UNIVERSITÄT DES SAARLANDES

FR 6.2 – Informatik
Prof. Dr. Jörg H. Siekmann
Dr. Claus-Peter Wirth

# Klausur zu
# „Einführung in die Informatik für Hörer aller Fakultäten I"
# (Sommersemester 2002)

Name, Vorname: ..............................................................

Matr.Nr.: ..............................................................

Vor der Bearbeitung:

- Legen Sie Personal– und Studentenausweis offen vor sich hin.
- Schreiben Sie auf *jedes* Blatt Ihren Namen und Ihre Matrikelnummer.
- Benutzen Sie nur die vorbereiteten Blätter – auch als Schmierblätter.
- Bei vorübergehendem Verlassen des Raumes sind sämtliche Unterlagen bei der Aufsicht abzugeben.
- Auch aus Versehen mitgenommene Lösungen können nicht mehr gewertet werden.
- Benutzen Sie keinen roten Stift.

Bei der Abgabe:

- Kennzeichnen Sie Schmierblätter gesondert, und geben Sie diese zusammen mit allen Aufgabenblättern ab.
- Kreuzen Sie jede bearbeitete Aufgabe in untenstehender Tabelle an.

Unterschreiben Sie hier: **Viel Erfolg!**

..............................................................

| Aufgabe | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | $\Sigma$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| bearbeitet | | | | | | | | | | | |
| max. Punktzahl | 35 | 35 | 45 | 35 | 45 | 105 | 60 | 170 | 40 | 40 | **610** |
| **Punkte** | | | | | | | | | | | |

Name: Matrikelnr.:

# (35) 1 Zahlendarstellungen

(10) 1. Geben Sie für die Zahlen 22 und 69 (beide in Dezimaldarstellung, d. h. zur Basis 10) die Binärdarstellung (d. h. zur Basis 2) an!

10110 und 1000101

(15) 2. Gegeben sind die Zahlen 10110 und 100111 in Binärdarstellung (d. h. zur Basis 2). Addiere beide Zahlen. Geben Sie die Berechnung sowohl in Binär- als auch in Dezimaldarstellung an!

| | | |
|---|---|---|
| 10110 | und | 22 |
| 100111 | | 39 |
| 1100 | | 10 |
| 111101 | | 61 |

(10) 3. Geben Sie für die Zahlen 1101 und 101101101101 (beide in Binärdarstellung, d. h. zur Basis 2) die Oktaldarstellung (d. h. zur Basis 8) an!

15 und 5555.

Name: Matrikelnr.:

## (35) 2 Logische Gatter

Gegeben seien die Boole'schen Werte für folgende Gatter:

| A | B | AND(A,B) | NAND(A,B) | OR(A,B) | NOR(A,B) | EQ(A,B) |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |

(15) 1. Berechnen Sie die Boole'schen Werte für folgendes Gatter: NAND(A,OR(A,NOR(B,C)))

| A | B | C | NOR(B,C) | OR(A,NOR(B,C)) | NAND(A,OR(A,NOR(B,C))) |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 |

2. Mit Hilfe der obigen Tabelle kann man leicht nachvollziehen, dass die Boole'schen Werte der Gatter AND(A,B) und NOT(NAND(A,B)) bzw. OR(A,B) und NOT(NOR(A,B)) identisch sind. Mit Hilfe von NAND und NOT kann man folglich das AND und mit NOT und NOR das OR bilden.

(10) (a) Geben Sie unter Verwendung der in der Tabelle aufgeführten Gatter eine weitere Möglichkeit zur Bildung von EQ an!

OR(AND(A,B),NOR(A,B))

(10) (b) Geben Sie unter Verwendung der in der Tabelle aufgeführten Gatter eine weitere Möglichkeit zur Bildung von AND an!

NOR(NAND(A,B),NOR(A,B)) oder
NAND(NAND(A,B),NAND(A,B))

Name: Matrikelnr.:

## (45) 3 Allgemeine Wissensfragen

(5) 1. Was ist ein Bus und in welchem Gebiet der Informatik spielt er eine Rolle?

Datenleitung / Hardware (Technische Informatik)

(10) 2. Was ist ein Betriebsystem?

Software zur Koordination von Programmen, Multi-Tasking, Prozeßsteuerung, Hardwarezugriffskoordination, Fehlerbehandlung, Ausfallsicherheit. Z. B. Windows und Linux.

(5) 3. Was ist ein Emulator und in welchem Gebiet der Informatik spielt er eine Rolle?

