Fakultät Elektrotechnik/Informatik

Klausur Einführung in die Informatik II für Elektrotechniker 20. Februar 2003

Name: .....

Matr.-Nr. .....

Bearbeitungszeit: 120 Minuten

Bewertung
(bitte offenlassen :–)

Aufgabe	Punkte	Erreichte Punkte
1	6	
2	7	
3	6	
4	7	
5	7	
6	9	
Summe	42	

Spielregeln (**Jetzt lesen!**):

- Benutzen Sie für die Lösung der Aufgaben **nur** das mit diesem Deckblatt ausgeteilte Papier. Lösungen, die auf anderem Papier geschrieben werden, können **nicht** bewertet werden. Schreiben Sie ihre Lösung auch auf die Rückseiten der Blätter; benötigen Sie für eine Lösung mehr als ein Blatt, finden Sie am Ende der Klausur Leerblätter. Zusätzliches Papier können Sie von den Tutoren bekommen.
- Tragen Sie jetzt (vor Beginn der eigentlichen Bearbeitungszeit !!!) auf *allen* Blättern ihren Namen und ihre Matrikelnummer ein.
- Schreiben Sie deutlich! Unleserliche oder zweideutige Lösungen können nicht gewertet werden.
- Schreiben Sie **nicht** mit Bleistift.
- Bitte schreiben Sie nicht mit rotem oder grünem Stift (das sind die Farben für die Korrektur).
- Lesen Sie die Aufgaben jeweils bis zum Ende durch; oft gibt es hilfreiche Hinweise!
- Wir weisen noch einmal darauf hin, daß die Benutzung von Taschenrechnern und anderen elektronischen Hilfsmitteln nicht gestattet ist.

Viel Erfolg!

• AUFGABE 1 (6 Punkte) Theorie.

1. (2 Punkte) Ist *Selection Sort* besser als *Mergesort*? Welchen Vorteil bietet jedes der beiden Sortierverfahren? Erläutern Sie Ihre Antwort.

2. (2 Punkte) Erstellen Sie einen balancierten Binärbaum, bei dem durch Preorder-Traversierung die Zeichenkette $I\ N\ F\ E\ T - K\ L\ A\ U\ S\ U\ R$ (ohne Leerzeichen, mit Bindestrich) ausgegeben wird.

3. (2 Punkte) Was ist *Vererbung* und was hat es mit dynamischer Bindung zu tun?

• **AUFGABE 2 (7 Punkte) JAVA.**

1. (1 Punkt) Was sind *abstrakte Klassen* und was kann man mit ihnen machen?

2. (2 Punkte) Was versteht man unter *Referenzen*? Welche beiden Arten von „Gleichheit“ gibt es deshalb für Objekte?

3. (1 Punkt) Was muß man machen, um von einem Objekt einer anderen Klasse ein als *private* deklariertes Attribut lesen zu können?

4. (3 Punkte) Wo sind die Fehler im folgenden *JAVA*-Code? Beschreiben Sie die Fehler und geben Sie die Zeilenzahlen des Auftretens an. (Folgefehler werden wie üblich ignoriert.)

```
 1 public class Main {
 2     private int value = 4711;
 3
 4     public static void main(String[] args) {
 5         Terminal.println("Hallo");
 6         run();
 7     }
 8
 9     void run() {
10         Bar b = new Bar();
11         b.work();
12         Terminal.println(b.sum);
13     }
14 }
15
16 interface Foo {
17     public void work();
18
19     public int triple(int a) { return 3 * a; }
20 }
21
22 class Bar extends Main implements Foo {
23     public int sum = 0;
24
25     public void work() {
26         for (int i = 1; i <= value; i++) {
27             sum = sum + i;
28         }
29     }
30 }
```

• AUFGABE 3 (6 Punkte) Numerik.

Schreiben Sie eine *JAVA*-Methode

```
double wurzel(int n, double a, double eps)
```

welche $\sqrt[n]{a}$, d.h. die `n`-te Wurzel aus `a`, mit der Genauigkeit `eps` berechnet.

