technische universität dortmund

Dr. Lars Hildebrand, Fakultät für Informatik, Lehrstuhl 14

Einführung in die Informatik für Ingenieure und Naturwissenschaftler
Einführung in die Informatik für Wirtschaftsmathematiker
Einführung in die Informatik für Medizinphysiker

Klausur am 19. März 2020

Mit Lösungen

Bitte kreuzen Sie hier an, nach welcher Prüfungsordnung Sie studieren:

- ◯ Bachelor Logistik
- ◯ Bachelor Wirtschaftsingenieurwesen
- ◯ Bachelor Wirtschaftsmathematik
- ◯ Bachelor Medizinphysik

◯ andere: _____

Semester: ◯ erstes

◯ höheres: _____

Hinweise

- Diese Klausur besteht aus 10 Aufgaben. Diese Klausur ist bestanden, wenn 40 der möglichen 100 Punkte erreicht werden.
- Für die Bearbeitung dieser Klausur stehen insgesamt 120 Minuten zur Verfügung.
- **Bevor** Sie mit der Bearbeitung dieser Klausur beginnen, **müssen** Sie auf allen Blättern **Ihren Namen und Ihre Matrikelnummer** eintragen.
- Bei der Bearbeitung der Klausur dürfen **keine Hilfsmittel** verwendet werden.
- Schreiben Sie mit einem dokumentenechten **blauen oder schwarzen Stift**. Korrekturflüssigkeiten und ähnliche Materialien dürfen nicht benutzt werden. Bitte trennen Sie die Blätter nicht.
- Bearbeiten Sie jede Aufgabe möglichst auf der Seite, auf der sie steht. Wenn der Platz nicht reicht, können Sie die Zusatzseiten und nutzen. Notieren Sie in jedem Fall, wo die Lösung zu finden ist.
- Wenn Sie zusätzliche Blätter benötigen, wenden Sie sich an die Aufsicht. Die Nutzung von eigenem Papier ist nicht gestattet.
- Da es verschiedene Sprachgebräuche und unterschiedliche Ausprägungen einzelner Definitionen gibt, werden hier ausdrücklich die Ihnen aus der Vorlesung und den Übungen bekannten Ausdrucksweisen und Realisierungen zu Grunde gelegt.

Viel Erfolg!

Aufgabe	1	2	3	4	5	6	7	8	9	10	$\sum$
Mögliche Punkte	8	8	8	9	10	10	14	10	8	15	100
Erreichte Punkte											

Aufgabe 1 – Binärsystem (8 Punkte)

a) In dieser Aufgabe sollen Sie Zahlen vom Binär- ins Dezimalsystem und umgekehrt umrechnen. In jeder Zeile der unten stehenden Tabelle soll der äquivalente Wert in Binär- und Dezimalschreibweise stehen. Ergänzen Sie die freien Felder der Tabelle entsprechend (es kommen keine negativen Zahlen vor). (6 Punkte)

Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Dezimalzahl
1	1	0	0	0	1	**49**
0	0	1	1	0	1	**13**
1	1	0	0	1	0	**50**
1	0	1	0	0	0	40
1	1	0	1	1	1	55
0	1	0	1	1	1	23

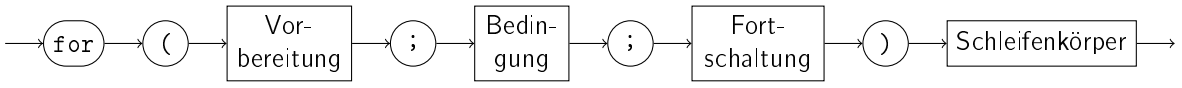
b) Vervollständigen Sie die Wahrheitstabellen für die logischen Verknüpfungen „und“ ($\wedge$) und „exklusiv oder“ ($\otimes$) mit den Eingängen **a**, **b** und dem Ausgang **f**. (2 Punkte)

a	b	f = a $\wedge$ b
false	false	**false**
false	true	**false**
true	false	**false**
true	true	**true**

a	b	f = a $\otimes$ b
false	false	**false**
false	true	**true**
true	false	**true**
true	true	**false**

Aufgabe 2 – Syntaxdiagramm (8 Punkte)

Betrachten Sie folgendes Syntaxdiagramm für `for`-Schleifen.