Programm, das vorgaukelt, eine bestimmte Maschine zu sein. / Software (Praktische Informatik)

(10) 4. Was ist ein Interpreter?

Ein spezielles Emulator-Programm, das ein Programm einer Quellsprache als Folge von Befehlen einer Zielsprache interpretiert.

(15) 5. Worin unterscheidet sich ein Compiler von einem (eigentlichen) Interpreter?

Ein Compiler übersetzt das ganze Programm nur einmal (aber vollständig) in die Zielsprache. Ein eigentlicher Interpreter übersetzt nur den Teil, der gerade ausgeführt wird, aber dafür auch sooft dieser ausgeführt wird.

## (35) 4 Datentypen in JAVA

(8) 1. Welcher Unterschied besteht zwischen Ganz– und Gleitkommazahlen?

Ganzzahlen: Genaue Darſtellung von Intervall der ganzen Zahlen.
Gleitkommazahlen: Ungefähre Darſtellung eines großen Bereichs von Bruchzahlen oder ſehr großen Zahlen (mit Rundungsfehlern).

(5) 2. Ist der CHAR–Datentyp besser als der STRING–Datentyp zur Handhabung natürlicher Sätze geeignet, oder verhält es sich umgekehrt?
Antwort bitte erläutern!

STRING iſt beſſer.
Weil CHAR nur für einzelne Zeichen definiert iſt – im Gegenſatz zu STRING.

(2) 3. Welchen Datentyp verlangt der Bedingungsteil einer `if`–Anweisung in Java?

Den Boole'ſchen Datentyp BOOLEAN.

(8) 4. Ordne mittels Pfeilen den nachfolgenden Ganzzahltypen die passende Bitlänge zu — bitte beachten, daß $n$ Bits einem Intervall $(-2^{n-1})\dots(2^{n-1}-1)$ entspricht:

| | |
|---|---|
| byte | 64 |
| long | 8 |
| short | 16 |
| int | 32 |

(10) 5. Ein Mensch besteht aus Abermilliarden (über 100 Milliarden) Zellen.
Welche Datentypen sind zur Darstellung dieser Zahl überhaupt geeignet? Gibt es auch Datentypen zur genauen Darstellung? Was ist die Voraussetzung einer genauen Darstellung der Anzahl?

FLOAT, DOUBLE; LONG.
LONG für genaue Darſtellung nötig.
Zur genauen Darſtellung muß die Zahl der verfügbaren Bits ausreichen, damit die Zahl voll darſtellbar iſt.

(2) 6. Welche Werte gibt es beim Datentyp `boolean`?

true, falſe.

Name: Matrikelnr.:

# (45) 5 JAVA Wissenfragen

(10) 1. Was bedeutet das Schlüsselwort `static` vor dem Namen einer Methode in JAVA?

Eine statische Methode bezieht sich nicht auf ein Objekt der Klasse als implizites Argument, sondern bekommt nur die in Klammern angegebenen Argumente.

(10) 2. Warum gibt es in objektorientierten Sprachen Klassenhierarchien? (Worin besteht der grosse praktische Vorteil beim Programmieren einer Unterklasse?)

In der Vererbung. Vererbt werden: Abstrakte Methoden (Signatur), Konkrete Methoden (Programm), wenn nicht überschrieben. Slots.

(15) 3. Was kann eine Unterklasse ihren Oberklassen hinzufügen?

Slots. Konstruktoren. Konkrete Methoden für abstrakte. Ueberschreibung (overriding) von Methoden.

(4) 4. Was kann eine abstrakte Klasse in JAVA nicht haben?

Konstruktoren

(6) 5. Wieviele direkte Oberklassen darf eine Klasse in JAVA haben und wie heißt das Schlagwort dazu?

Nur eine! Keine multiple Vererbung!

