

Name:  
Hauptfach:  
Tutor:

Matrikelnummer:

---

Sommersemester 2002  
Dr. Frank Hoffmann

12. Juli 2002

**Klausur zur Vorlesung  
Informatik B**

1.	2.	3.	4.	5.	6.	$\Sigma$
/6	/5	/4	/5	/4	/4	/28

Beginn: 8<sup>20</sup>, Ende: 10<sup>00</sup> (100 min.)

**Außer Schreibutensilien sind keine Hilfsmittel erlaubt!  
Nicht mit Bleistift schreiben!**

Auf diesen Klausurblättern ist genügend Platz, um die Lösungen der Aufgaben aufzuschreiben. Falls **zusätzliche Blätter** benötigt werden, so werden diese ausgeteilt und sie müssen mit der Matrikelnummer und der Aufgabennummer versehen werden. *Alle Klausurblätter sowie dieses Deckblatt sind aber auf jeden Fall abzugeben, auch falls einzelne Aufgaben nicht bearbeitet wurden!*

**Nicht vergessen, auf allen Blättern die Matrikelnummer einzutragen, auf diesem Deckblatt auch den Namen sowie Ihren Studiengang und Tutor!**

**Viel Erfolg!**

1. Objektorientierung in Java (6 Punkte).

/6

- (a) (2 Punkte) Was versteht man in Java unter einem interface und wozu benutzt man es? (Höchstens 3 Sätze!)
- (b) (4 Punkte) Betrachten Sie die folgenden Java-Klassen. Was ist der Output eines Aufrufes der main-Methode in der Klasse Master, begründen Sie jeweils mit einem kurzen Argument Ihre Antwort!

```
public class Person extends Object {
    Person() { /* Nullkonstruktor */ }
    public void printMe() { System.out.println("Ich bin eine Person."); }
}

public class Student extends Person {
    Student() { /* Nullkonstruktor */ }
    public void printMe() { System.out.println("Ich bin ein Student."); }
}

public class Bachelor extends Student {
    Bachelor() { /* Nullkonstruktor */ }
    public void printMe() { System.out.println("Ich bin
                                                ein Bachelorstudent."); }
}

public class Master extends Student {
    Master() { /* Nullkonstruktor */ }
    public void printMe() { System.out.println("Ich bin
                                                ein Masterstudent."); }
}

public static void main (String[] args) {
    Person maria = new Bachelor();
    Student paul = new Master();
    Object obj = new Student();
    maria.printMe();
    paul.printMe();
    ((Person) obj).printMe();
    obj = paul;
    ((Student) obj).printMe();
}
}
```

Matrikelnummer:

---

2. Baumtraversierung (5 Punkte).

/5

- (a) (2 Punkte) Zeichnen Sie einen binären gewurzelten Baum, dessen Knoten mit einzelnen Buchstaben markiert sind, so dass eine Preorder-Traversierung das Wort EXAMFUN und eine Inorder-Traversierung MAFXUEN ergibt.
- (b) (1 Punkt) Was ergibt die Postorder-Traversierung?
- (c) (2 Punkte) In einem gewurzelten Binärbaum  $T$  mit  $n$  Knoten seien seine Blätter bezüglich Inorder-Traversierung nummeriert mit 1 bis  $b$ . Beschreiben Sie (verbal) einen Algorithmus, der für alle inneren Knoten bestimmt, welche Blätter unter ihnen liegen. Welche Laufzeit hat Ihr Verfahren? (Hinweis: Achten Sie auf eine kompakte Ausgabe.)

Matrikelnummer:

---

3. Asymptotisches Wachstum (4 Punkte).

/4
----

Seien  $f, g$  Funktionen die natürliche Zahlen in positive reelle Zahlen abbilden.

- (a) (2 Punkte) Definieren Sie, was es heißt, dass  $f(n) = O(g(n))$  gilt.
- (b) (2 Punkte) Benutzen Sie diese (!) Definition, um für  $f(n) = 1000n + \log_2 n + 42$  zu zeigen

$$f(n) = O(n \log_2 n)$$

Matrikelnummer:

---

4. **Algorithmik + Zusatzaufgabe** (4 Punkte).

/4

In einem Array der Größe  $n$  werden ganze Zahlen aus dem Bereich von 0 bis  $n$  abgespeichert. Diese Einträge seien paarweise verschieden.

- (a) (4 Punkte) Nehmen Sie an, die Einträge sind aufsteigend sortiert. Geben Sie in Pseudocode einen  $O(\log_2 n)$  Algorithmus an, der bestimmt, welche ganze Zahl zwischen 0 und  $n$  nicht im Array vertreten ist. Begründen Sie die Laufzeit.
- (b) (4 mögliche Zusatzpunkte) Geben Sie einen asymptotisch optimalen Algorithmus an für den Fall, dass die Folge nicht notwendigerweise sortiert gegeben ist. Warum ist er optimal?

Matrikelnummer:

---

5. Datenstrukturen I (5 Punkte).

/5
----

- (a) (1 Punkte) Wann ist ein binärer Suchbaum ein AVL-Baum?
- (b) (2 Punkte) Zeichnen Sie einen AVL-Baum  $T$  mit insgesamt 6 inneren Knoten, die alle gerade ganze Zahlen als Schlüssel haben. Dabei soll wenigstens einer der Knoten  $v$  genau ein Nichtblatt-Kind  $w$  haben.
- (c) (2 Punkte) Fügen Sie eine ungerade Zahl als Schlüssel in den Baum ein, so dass dieser zunächst Kind von  $w$  ist. Falls notwendig, stellen Sie die AVL-Eigenschaft wieder her.

Matrikelnummer:

---

6. Datenstrukturen II (4 Punkte).

/4
----

- (a) (1 Punkt) Welche Methoden hat der ADT Stack?
- (b) (3 Punkte) Invertieren Sie einen Stack  $S$ . Sie können weitere Stacks als Hilfsdatenstruktur benutzen, die Ausgabe ist aber wiederum  $S$ . Was ist die Laufzeit Ihres Verfahrens und warum?