In der linken Spalte (Quelltext) der folgenden Tabelle sehen Sie Programmfragmente. Kennzeichnen Sie durch Ankreuzen, ob diese Fragmente entsprechend des obigen Syntaxdiagrammes gültig oder ungültig sind. Gehen Sie davon aus, dass der Quelltext, der zu den Platzhaltern Vorbereitung, Bedingung, Fortschaltung und Schleifenkörper gehört, fehlerfrei ist.

(Richtige Antwort: 1 Punkt, falsche Antwort: -1 Punkt, keine Antwort 0 Punkte; die Aufgabe gibt insgesamt mindestens 0 Punkte.)

Quelltext	gültig	ungültig
`for (char c='a'; c < 'Z'; c++){}`	×	◯
`for (int i=0, i-- < 4, i+=2){}`	◯	×
`for (int k=42; k > 0; k--){}`	×	◯
`while (i-- < 4) { i+=2; }`	◯	×
`for {int j=1; j*j < 27513; j*=2}{}`	◯	×
`for (int i=1; j == 1; i++;) System.out.println();`	◯	×
`for (int m=0; m < 9; m++) while (m != 10) { m++; }`	×	◯
`do { m++; } while(m != 10);`	◯	×

Aufgabe 3 – Muster (8 Punkte)

Schreiben Sie öffentliche, statische Methoden, die die folgenden Muster erzeugen. Für jedes Muster müssen zwei ineinander verschachtelte for-Schleifen verwendet werden. Die Schleifen müssen maßgeblich an der Mustererzeugung beteiligt sein. Ausgaben außerhalb der Schleifen, wie auch die Verwendung von Strings mit Ausnahme von einem "*" sind **nicht** zulässig.

a) (2 Punkte)

```
1
12
123
1234
12345
123456
```

Lösung:

```
public static void muster(){
   for(int i = 1; i <= 6; i++) {
      for(int j = 1; j <= i; j++) {
         System.out.print(j);
      }
      System.out.println();
   }
}
```

b) (3 Punkte)

```
*****
****
***
**
*
```

Lösung:

```
public static void muster(){
   for(int i = 5; i >= 1; i--) {
      for(int j = i; j >= 1; j--) {
         System.out.print("*");
      }
      System.out.println();
   }
}
```

c) (3 Punkte)

```
1*3*5*7*9
1*3*5*7*9
1*3*5*7*9
1*3*5*7*9
1*3*5*7*9
```

Lösung:

```
public static void muster(){
   for(int i = 1; i <= 5; i++) {
      for(int j = 1; j <= 9; j++) {
         if(j % 2 == 1) {
            System.out.print(j);
         }
```

```
            else {
               System.out.print("*");
            }
         }
         System.out.println();
      }
   }
```

Aufgabe 4 – Schleifen (9 Punkte)

a) Ersetzen Sie das folgende Programmfragment gleichwertig, sodass anstelle einer for-Schleife eine while-Schleife verwendet wird. (3 Punkte)

```
int a, b;
a = 12;
b = 39;
for (int i=5*a; i<8*b; i= i+10) {
   System.out.println(i);
   b--;
}
```

Lösung:

```
int a, b;
a = 12;
b = 39;
int i = 5 * a;
while (i < 8*b) {
   System.out.println(i);
   b--;
   i = i+10;
}
```

b) Ersetzen Sie das folgende Programmfragment gleichwertig, sodass anstelle einer while-Schleife eine for-Schleife verwendet wird. (3 Punkte)

```
int i = 3;
while (i < 12 && 25 - i > 10) {
   System.out.println(i);
   i = i + 5;
}
```

