Prof. Dr.-Ing. J. Hornegger
Lehrstuhl für Mustererkennung
Universität Erlangen-Nürnberg

19. Februar 2007

Schriftliche Prüfung zu der Vorlesung Grundlagen der Informatik I Februar 2007

(Prüfungsnummern 11 248 143, 27 348 110)

Angaben zur Person:

Name, Vorname .....

Geburtsdatum .....

Matrikelnummer .....

Studiengang .....

Nicht von der Kandidatin/dem Kandidaten auszufüllen!

Bewertung:

Aufgabe	1	2	3	4	5	6	$\sum$	Note
Punktezahl								

Organisatorische Hinweise

Die folgenden Hinweise bitte aufmerksam lesen und die Kenntnisnahme durch Unterschrift bestätigen!

- Bitte legen Sie Ihren Studentenausweis und einen Lichtbildausweis zur Personenkontrolle bereit!
- Schreiben Sie deutlich und ausschließlich mit blauer oder schwarzer Tinte. Benutzen Sie keinen Bleistift. Unleserliche Antworten gehen nicht in die Bewertung ein.
- Hilfsmittel (außer Schreibmaterial) sind nicht zugelassen. Dies gilt auch für Taschenrechner, Mobiltelefone, elektronische Assistenten, etc.
- Fragen zu den Prüfungsaufgaben werden grundsätzlich nicht beantwortet!
- **Überprüfen Sie die Prüfungsaufgaben auf Vollständigkeit (16 Seiten inklusive Deckblatt) und einwandfreies Druckbild!** Vergessen Sie nicht, auf dem Deckblatt die Angaben zur Person einzutragen!
- Die Bearbeitungszeit beträgt 90 Minuten. Da Sie maximal 90 Punkte erreichen können, ist aus den möglichen Punkten pro Aufgabe leicht die Zeit zu berechnen, die Sie zum Lösen der jeweiligen Aufgabe investieren sollten. Die angegebene Punkteverteilung gilt unter Vorbehalt.
- Auf Ihrem Platz befindet sich 1 Blatt Schmierpapier. **Das Schmierpapier darf nicht mit abgegeben werden.** Die Lösung einer Aufgabe muss auf das Aufgabenblatt geschrieben werden und zwar in den freien Raum, der jeweils der Aufgabe folgt. Sollte der Platz nicht ausreichen, so müssen Sie bei der Aufsicht zusätzliche Formblätter anfordern und einheften lassen.
- Wenn Sie die Prüfung aus gesundheitlichen Gründen abbrechen müssen, so muss Ihre Prüfungsunfähigkeit durch eine Untersuchung in der Universitätsklinik nachgewiesen werden. Melden Sie sich in jedem Fall bei der Aufsicht und lassen Sie sich das entsprechende Formular aushändigen.

Erklärung

Durch meine Unterschrift bestätige ich den Empfang der vollständigen Klausurunterlagen und die Kenntnisnahme der obigen Informationen.

Erlangen, der 19. Februar 2007 .....
(Unterschrift)

Ich bin damit einverstanden, dass mein Prüfungsergebnis unter Angabe der Matrikelnummer anonymisiert veröffentlicht wird:

ja: ☐ nein: ☐

Erlangen, der 19. Februar 2007 .....
(Unterschrift)

Aufgabe 1 (10 Punkte)

Kreuzen Sie die richtige Antwort auf die jeweilige Frage an. Pro Frage ist immer nur eine Antwort richtig.

1. Welche Aussage trifft auf Unterprogramme nicht zu?
   (a) Mehrere Anweisungen werden zu einem Block zusammengefasst. ☐
   (b) Ein Unterprogramm wird mit `return` verlassen. ☐
   (c) Es muss mindestens ein Parameter übergeben werden. ☐

2. Welchen Rekursionstyp gibt es wirklich?
   (a) Geschwungene Rekursion ☐
   (b) Geschachtelte Rekursion ☐
   (c) Gestauchte Rekursion ☐

3. Welche Aussage trifft auf die Kodierung von Zeichen mit UTF-8 zu?
   (a) Anzahl der 1-Bits vor dem ersten 0-Bit im Start-Byte entspricht der Anzahl der Bytes des Mehrbyte-Zeichens. ☐
   (b) UTF-8 wird in Java für die Kodierung des Grunddatentyps `Character` verwendet. ☐
   (c) Selbe Länge (8-Bit) pro Zeichen macht die Kodierung einfach. ☐