Verwenden Sie dabei nicht die Funktion `Math.pow` aus der *Java*-Klasse `Math`, sondern berechnen Sie die Wurzel durch Approximation mittels Newton Verfahren:

$$\begin{aligned} x_0 &= a \\ x_{k+1} &= x_k - \frac{x_k^n - a}{n \cdot x_k^{n-1}} \end{aligned}$$

Beenden Sie jeweils den Approximationsprozeß, wenn eine Genauigkeit von `eps` erreicht ist. Formulieren Sie diese Bedingung mit Hilfe einer zu erstellenden Hilfsfunktion `boolean nahe( double x, double y, double eps )`.

Falls `a` negativ ist oder `n` bzw. `eps` keine positiven Zahlen sind, ist die Wurzel nicht definiert. Lösen Sie in diesem Fall eine `IllegalArgumentException` aus.

• AUFGABE 4 (7 Punkte) Vererbung.

Winkel können sowohl in Grad ($0° - 360°$) als auch im Bogenmaß ($0 - 2\pi$) angegeben werden.

1. (1 Punkt) Schreiben Sie eine abstrakte Klasse `Winkel`, welche einen Winkel in beliebiger Repräsentation darstellt.

   Sie soll die abstrakten Methoden

   - `Winkel scale(double factor)` zur Skalierung des Winkels um einen bestimmten Faktor sowie
   - `Winkel add(Winkel other)` zur Addition des eigenen zu einem anderen Winkel

   bereitstellen.

2. (2 Punkte) Leiten Sie von der Klasse `Winkel` eine Klasse `Grad` ab, welche einen Winkel in Grad repräsentiert.

   Die Klasse soll einen geeigneten Konstruktor sowie die Methoden aus der Klasse `Winkel` implementieren.

3. (2 Punkte) Leiten Sie von `Winkel` ebenfalls eine Klasse `Bogenmaß` ab, welche einen Winkel im Bogenmaß darstellt.

   Die Klasse soll ebenfalls einen geeigneten Konstruktor sowie die Methoden aus der Klasse `Winkel` implementieren.

4. (2 Punkte) Erweitern Sie nun die Klasse `Winkel` um eine Methode `Winkel average(Winkel[] w)`, welche den Durchschnitt aller Winkel im nichtleeren Feld `w` berechnet.

• AUFGABE 5 (7 Punkte) Pizza-Queue.

Ein Pizzadienst will seine Bestellungen mit dem Computer verwalten. Die aufgenommenen Bestellungen werden dabei in der Reihenfolge ihres Einganges bearbeitet, d.h. diejenigen Kunden, die am längsten warten, werden als erste bedient.

Es sei die Klasse `Pizza` geeignet definiert, dann läßt sich eine `Bestellung` wie folgt modellieren:

```
class Bestellung {
    String  kunde;
    Pizza[] wunsch;
}
```

Ihre Aufgabe ist es nun, die Warteschlange der Bestellungen zu implementieren.

1. (1 Punkt) Schreiben Sie eine Klasse `QueueElem`, welche ein Element der Warteschlange verwaltet. Dazu sind Verweise auf den eventuell gespeicherten Dateninhalt sowie das nächste Element erforderlich.

   Ihre Klasse soll einen geeigneten Konstruktor zum Setzen der Attribute bereitstellen.

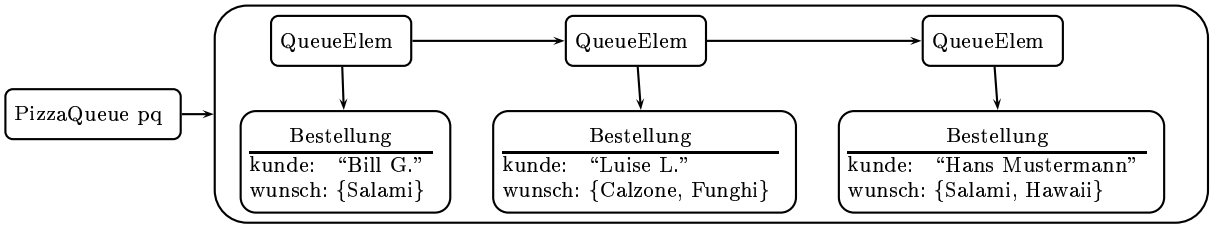
2. (2 Punkte) Schreiben Sie ferner eine Klasse `PizzaQueue` zur Verwaltung der Warteschlange. Die Klasse soll als abstrakter Datentyp realisiert werden, d.h. die Attribute sollen von außen nicht direkt zugreifbar sein, sondern nur über die unten angegeben Methoden.