Lösung:

```
for (int i=3; i<12 && 25-i>10; i = i+5)
   System.out.println(i);
```

c) Ersetzen Sie das folgende Programmfragment gleichwertig, sodass anstelle einer do-while-Schleife eine while-Schleife verwendet wird. (3 Punkte)

```
int x, y;
x = 19;
y = 3;
do {
   x = x - y;
   System.out.println(x);
} while (x > 5);
```

Lösung:

```
int x,y;
x = 19;
y = 3;
x = x - y;
System.out.println(x);
while (x > 5) {
   x = x - y;
   System.out.println(x);
}
```

Aufgabe 5 – Arrays (10 Punkte)

a) Schreiben Sie eine öffentliche, statische Methode add, die für zwei gleich lange int-Arrays a und b ein int-Array erzeugt und zurückgibt. Das Ergebnis soll die elementweise Summe von a und b enthalten. Beispielsweise liefert add(a, b) für die Arrays a = [1,2,3] und b = [10,20,30] das Array [11,22,33].

(5 Punkte)

Lösung:

```
public static int[] add(int[] a, int[] b) {
   int[] result = new int[a.length];
   for (int i=0; i<a.length; i++)
      result[i] = a[i] + b[i];
   return result;
}
```

b) Schreiben Sie eine öffentliche, statische Methode count, die für ein int-Array a und einen int-Wert value die Anzahl der Vorkommen des Wertes value im Array ermittelt und zurückgibt. Beispielsweise liefert count(a, 5) für das Array a = [5,4,5] den Wert 2, während count(a, 1) den Wert 0 liefert.

(5 Punkte)

Lösung:

```
public static int count(int[] a, int value) {
   int occurrences = 0;
   for (int i=0; i<a.length; i++)
      if (a[i] == value)
         occurrences++;
   return occurrences;
}
```

Aufgabe 6 – Rekursion (10 Punkte)

a) Programmieren sie eine öffentliche, statische, rekursive Methode `f`, die die unten abgebildete mathematische Funktion für $x \in \mathbb{N}_0$ und $y \in \mathbb{N}_0$ mit $x \geq y$ umsetzt. (6 Punkte)

$$f(x,y) = \begin{cases} 1 & \text{falls } x = y \text{ oder } y = 0 \\ f(x-1,y) + f(x-1,y-1) & \text{sonst} \end{cases}$$

Wählen Sie für die Methode geeignete Argument- und geeignete Rückgabetypen.

Lösung:

```
public static int f(int x, int y) {
     if (x==y || y == 0) {
          return 1;
     } else return f(x-1, y) + f(x-1, y-1);
}
```

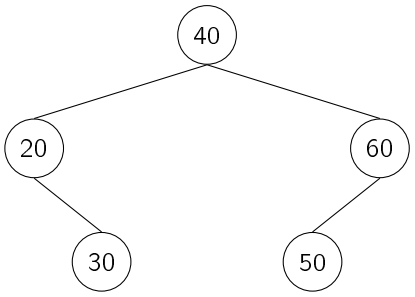
b) Berechnen Sie schrittweise $f(4,2)$. (4 Punkte)

$$f(4,2) = f(3,2) + f(3,1) = \big(f(2,2) + f(2,1)\big) + \big(f(2,1) + f(2,0)\big)$$
$$= \Big(1 + \big(f(1,1) + f(1,0)\big)\Big) + \Big(\big(f(1,1) + f(1,0)\big) + 1\Big)$$
$$= \Big(1 + \big(1+1\big)\Big) + \Big(\big(1+1\big) + 1\Big) = 6$$

Aufgabe 7 – Bäume (14 Punkte)