4. Welche Aussagen gelten für die Darstellung von Gleitkommazahlen im IEEE-Format?
   (a) Ist das letzte Bit der Gleitkommazahl auf 0 gesetzt handelt, es sich um eine positive Zahl. ☐
   (b) Die Zahl der Bits des Exponenten ist im IEEE-Format genau spezifiziert. ☐
   (c) Im IEEE-Format können natürliche Zahlen exakt dargestellt werden. ☐

5. Welche Aussagen stimmen beim Übersetzen eines Programmes?
   (a) Die lexikalische Analyse zerlegt ein Quellprogramm in eine Folge terminaler Symbole. ☐
   (b) Bei der syntaktischen Analyse wird überprüft, ob das Programm terminiert. ☐
   (c) In der Optimierungsphase werden beispielsweise Kommentare entfernt, da sie die Programmausführung verlangsamen würden. ☐

6. Welche Einheit existiert im klassischen Universalrechner nicht?

   (a) Adresswerk ☐
   (b) Ausgabeeinheit ☐
   (c) Steuerwerk ☐

7. Was trifft bei der Vererbung nicht zu?

   (a) Statische Variablen werden vererbt. ☐
   (b) Überschriebene Methoden werden vererbt. ☐
   (c) Konstruktoren werden vererbt. ☐

8. Sortieren durch Mischen (Merge Sort) ist

   (a) kein Easy Split/Hard Join – Verfahren. ☐
   (b) kein Divide–And–Conquer – Verfahren. ☐
   (c) kein Greedy – Verfahren. ☐

9. Welche Aussage trifft auf Divide–And–Conquer – Verfahren zu?

   (a) Das Divide–And–Conquer – Verfahren besteht aus den Schritten Aufteilen des Problems und Lösen der Teilprobleme. ☐
   (b) Das Sortierverfahren *Merge Sort* hat die Komplexitätsklasse $O(n \log n)$ und sortiert die Daten nicht *in place*. ☐
   (c) Das beste Pivotelement für das Sortierverfahren *Quicksort* ist der Median der gegebenen Werte. ☐

10. Welche Aussage trifft auf kontextsensitive Grammatiken zu?

    (a) Kontextsensitive Grammatiken sind eine Teilmenge kontextfreier Grammatiken. ☐
    (b) Regeln einer kontextsensitiven Grammatik sind nichtkürzend. ☐
    (c) Kontextsensitive Grammatiken sind stets rechtslinear. ☐

Aufgabe 2 (14 Punkte)

1. Das folgende Programm soll dazu dienen, das Produkt der Zahlen im Intervall von [1;630] zu berechnen, die sowohl ein Vielfaches von 9 als auch von 7 sind. Das Ergebnis soll auf `stdout` ausgegeben werden. Korrigieren Sie die im Programm enthaltenen Fehler, so dass es sich um ein fehlerfreies Programm handelt und es die geforderte Funktionalität erfüllt. Schreiben Sie das Programm nicht neu.

```
public class  {

    public static berechneProdukt() {

        int produkt = 1;
        for (int i = 1; i <= 200, i++) {
            if (i % 9 = 0 || i % 7 = 0) {
                produkt = produkt * j;
            }
        }
        return ;
    }

    public static void main(String[] args){

        System.out.println( berechneProdukt() );
    }
}
```

2. Geben Sie an, was beim Ablauf des Programms der Reihe nach auf `stdout` ausgegeben wird.

```
public class Scope {

    private int[] feld = {9, 8, 7, 6, 5};

    public static void switchData(int[] feld, int value1, int value2) {
        int temp;
        temp = feld[0];
        feld[0] = feld[1];
        feld[1] = temp;

        temp = value1;
        value1 = value2;
        value2 = temp;
        System.out.println("feld " + feld[0] + " " + feld[1]);
        System.out.println("value " + value1 + " " + value2);
    }

    public static void main(String[] args){
        int[] feld = {1, 2, 3, 4};
        switchData(feld, feld[2], feld[3]);
        for (int i = 0; i < 4; i++) {
            System.out.println("feld " + i + " " + feld[i]);
        }

    }
}
```

Auf `stdout` wird ausgegeben:

Aufgabe 3 (10 Punkte)

Der Betrag der Differenz zweier Zahlen $m, n \in N_0$ kann wie folgt definiert werden:

$$\|m-n\| := \text{absdiff(m, n)} = \begin{cases} m & \text{wenn} \quad \text{n} = 0 \\ n & \text{wenn} \quad \text{m} = 0 \\ \text{absdiff(m} - 1, \text{n} - 1) & \text{sonst} \end{cases} \tag{1}$$