# (105) 6 JAVA Programmieren: Stapel

Der Datentyp "Keller" oder "Stapel" (engl. "Stack") realisiert ein LIFO (last in, first out) Verhalten. Der Name ergibt sich aus der Ähnlichkeit mit einer Stapelverwaltung in der Lagerhaltung (z. B. bei Getränkekisten). Es gibt die folgenden Funktionen :

| | | | | |
|---|---|---|---|---|
| empty | : | | $\rightarrow$ | Stack |
| push | : | Stack, Object | $\rightarrow$ | void |
| pop | : | Stack | $\rightarrow$ | void |
| top | : | Stack | $\rightarrow$ | Object |
| Null | : | Stack | $\rightarrow$ | boolean |

Die Funktion empty stellt einen leeren Stapel bereit. Die Prozedur push verändert den bestehenden Stapel destruktiv, indem sie ein weiteres Objekt oben auf den Stapel stellt. Die Prozedur pop verändert den bestehenden Stapel destruktiv, indem sie das oberste Objekt vom Stapel entfernt; bei leerem Stapel als Eingabe wird der Stapel nicht verändert. Die Funktion top gibt das oberste Objekt des Stapels als Wert zurück. Die Funktion Null überprüft, ob der Stapel leer ist.

Zur Implementierung dürfen Sie die aus den Übungen bekannte Klasse Cons verwenden, die verkettete Listen mit Cons-Zellen implementiert:

```
public class Cons {
  /* Cons-cells for lists */
  private Object car; /* entry */
  private Cons   cdr; /* rest of list */
  public static Cons nil = new Cons();
    /* empty list */
  private Cons() {
    /* only for nil */
    car = null;
    cdr = null;}
  public Cons(Object head, Cons tail) {
    car = head;
    cdr = tail;}
  public Object car() {
    return this.car;}
  public Cons cdr() {
    return this.cdr;}
  public String toString() {
    return (this == nil) ? "[]" : ("[" + car + cdr.nonemptytoString() + "]");}
  public String nonemptytoString() {
    return (this == nil) ? "" : (", " + car +cdr.nonemptytoString());}
}
```

Die abstrakte Klasse Stack hat die Form:

```
public abstract class Stack{
  abstract public void    push (Object o);
  abstract public void    pop          ();
  abstract public Object  top          ();
  abstract public boolean Null         ();
}
```

MyStack soll als Unterklasse der Klasse Stack den Datentyp Stack implementieren. Bitte ergänzen Sie die als Leerzeilen freigelassenen Programmteile in den Methoden!

```
public class MyStack extends Stack {
  Cons headElement;
  public MyStack (Cons c) {
    headElement = c;}
  static MyStack nilstack = new MyStack (Cons.nil);
  public static MyStack empty() {
(20)          return nilstack;
  }
  public void push(Object o){
(20)          headElement = new Cons (o, headElement);
  }
  public void pop() {
(30)          if (Null()) –”
          else –
            headElement = headElement.cdr();”
  }
  public Object top() {
(15)          return headElement.car();
  }
  public boolean Null() {
(20)          return headElement == Cons.nil;
  }
  public String toString() {
    return headElement.toString();}
  public static void main(String [] args) {
    MyStack q = empty ();
    System.out.println();
    System.out.println(q.toString());
    System.out.println("5 neue Elemente werden hinzugefuegt ...");
    for(int i=0; i<5; i++) {
      q.push("element "+i);}
    System.out.println();
    System.out.println(q.toString());
    System.out.println();
    System.out.println("Ein Element wird gepoppt ...");
    q.pop();
    System.out.println("Jetzt ist " + q.top().toString() + " oben.");
    System.out.println(q.toString());
    System.out.println();
    System.out.println("Noch 3 Elemente werden gepoppt ...");
    for(int i=0;i<3;i++) {q.pop();}
    System.out.println("Jetzt ist" + q.top().toString() + " oben.");
    System.out.println(q.toString());}}
```

## (60) 7 Programmkorrektur: Quicksort

Aus den Übungen kennen Sie das Programm, dessen Anfang unten angegeben ist. Das Programm ist syntaktisch korrekt und läuft fehlerfrei durch den Compiler. In die Methode `quicksort1` haben sich jedoch vier Programmierfehler eingeschlichen. Korrigieren Sie diese! (Hinweis: Achten Sie u. a. auf die Arrayindizes beim Aufruf!)

```
import java.io.*;
import java.util.*;
public class MyArrays{
  public static boolean smallerorequal (Object a, Object b, Comparator c){
    /* Is a smaller or equal to b compared according to c? */
    return (c.compare(a,b) <= 0);}
  public static void quicksort1 (Object[] a, Comparator c){
    int last = a.length;
    if (0 < last){
      quicksort1 (a, 0, last, c);}}
  public static void quicksort1 (Object[] a, int start, int end, Comparator c){
    /* (start < end) must hold for any call */
    int    L     = start;
    int    R     = end  ;
    Object pivot = a[L];
    Object temp  = a[R];
    while (L < R){
      /* To the left  of L (excl.):       Everything is smaller than
                                                         or equal to pivot
         To the right of R (excl.):       Everything is greater than pivot
         Between L (excl.) and R (excl.): Items to be sorted
                                          besides temp and pivot            */
      if (smallerorequal(temp,pivot,c)){
        a[R] = temp;
        temp = a[++L];}
      else{
        a[R] = temp;
        temp = a[++R];}}
    /* now (L == R) holds */
    /* To the left  of L (excl.): Everything is smaller than or equal to pivot
       To the right of L (excl.): Everything is greater than pivot
       temp is irrelevant because it contains the element under L.           */
    a[L] = pivot;
    if (start < --R){
      quicksort1 (a, start, R    , c);}
    if (++L < end){
      quicksort1 (a, L    , start, c);}}

...
```

