Fachbereich Informatik

Klausur Einführung in die Informatik II für Elektrotechniker 9. Oktober 2001

Name: .....

Matr.-Nr. .....

Bearbeitungszeit: 120 Minuten

Bewertung
(bitte offenlassen :–)

Aufgabe	Punkte	Erreichte Punkte
1	4	
2	8	
3	4	
4	6	
5	10	
6	12	
Summe	44	

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 nicht gestattet ist.

Viel Erfolg!

- **AUFGABE 1 (4 Punkte) Theorie.**

1. (2 Punkte) Erläutern Sie die Vor– und Nachteile der Speicherung von sortierten Daten in Arrays sowie Binärbäumen?

2. (1 Punkt) Was verstehen Sie unter *Exceptions*?

3. (1 Punkt) Was verstehen Sie unter *Vererbung*?

- **AUFGABE 2 (8 Punkte) JAVA.**

1. (2 Punkte) Wie kann man die softwaretechnisch wichtige Trennung von extern sichtbarer Schnittstelle sowie interner Realisierung in *Java* ermöglichen? Nennen Sie zwei Möglichkeiten und erläutern Sie diese kurz.

2. (1 Punkt) Wozu dienen *Packages* in Java?

3. (2 Punkte) Was wird in folgendem Java-Programm ausgegeben?

```
 1 class Main {
 2     public static void main(String[] args) {
 3         Number six = new Number(6);
 4         Number two = new Number(2);
 5         six.add(two);
 6     }
 7 }
 8
 9 class Number {
10     static int n = 0;
11
12     Number(int num) {
13         n = num;
14     }
15
16     void add(Number other) {
17         for (int i = 1; i <= other.n; i++) {
18             Terminal.println(n);
19             n++;
20         }
21     }
22 }
```

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

```
 1 class Foo {
 2     int value;
 3     private int counter;
 4
 5     public int getValue() {
 6         Run.counter++;
 7         return this.value;
 8     }
 9 }
10
11 class Bar extends Foo {
12     Bar(int n) {
13         value = n;
14     }
15
16     void addValue(Bar other) {
17         this.value = value + other.value;
18         this.counter++;
19     }
20 }
21
22 class Run {
23     static int counter = 0;
24
25     void run() {
26         Foo f = new Foo(10);
27         Bar b = new Bar(20);
28         b.addValue(f);
29         Terminal.println(b.getValue());
30     }
31 }
```

• AUFGABE 3 (4 Punkte) Numerik.

Schreiben Sie eine *JAVA*-Methode `double pi(int stellen)`, welche die Kreiszahl π mit einer Genauigkeit von mindestens `stellen` nach dem Komma berechnet.

Dabei sollen Sie die Berechnungsformel des *Wallischen Produktes* verwenden:

$$\frac{\pi}{2} = \prod_{n=1}^{\infty} \frac{4n^2}{4n^2 - 1}$$

Beenden Sie den Approximationsprozeß, wenn die vorgegebene Genauigkeit erreicht ist. Formulieren Sie diese Bedingung mit Hilfe einer zu erstellenden Hilfsfunktion `boolean close( double x, double y, double eps )`.

• AUFGABE 4 (6 Punkte) Binärbäume.

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()`

Erweitern Sie die Klasse `BinTree` um die folgenden Methoden:

1. (3 Punkte) Fügen Sie (zur Klasse `BinTree`) eine Methode `int depth()` hinzu, welche die Tiefe des Baumes, d.h. die maximale Entfernung der Blätter von der Wurzel, berechnet.
2. (3 Punkte) Schreiben Sie ferner eine Methode `boolean sameStructure(BinTree other)`, welche `true` liefert, wenn zwei Bäume die gleiche Struktur haben. Der Dateninhalt spielt dabei keine Rolle.

• AUFGABE 5 (10 Punkte) Vererbung.

In dieser Aufgabe sollen Sie ein einfaches Grafiksystem unter Verwendung von Vererbung implementieren.

Wir gehen zur Vereinfachung davon aus, daß wir nur rechteckige Elemente darstellen. Die Position auf dem Bildschirm sowie die Größe können dann durch die linke untere sowie die rechte obere Ecke des Rechtecks repräsentiert werden.

Neben den direkt darstellbaren Elementen gibt es noch Containerobjekte (wie beispielsweise Fenster), die selbst auch eine rechteckige Teilfläche des Bildschirms überdecken und mehrere Grafikobjekte als Unterkomponenten enthalten können.

1. (1 Punkt) Schreiben Sie ein Interface `Moveable`, welche eine Methode `void move(Point p)` zum Verschieben eines Grafikobjektes um den angegebenen Vektor `p` vom Typ `Point` (siehe unten) deklariert.

2. (2 Punkte) Schreiben Sie eine Klasse `Point`, welche die `x`– und `y`–Koordinate eines Punktes enthält.

   Sie soll das Interface `Moveable` realisieren, d.h. eine Methode `void move(Point p)` zum Verschieben bereitstellen.

3. (3 Punkte) Schreiben Sie ferner eine Klasse `Rectangle`, welche ein rechteckiges Grafikelement repräsentiert. Es soll den überdeckten Bildschirmbereich mittels linker unterer sowie rechter oberer Ecke speichern, einen Bezeichner vom Typ `String` sowie einen geeigneten Konstruktor besitzen.