Schreiben Sie eine rekursive und eine iterative Methode um den Betrag der Differenz zweier natürlicher Zahlen mittels dem Verfahren (1) zu berechnen. Verwenden Sie dabei den Inkrement- und Dekrementoperator, aber nicht den Additions- bzw. Subtraktionsoperator von Java.

```
// Iterative Variante der Funktion

-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
```

```
// Rekursive Variante der Funktion

-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
```

Aufgabe 4 (16 Punkte)

Eine Warteschlange (beispielsweise an der Kasse eines Supermarktes) ist durch folgende Eigenschaften gekennzeichnet:

- Eine Person kann sich nur am Ende der Warteschlange anstellen.
- Wird jemand aufgerufen (bzw. ist mit dem Bezahlen an der Reihe), verlässt immer die Person am Anfang die Warteschlange.

Diese Warteschlange soll mittels einer verketteten Liste modelliert werden. Die Personen, die sich in der Warteschlange befinden, werden durch folgende Hilfsklasse innerhalb der Klasse `WarteSchlange` modelliert.

```
class Person {
    String name;           //Name der Person
    Person next;           //Nachfolger in der Warteschlange
    Person(String name) {  //Konstruktor
        this.name = name;
        this.next = null;
    }
}
```

Die Klasse `WarteSchlange` soll zwei Attribute `anfang` und `ende` besitzen. Diese sollen vom Typ `Person` sein und nur innerhalb der Klasse `WarteSchlange` sichtbar sein. Das Attribut `anfang` ist eine Referenz auf die erste Person in der Warteschlange (die also als nächstes an der Reihe ist). Das Attribut `ende` ist eine Referenz auf die letzte Person, die sich an der Warteschlange angestellt hat.
Schreiben Sie eine öffentlich verfügbare Methode `anstellen`, die als Parameter den Namen der Person (Typ `String`) übergeben bekommt und eine Person mit dem entsprechenden Namen, an das Ende der Warteschlange setzt. Die Methode gibt keinen Wert zurück.
Schreiben Sie eine öffentlich verfügbare Methode `naechstePersonAufrufen`, die keine Parameter übergeben bekommt und den Namen der Person, die an der Reihe ist, zurückliefert und aus der Warteschlange entfernt. Ist keine Person in der Warteschlange soll eine `WarteSchlangeLeerException` geworfen werden.

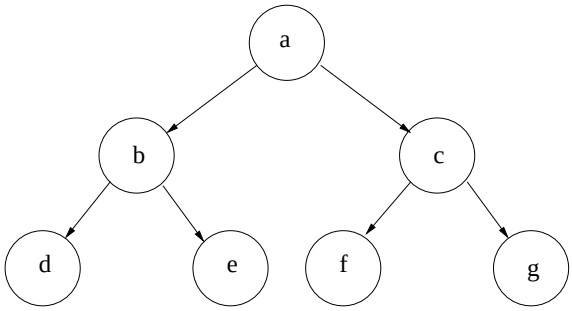
```
//Klasse WarteSchlangeLeerException

-----------------------------------------------------------------

    //parameterloser Konstruktor
    -----------------------------------------------------------
    -----------------------------------------------------------
    -----------------------------------------------------------
    -----------------------------------------------------------

    //Konstruktor mit einem String als Input-Parameter
    -----------------------------------------------------------
    -----------------------------------------------------------
    -----------------------------------------------------------
    -----------------------------------------------------------

}
```

```
public class WarteSchlange {

    // Attribute der Klasse WarteSchlange
    private Person anfang = null;
    private Person ende = null;

    // Methode anstellen
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------

    // Methode naechstePersonAufrufen
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
}
```

Aufgabe 5

(16 Punkte)

1. Welche der Beschriftungen des gegebenen binären Baumes entsprechen den folgenden Begriffen: Wurzel, Blätter und innere Knoten.

