Universität Mannheim
Fakultät für Mathematik und Informatik
Dr. Heinz Kredel

Teilprüfung
Software- und Internettechnologie
Programmierkurs 1
Sommersemester 2005

Name: .....
Vorname: .....
Matrikel-Nr.: .....
Studienfach: .....

Hinweise:

1. Überprüfen Sie die Klausur auf Vollständigkeit (**11** einseitig bedruckte Seiten).
2. Tragen Sie die Lösungen direkt in die Klausur ein. Benutzen Sie ggf. auch die Rückseiten der Aufgabenblätter.
3. Unterschreiben Sie die Klausur auf dem letzten Blatt.
4. Zugelassene Hilfsmittel: nicht programmierbarer Taschenrechner
5. Die Bearbeitungszeit beträgt 66 Minuten.

Aufgabe	max. Punktzahl	erreichte Punktzahl
1	12	
2	16	
3	12	
4	10	
5	16	
Summe	66	

Aufgabe 1: Verständnisfragen [12 Punkte]

a) [2 Punkte] Erläutern Sie **kurz** den Zusammenhang zwischen Wohlgeformtheit und Gültigkeit von XHTML-Dokumenten.

b) [2 Punkte] In welcher Hintergrundfarbe und mit welcher Schriftgröße wird das im folgenden HTML/CSS Codefragment formatierte Wort `Test` ausgegeben?

```
<style>
  body {background-color : blue; font-size : 12pt}
  p {font-size : 14pt}
</style>
</head>
<body>
 <p>Test</p>
</body>
```

c) [2 Punkte] Geben Sie eine CSS2 style-rule an, um genau alle bereits besuchten Hyperlinks in blauer Schrift darzustellen.

d) [3 Punkte] Korrigieren Sie die Fehler in folgender Java-Methode, die alle Einträge des übergebenen Felds `a` aufsummieren soll:

```
public int sumArray(int[] a){
  for (int i=1 ; i <= a.length ; i++){
    int sum+=a[i];
  }
  return sum;
}
```

e) [3 Punkte] Was gibt das folgende Java-Programm aus (mit kurzer Begründung)?

```
public class Minuten{

    public static void main(String args[]){
        int minuten = 0;
        for ( int ms = 0; ms < 60*60*1000; ms++ ) {
            if ( ms % 60*1000 == 0 ) {
                minuten++;
            } /* if */
        } /* for */
        System.out.println(minuten);
    } /* main */

} /* Minuten */
```

Aufgabe 2: XHTML und CSS [16 Punkte]

a) [8 Punkte] Schreiben Sie eine gültige *XHTML 1.0 Strict* Seite, die das Weihnachtsessen der Mensa ankündigt. Die Seite soll die Überschrift `Weihnachtsesssen` sowie eine Tabelle enthalten, in der

- Fasanensuppe als Vorspeise,
- Filetspitzen mit Röstiecken als Hauptgericht und
- Herrencreme als Nachspeise

aufgeführt sind.

Formatieren Sie die Seite so, dass

- die Überschrift kursiv und in einer Schriftart der Familie Helvetica und
- alle Tabelleneinträge zentriert in den Tabellenzellen dargestellt werden.

Fügen Sie Ihren Code in die folgende Datei ein:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">

<head>
 <meta http-equiv="Content-Type"
       content="text/html; charset=iso-8859-1" />
 <title>Weihnachtsessen 2005</title>
```

b) [8 Punkte] Schreiben Sie eine gültige *XHTML 1.0 Strict* Formularseite, mit der man das Weihnachtsessen bewerten kann. Das Formular soll

- über Radiobuttons abfragen, ob der Benutzer das Essen gut oder schlecht fand,
- eine Textarea mit der Bezeichnung `Kommentar`, die drei Zeichen hoch und 40 Zeichen breit ist, und
- einen mit `Abschicken` beschrifteten Submit-Button enthalten
- sowie die Formulardaten mit `GET` an das CGI-Programm `http://www.uni-mannheim.de/cgi-bin/mensa.cgi` übermitteln.

Fügen Sie Ihren Code in die folgende Datei ein:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">

<head>
 <meta http-equiv="Content-Type"
       content="text/html; charset=iso-8859-1" />
 <title>Bewertung Weihnachtsessen 2005</title>
```

Aufgabe 3: Java-Programmierung [12 Punkte]

Schreiben Sie eine **vollständige** Java-Application `Distinct`, der über die Kommandozeile ganze Zahlen aus dem Intervall $[-100, 100]$ übergeben werden. Das Programm soll herausfinden, ob alle eingegebenen Zahlen verschieden sind und ggf. die kleinste mehrfach vorkommende Zahl ausgeben.