   Die Klasse `Rectangle` soll ebenfalls das Interface `Moveable` realisieren, d.h. eine Methode `void move(Point p)` anbieten, welche das Rechteck entsprechend verschiebt.

   Darüber hinaus soll eine Methode `String activeRectangle(Point MousePosition)` angeboten werden, welche die Bezeichnung des Rechtecks zurückliefert, sofern die Maus darauf zeigt, anderenfalls eine leere Zeichenkette.

4. (4 Punkte) Leiten Sie von `Rectangle` eine Klasse `Container` ab, die neben den Bildschirmkoordinaten des Bildschirmausschnitts und dem Bezeichner darüber hinaus ein Feld von Rechtecken als abhängige Unterkomponenten enthält.

   Hinweise:

   - Bedenken Sie, daß Sie bei der Vererbung den Konstruktor anpassen sollten.
   - Beim Verschieben (Methode `move`) müssen auch die abhängigen Unterkomponenten mitverschoben werden.
   - Die Methode `activeRectangle` soll die Bezeichnung desjenigen Rechtecks zurücklieferen, auf das die Maus gerade zeigt. Falls es ein solches Grafikelement nicht gibt, soll eine leere Zeichenkette zurückgegeben werden, bei mehreren (sich überlappenden) Rechtecken an `MousePosition` die Bezeichnung eines beliebigen dieser Rechtecke. Dabei darf nicht der Bezeichner eines `Container`–Objektes, sondern nur eines echten `Rectangle`–Objectes zurückgeliefert werden.

• AUFGABE 6 (12 Punkte) Multimengen.

Eine *Multimenge* ist eine Datenstruktur, bei der die Reihenfolge der Elemente keine Rolle spielt. Im Gegensatz zu normalen Mengen können Elemente jedoch mehrfach enthalten sein. So sind beispielsweise die Multimengen $\{1, 2, 2\}$ und $\{2, 1, 2\}$ identisch, da sie beide je einmal die 1 und zweimal die 2 enthalten, jedoch verschieden von $\{1, 2\}$, da hier nur einmal die 2 enthalten ist.

In dieser Aufgabe sollen sie eine einfache *Lagerverwaltung* mittels Multimengen implementieren. Dabei sollen als Datenobjekte die jeweiligen Artikel gespeichert werden.

Eine mögliche Implementierung verwendet eine listenartige Struktur, bei der in den Zellen jeweils die Datenobjekte gespeichert werden zusammen mit einem Zähler, der ihre Anzahl angibt.

1. (2 Punkte) Erstellen Sie eine Klasse `Article`, welche die Daten eines Artikels enthalten. Dies sind die Artikelgruppe sowie die Artikelbezeichnung (jeweils als String). Die Attribute sollen von außen nicht zugreifbar sein und im Konstruktor gesetzt werden.

   Die Klasse *Article* soll ferner eine Methode `boolean equals(Article other)` bereitstellen, welche zwei Artikel miteinander vergleicht, sowie eine geeignete Methode `String toString()` zur Ausgabe.

   Hinweis: Sie können zwei Strings mit der Methode `int String.compareTo(String other)` vergleichen. Das Ergebnis ist negativ/Null/positiv, falls der entsprechende String kleiner/gleich/größer als der andere ist.

2. (1 Punkt) Schreiben Sie ferner eine Klasse `BagCell`, die ein einzelnes Element einer einfach verketteten Liste repräsentiert, welches neben den eigentlichen Datenobjekten einen Zähler für deren Anzahl enthält.

3. (5 Punkte) Verwenden Sie obige Klassen, um eine Klasse `ArticleBag` zu implementieren. Diese soll folgende Operationen bereit stellen:

   - eine Methode `void insert(Article art)`, welche einen Artikel einmal der Multimenge hinzufügt.
   - eine Methode `void remove(Article art, int count)`, welche einen Artikel `count`-mal aus der Multimenge entfernt.

     Falls der Artikel nicht oder nicht in genügender Anzahl vorhanden ist, soll eine `Exception(String message)` ausgelöst werden und die Multimenge unverändert bleiben.

4. (4 Punkte) Erweitern Sie die Klasse `ArticleBag` um eine Methode `void showSortedBag()`, welche die im Lager vorhanden Artikel zusammen mit deren Anzahl auf dem Terminal ausgibt.

   Die Artikel sollen dabei nach deren Anzahl geordnet ausgegeben werden. Bei gleicher Anzahl ist die Reihenfolge der Artikel nicht relevant.

   Beispielausgabe:

   ```
   20 x Tisch, Modell Equisit
   10 x Stuhl, Campingstuhl
   10 x Schrank, Buecherschrank Bork
   10 x Sofa, Ledersofa
   4 x Stuhl, Kuechenstuhl Dreamland
   1 x Schrank, Kleiderschrank
   1 x Liege, Gartenliege Sonnenland
   ```

   Hinweis: Da die Reihenfolge der Elemente in der Multimenge (außer für diese Methode) keine Rolle spielt, können Sie diese intern beliebig wählen und gegebenenfalls ändern. Sie können also die Liste intern sortieren, falls gewünscht. Es sind jedoch auch andere Implementierungen möglich.