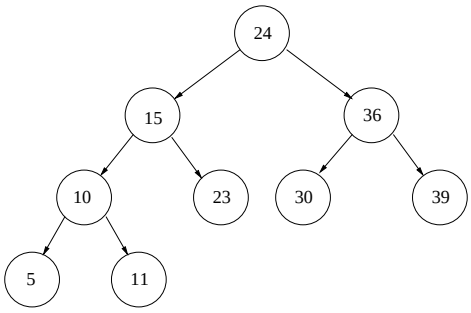


Wurzel: _____
Innere Knoten: _____
Blätter: _____

2. Definieren Sie, was man unter einem AVL-Baum versteht:

3. Die Komplexität des Einfügens neuer Elemente unterscheidet sich bei sortierten verketteten Listen und AVL-Bäumen. Illustrieren Sie den Unterschied an einem einfachen Beispiel mit sieben Elementen.

4 Fügen Sie in den folgenden AVL-Baum das Element 4 ein und führen Sie die notwendigen Rotationen aus. Kennzeichnen Sie dabei, an welchen Knoten Sie welche Rotation durchführen müssen und zeichnen Sie den Baum nach jeder Einzelrotation. Geben Sie bei allen Bäumen für jeden Knoten die Höhendifferenz zwischen dem rechten und dem linken Teilbaum an.



Aufgabe 6

(24 Punkte)

1. Modellieren Sie eine Klasse `Fahrzeug`, welche durch das Volumen des Benzintanks, dessen Füllstand und dem Verbrauch pro Hundert Kilometer definiert ist. Die Attribute sollen beim Erstellen eines Objekts gesetzt werden können. Der Zugriff auf die Attribute soll nur von abgeleiteten Klassen aus gestattet sein. Weiterhin soll die Klasse `Fahrzeug` eine Methoden `fahre` besitzen, welcher die gewünschte Anzahl zu fahrender Kilometer übergeben wird und zurückliefert, ob das Ziel erreicht wurde (ja, es war ausreichend Benzin vorhanden/ nein, die Strecke konnte mit dem Füllstand nicht gefahren werden). Dabei soll sich der Füllstand entsprechend der gefahrenen Kilometer senken, aber nicht unter 0 abfallen. Weiterhin benötigt ein Fahrzeug eine Methode `tanke`, welche die gewünschte Anzahl zu tankender Liter übergeben bekommt und die tatsächliche Anzahl getankter Liter zurückliefert (*Hinweis:* Das maximale Tankvolumen darf nicht überschritten werden).

```
// Klasse Fahrzeug

----------------------------------------------------------------

    // Attribute
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------

    // Konstruktor
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
    ------------------------------------------------------------
```

```
    // Methoden
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
}
```

2. Ein Auto ist eine spezielle Form eines Fahrzeuges. Implementieren Sie eine Klasse `Auto`, welche von `Fahrzeug` abgeleitet ist. Zusätzlich zu den Eigenschaften eines Fahrzeugs besitzt ein Auto eine Klimaanlage, welche entweder ein- oder ausgeschaltet sein kann. Der Standardverbrauch eines Autos ohne laufende Klimaanlage beträgt 8 Liter/100km. Ist die Klimaanlage eingeschaltet steigt der Verbrauch des Fahrzeugs um 10%. Weiterhin beträgt das Standardfassungsvermögen des Benzintanks 60 Liter. Implementieren Sie die Klasse `Auto`. Zur Steuerung der Klimaanlage soll eine Methode `klimaAnAus` bereitgestellt werden (*Hinweis:* Berücksichtigen Sie die Auswirkungen auf die Methode `fahre`). Auf alle Attribute soll von außen nur über Methoden zugegriffen werden können.

```
// Klasse Auto

----------------------------------------------------------------

    // Attribute
    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------

    // Konstruktoren
    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------


    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------
    ----------------------------------------------------------------
```

```
    // Methoden
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------
    --------------------------------------------------------------

}
```

3. Ein Motorrad ist ein weiteres spezialisiertes Fahrzeug. Im Gegensatz zu einem Auto kann ein Motorrad gedrosselt sein (ja/nein). Hierfür soll ein Motorrad zusätzlich die Methoden `getDrosselung` und `setDrosselung` besitzen. Der Methode `setDrosselung` wird ein Parameter übergeben, welcher angibt ob das Motorrad gedrosselt oder eine bestehende Drosselung aufgehoben werden soll. Auf alle Attribute soll von außen nur über die gegebenen Methoden zugegriffen werden können. Die Klasse `Motorrad` ist nicht zu implementieren. Zeichnen Sie für die Klassen `Fahrzeug`, `Auto` und `Motorrad` ein UML-Klassendiagramm, das alle Attribute und Methoden der einzelnen Klassen enthält und aus dem die Beziehungen der Klassen untereinander hervorgehen.