Name: Matrikelnr.:

## (170) 8 Sortieren

1. Gegeben sei der Array

ZAHLEN:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 4 | 8 | 2 | 9 | 1 | 6 | 2 | 5 |

Sortieren Sie diese Zahlen nach Größe mit den drei aus der Vorlesung bekannten Suchalgorithmen. Das linke Feld (beim ersten Durchlauf `ZAHLEN[0]`) ist das Pivot-Element. Um Ihre Lösung nachvollziehbar zu machen, schreiben Sie bitte den Array nach jedem Bearbeitungsschritt erneut auf! Hierbei können Sie sich bei jeder Zeile auf das Ausfüllen der geänderten Spalten beschränken.

(50) (a) Quicksort:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 4 | 8 | 2 | 9 | 1 | 6 | 2 | 5 |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Name: Matrikelnr.:

(20) (b) Merge-Sort:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 4 | 8 | 2 | 9 | 1 | 6 | 2 | 5 |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

(20) (c) Exchange-Sort:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 4 | 8 | 2 | 9 | 1 | 6 | 2 | 5 |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Name: Matrikelnr.:

(10) 2. Wie aufwendig ist die Suche? Geben Sie den Zeitaufwand im schlechtesten Fall (worst case complexity) in Landau-O-Notation an!

(a) Quick-Sort: $\mathfrak{O}(n^2)$

(b) Merge-Sort: $\mathfrak{O}(n \log n)$

(c) Exchange-Sort: $\mathfrak{O}(n^2)$

(10) 3. Angenommen man würde in die Algorithmen eine Verbesserung einbauen, die insgesamt 10 Berechnungsschritte spart: Wie verändert sich dann der Zeitaufwand gegenüber (2.)?

Gar nicht, weil asymtotisch irrelevant!

(20) 4. Wieso ist Quick-Sort trotz des Vergleiches in (2.) so schnell?

Erwartungswert von $n \log n$ mit in der Regel noch viel weniger Austauschoperationen. Bei den anderen Such-Algorithmen besteht kein Unterschied zwischen worst case und average complexity.

(20) 5. Unterscheiden Sie graphisch und an einem Beispiel die folgenden Zeitkomplexitäten:

(a) Exponentiell:

Allgemeingültigkeit aussagenlogischer Formeln. (Sonderpunkt, war nicht dran) $2^n$.

(b) Quadratisch:

Exchange-Sort. $n^2$

(c) Linear:

Merge. $n$

(d) Konstant:

Arrayzugriff. $1$

Name: Matrikelnr.:

(20) 6. Berechnen Sie:

$$
\begin{aligned}
O(1) + O(1) &= \mathfrak{O}(1) \\
O(n) + O(1) &= \mathfrak{O}(n) \\
O(n) + O(n) &= \mathfrak{O}(n) \\
n * O(1) &= \mathfrak{O}(n) \\
4 * O(1) + 7 &= \mathfrak{O}(1) \\
O(n) + O(n^2) &= \mathfrak{O}(n^2) \\
n * O(n^2) &= \mathfrak{O}(n^3) \\
n * O(2^n) &= \mathfrak{O}(2^n)
\end{aligned}
$$

Name: Matrikelnr.:

# (40) 9 Klassische Prädikatenlogik 1. Stufe

Gegeben sei eine Signatur $\Sigma$ die aus den Konstanten (= nullstelligen Funktionssymbolen) $\mathsf{Romeo}$, $\mathsf{Julia}$, $\mathsf{Macbeth}$, und dem zweistelligen Prädikat $\mathsf{Liebt}$ besteht.