   Die Klasse `PizzaQueue` soll die folgenden Methoden bereitstellen:

   - Der Konstruktor soll eine leere Warteschlange erzeugen.
   - `Bestellung loeschen()`, welche das erste Element aus der Warteschlange entfernt und die entsprechende Bestellung zurückliefert. Wenn die Warteschlange leer ist, soll `null` zurückgegeben werden.
   - `void einfuegen(Bestellung b)`, welche eine neue Bestellung in der Warteschlange einfügt.

     Dabei werden neue Bestellungen derart behandelt, daß sie als letzte, d.h. nach allen bereits in der Warteschlange vorhandenen Bestellungen, geliefert werden.

Beispiel:



- `pq.einfuegen(("Gerhard S.", {Margarita}))` ⟶ hinter *Hans Mustermann* einreihen
- `pq.loeschen() = ("Bill G.", {Salami})`

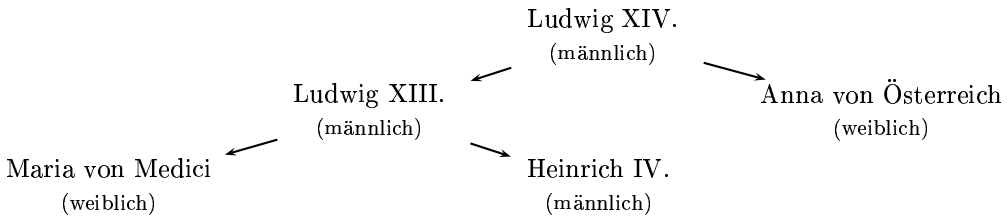
• AUFGABE 6 (9 Punkte) Stammbaum-Verwaltung.

Wenn wir den Stammbaum einer Person, d.h. seine (biologischen) Vorfahren darstellen wollen, können wir dafür einen Binärbaum verwenden, in dem jeweils Vater und Mutter als Nachfolger eingetragen sind.

Hinweis: Bei der Bearbeitung dieser Aufgabe können Sie voraussetzen, daß Ihnen die Klasse `BinTree` aus dem Skript zur Verfügung steht. Sie stellt Ihnen die folgenden Methoden bereit:

- Die Konstruktoren `BinTree()`, `BinTree(Object o)` und `BinTree(Object o, BinTree l, BinTree r)`
- Die Zugriffsmethoden `Object value()`, `BinTree left()` und `BinTree right()`
- Die Hilfsfunktionen `boolean isEmpty()`, `boolean isLeaf()` und `boolean isNode()`

Die im Stammbaum dargestellten Personen haben einen Namen sowie ein Flag, ob sie männlich sind oder nicht.



1. (2 Punkte) Erstellen Sie eine Klasse `Person`, welche den Namen einer Person (als `String`) sowie dessen Geschlecht verwaltet. Die Attribute sollen von außen nicht zugreifbar sein und im Konstruktor gesetzt werden.

   Die Klasse `Person` soll Methoden `getName` und `isMann` zum Lesen von Namen und Geschlecht sowie eine geeignete Methode `public String toString()` zur Ausgabe bereitstellen.

   Ferner soll eine Methode `boolean equals(Person p)` in der Klasse `Person` vorhanden sein, welche `true` liefert, wenn Name und Geschlecht mit der anderen Person übereinstimmen.

   Hinweis: Zum Vergleichen von Strings kann die Methode `boolean String.equals(String otherString)` verwendet werden.

2. (3 Punkte) Erweitern Sie nun die Klasse `BinTree` um eine Methode

   ```
   boolean testeBaum()
   ```

   Sie soll feststellen, ob der Stammbaum (biologisch betrachtet) korrekt ist, d.h. jede Person an einem inneren Knoten im Stammbaum einen männlichen (Vater) und einen weiblichen Vorfahren (Mutter) besitzt.

3. (3 Punkte) Erweitern Sie die Klasse `BinTree` ferner um eine Methode

   ```
   boolean istVorfahreVon(Person p)
   ```

   welche herausfindet, ob die Person `p` ein Vorfahre der Person an der Wurzel ist.

4. (1 Punkt) Welche Traversierungsstrategie haben Sie in der obigen Methode `istVorfahreVon` implementiert und warum haben Sie diese gewählt?