Beispiel:

```
> java Distinct 10 15 -3 15 10
Die kleinste mehrfach vorkommende Zahl ist 10.
```

Sie können davon ausgehen, dass die übergebenen Zahlen korrekt formatiert sind.

Aufgabe 4: Objektorientierte Grundlagen [10 Punkte]

a) [3 Punkte] Worin unterscheiden sich Klassen- und Instanzvariablen?

b) [2 Punkte] Betrachten Sie das Interface `Koerper` und die Klasse `GanzeZahlen`, die wie folgt definiert sind:

```
public interface Koerper{

  Object getAddInverse(Object o);
  Object getMultInverse(Object o);

}

public class GanzeZahlen implements Koerper{

  Object getAddInverse(Object o){
    return new Integer(-((Integer)o).intValue());
  }

}
```

Wird die Klasse `GanzeZahlen` übersetzt (mit Begründung)?

c) [2 Punkte] **Überladen** Sie in der folgenden Klasse `X` die Methode `quotient` für den Datentyp `double`.

```
public class X{

 int quotient(int a, int b) { return a/b; }









}
```

d) [3 Punkte] **Überschreiben** Sie in einer Subklasse von `X` die Methode `quotient(int a, int b)`, indem Sie den Quotienten berechnen als `(int)Math.floor(a/b)`.

Aufgabe 5: Objektorientierung in Java [16 Punkte]

Gegeben seien ein Interface `Pruefung` und eine Klasse `Student`, die wie folgt definiert sind:

```
public interface Pruefung{

    /* gibt die ECTS-Punkte der Pruefung zurueck */
    public int getECTSPunkte();

    /* gibt die in der Pruefung erzielte Note zurueck */
    public double getNote();

} /* Pruefung */

public abstract class Student{

    /* Name des Studenten */
    private String name;

    /* aktuelle Semesterzahl */
    private int anzSemester;

    /* Pruefungen, an denen der Student teilgenommen hat */
    protected java.util.LinkedList pruefungen;

    public Student(String name) { ... }

    public void pruefungAbgelegt(Pruefung p){ ... }

    public int getAnzSemester(){ return anzSemester; }

    /* gibt genau dann true zurueck, wenn der Student seinen Pruefungsanspruch
       verloren hat
    */
    public abstract boolean prfAnspruchVerloren();

} /* Student */
```

Folgende Methoden sind eventuell für die Lösung der Aufgabe hilfreich:

Klasse `java.util.LinkedList`:

`int size()`	Returns the number of elements in this list.
`boolean add()`	Appends the specified element to the end of this list.
`Object get(int index)`	Returns the element at the specified position in this list, where the index of the first element is 0.
`java.util.Iterator iterator()`	Returns an iterator over the elements in this list (in proper sequence).

Interface `java.util.Iterator`:

`boolean hasNext()`	Returns true if the iteration has more elements.
`Object next()`	Returns the next element in the iteration.

a) [4 Punkte] Implementieren Sie den Konstruktor der Klasse `Student`, indem Sie die Attribute der Klasse geeignet initialisieren. Falls `null` als Name übergeben wird, soll der Konstruktor eine `NullPointerException` werfen.

b) [2 Punkte] Was passiert bei der Ausführung des folgenden Java-Codefragments?

```
Student s = new Student("Schulz, Klaus");
```

c) [2 Punkte] Geben Sie eine Implementation der Methode `pruefungAbgelegt` an, mit der eine Prüfung, an der der Student teilgenommen hat, der Liste seiner abgelegten Prüfungen hinzugefügt werden kann.

d) [8 Punkte] Schreiben Sie eine `Student`-Subklasse `SIT_Student` mit den folgenden Eigenschaften:

- Eine Methode `getECTSPunkte()` berechnet die Summe der ECTS-Punkte aus allen **bestandenen**, d.h. mit mindestens $4,0$ bewerteten Prüfungen des Studenten.
- Ein SIT-Student verliert seinen Prüfungsanspruch genau dann, wenn er bis zum Ende des zweiten Semesters nicht wenigstens 30 ECTS Punkte bzw. bis zum Ende des vierten Semesters nicht wenigstens 88 ECTS-Punkte aus bestandenen Prüfungen nachweisen kann.