(10) 1. Kreuzen Sie in der folgenden Tabelle an! Und zwar machen Sie bitte ein Kreuz in der ersten Spalte, wenn die in der jeweiligen Zeile stehende Zeichenfolge ein *Term* der Prädikatenlogik erster Stufe ist; in der zweiten Spalte, wenn sie eine syntaktisch korrekte *Formel* der Prädikatenlogik erster Stufe ist. Für die Terme und Formeln machen sie dann in der entsprechenden Zeile der dritten Spalte ein Kreuz, wenn es sich um einen Term oder eine Formel über der oben angegebenen Signatur $\Sigma$ handelt.

| | ... ist ein Term | ... ist eine Formel | ... gehört zur Signatur $\Sigma$ |
|---:|---|---|---|
| $0$ | | | |
| $0 < 1$ | | | |
| $\mathsf{Romeo}$ | | | |
| $\neg\mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Julia})$ | | | |
| $\mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Macbeth})$ | | | |
| $\mathsf{Liebt}(\mathsf{Liebt}(\mathsf{Julia}, \mathsf{Julia}, \mathsf{Julia}))$ | | | |
| $\mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Macbeth}) < \mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Julia})$ | | | |
| $\forall x.\exists y.\mathsf{Liebt}(x, y)$ | | | |

Bei den folgenden Fragen handelt es sich um Multiple-Choice-Aufgaben. Kennzeichnen Sie die korrekten Antworten mit einem Kreis um den Buchstaben!

(12) 2 Welche der folgenden Formeln sind allgemeingültig?

(a) $\mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Macbeth})$.

(b) $\mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Julia})$.

(c) $\mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Macbeth}) \Rightarrow \mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Macbeth})$.

(d) $\neg\mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Macbeth}) \vee \mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Macbeth})$.

(e) $\mathsf{true}$.

(f) $\mathsf{false} \Rightarrow \mathsf{Liebt}(\mathsf{Romeo}, \mathsf{Macbeth})$.

(18) 3 Welche der folgenden Aussagen ist äquivalent dazu, dass eine Formel $F$ aus einer Menge $\Gamma$ von Formeln (semantisch) folgt?

(a) Für jedes Modell $\mathcal{A}$ von $F$ gibt es eine Formel $B$ in $\Gamma$, so dass $\mathcal{A}$ ein Modell von $B$ ist.

(b) Jedes Modell für alle Formeln in $\Gamma$ ist auch ein Modell von $F$.

(c) Ist $\mathcal{A}$ eine $\Sigma$-Struktur, in der alle Formeln in $\Gamma$ gültig sind, so ist auch $F$ in $\mathcal{A}$ gültig.

(d) Für jedes Modell von $\Gamma$ gibt es ein Modell für $F$, das die gleiche Signatur hat.

(e) Es gibt ein Modell für $\Gamma$, das auch ein Modell für $F$ ist.

(f) $F$ und $\Gamma$ bestehen aus Formeln über der selben Signatur.

## (40) 10 Entscheidbarkeit

Bei den folgenden Fragen handelt es sich um Multiple-Choice-Aufgaben. Kennzeichnen Sie die korrekten Antworten mit einem Kreis um den Buchstaben!

(16) 1. Was ist das Halteproblem?

(a) Eingabe: Ein JAVA-Programm P und ein Eingabefile D.
Frage: Hält das Progamm P mit Eingabe D irgendwann an?

(b) Eingabe: Ein JAVA-Programm P und ein Eingabefile D.
Frage: Wie halte ich das Programm P mit Eingabe D an, wenn es unendlich lange läuft?

(c) Eingabe: Ein JAVA-Programm P und ein Eingabefile D.
Frage: Ist es wahr, dass das Programm P mit Eingabe D unendlich lange läuft?

(d) Eingabe: Ein JAVA-Programm P und ein Eingabefile D.
Frage: Ist es falsch, dass das Programm P mit Eingabe D unendlich lange läuft?

(e) Eingabe: Ein JAVA-Programm P und ein Eingabefile D.
Frage: Wie oft hält das Programm P mit Eingabe D an?

(24) 2. Das Halteproblem ...

(a) ... ist entscheidbar (= Es gibt ein Entscheidungsprogramm).

(b) ... ist unentscheidbar.

(c) ... ist co-semi-entscheidbar (= Es gibt ein Nein-Sager-Programm).

(d) ... ist semi-entscheidbar (= Es gibt ein Ja-Sager-Programm).

(e) ... hat eine aufzählbare Lösungsmenge.