass die dynamisch erzeugten HTML-Seiten, welche innerhalb der Servlets generiert werden, einen gewissen Muster folgen müssen. Etwa in der Form

<table>
<tr><td colspan="2">Symbol/Menüleiste</td></tr>
<tr><td>Navigationselemente</td><td>Ausgabe</td></tr>
</table>

Wenn diese Form nicht in allen Formularen eingehalten wird, kann definitiv auf einen Fehler geschlossen werden. Dieses Testkonzept ist, mit unterschiedlichen Benutzern, auf jeden Fall immer anzuwenden, da auf diese Weise, einfach überprüft werden kann, ob in der Ausgabe der Servlets Fehler vorliegen. Diese oben genannte Abfolge kann auf alle folgenden Methode ähnlich angewandet werden.

/F10/Geschäftsprozess: Authentifizierung
/F20/Geschäftsprozess: Anmeldung zu Übungsgruppe
/F30/Geschäftsprozess: Bereitstellung der Übungsblätter
/F50/Geschäftsprozess: Punkte anzeigen
/F60/Geschäftsprozess: Punkteliste verändern
/F110/Geschäftsprozess: Allgemeine Informationen einsehen

Zum Beispiel kann man mit einem Test—PDF, das Hochladen und anschließendes Herunterladen überprüfen. Der Test dieses Teil des Systems, funktioniert dann, wenn die PDF danach noch im Originalzustand vorliegt.
/F40/Geschäftsprozess: Upload der gelösten Aufgaben
/F130/Geschäftsprozess: Download der Lösungen

Zu 2.
Das Projekt besteht nun aus relativ wenigen statischen Elementen, die durch JUnit überprüfbar wären. Aber alle Teile, welche in einer Klassenform vorliegen, können auf ihre Ausgaben, durch ein automatisches Tool überprüft werden.
Ein Beispiel hierfür wäre, zum Beispiel, die Berechnung des Durchschnitts der erreichten Punkte im Seminar. Diese Aufgabe wird sicherlich eng mit der Klasse, die für das Auslesen der XML—Datei verantwortlich ist, erfolgen. So eine Rückgabe von errechneten Durchschnitten aus den Punkten, kann durch die Methode ***assertEquals(Objekt, Objekt)*** oder ähnlichen sehr gut überprüft werden.
Außerdem können hier auch gleich alle Arten von Tests, in einer Testklasse gesammelt werden und nach jeder Erweiterung des Programms automatisch ausgeführt werden. Andere Elemente des Programms, die hierfür geeignet sind, wären

/F70/Geschäftsprozess: Abgabetermin festlegen
/F80/Geschäftsprozess: Moderation des Diskussionsforums
/F90/Geschäftsprozess: Zugriff des Diskussionsforums
/F100/Geschäftsprozess: Klausureinschreibung
/F120/Geschäftsprozess: Festlegung der Übungsgruppen

Bei der Klausureinschreibung kann zum Beispiel geprüft werden, ob die Einschreibung innerhalb der Datenbank, bzw. der XML—Datei, welche diese Daten speichert, auch richtig und erfolgreich gesichert worden ist.

Bis jetzt scheinen nicht mehr, als diese zwei Konzepte, erforderlich zu sein, um das Programm komplett zu testen.