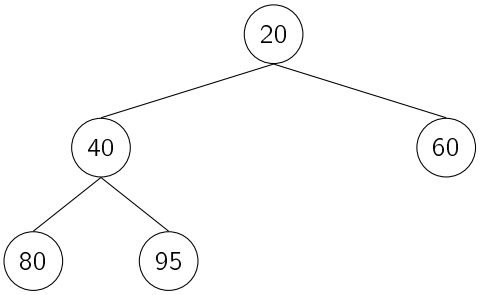
a) Prüfen Sie, ob es sich bei den folgenden Bäume um binäre Bäume und/oder binäre Suchbäume und/oder Heaps handelt. Kreuzen Sie jeweils an, welche Eigenschaften zutreffen.

i) (2 Punkte)



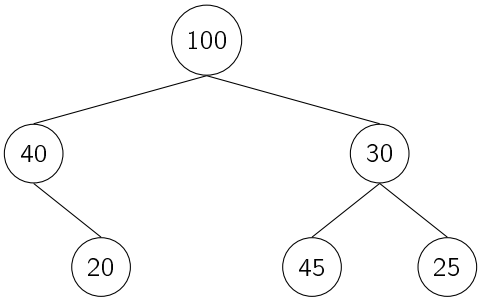
Eigenschaft	
Binärbaum	×
Binärer Suchbaum	×
Heap	○

ii) (2 Punkte)



Eigenschaft	
Binärbaum	×
Binärer Suchbaum	○
Heap	×

iii) (2 Punkte)



Eigenschaft	
Binärbaum	×
Binärer Suchbaum	○
Heap	○

iv) (1 Punkt)

Eigenschaft	
Binärbaum	×
Binärer Suchbaum	×
Heap	×

b) Wenden Sie folgende Durchläufe auf den Bäumen aus Aufgabe 7 a) an: (jeweils 1 Punkt)

i) Postoderdurchlauf auf Baum i)

30, 20, 50, 60, 40

ii) Preoderdurchlauf auf Baum ii)

20, 40, 80, 95, 60

iii) Inoderdurchlauf auf Baum iii)

40, 20, 100, 45, 30, 25

iv) Breitendurchlauf auf Baum ii)

20, 40, 60, 80, 95

c) Kreuzen Sie an, ob die folgenden Aussagen über binäre Bäume wahr oder falsch sind.

(Richtige Antwort: 0,5 Punkte, falsche Antwort: -0,5 Punkte, keine Antwort 0 Punkte; die Aufgabe gibt insgesamt mindestens 0 Punkte.) (3 Punkte)

Aussage	**wahr**	**falsch**
Jeder binäre Suchbaum ist ein rechtsvollständiger binärer Baum.	◯	×
Jeder Heap ist ein rechtsvollständiger binärer Baum.	◯	×
Jeder Inorder-Durchlauf eines Heaps ist aufsteigend sortiert.	◯	×
Jeder Teilbaum eines Binärbaums ist selber ein Binärbaum.	×	◯
Jeder Teilbaum eines binären Suchbaums ist selber ein binärer Suchbaum.	×	◯
Jeder Teilbaum eines Heaps ist selber ein Heap.	×	◯

Aufgabe 8 – Fehlerarten und Eigenschaften von Algorithmen (10 Punkte)

a) Geben Sie jeweils ein Beispiel für die folgenden Fehlerarten an. Erläutern Sie kurz ihr Beispiel.

i) Syntaxfehler (2 Punkte)

ii) Laufzeitfehler (2 Punkte)

iii) Semantikfehler (2 Punkte)

b) Um welche Fehlerart handelt es sich bei einer unabsichtlichen Endlosschleife? Begründen Sie ihre Antwort. (2 Punkte)

Semantikfehler, da es zu keinem Abbruch kommt und die Endlosschleife unbeabsichtigt war.

c) Entscheiden Sie für jedes der folgenden Probleme, ob es sich dabei um ein Einzelproblem oder um eine Problemklasse handelt. Begründen Sie!

i) Wie viele Tage hat das Jahr 2020. (1 Punkt)

Einzelproblem

Alle Angaben zum sofortigen berechnen da.

ii) Wie häufig kommt der Buchstabe 'a' in der Zeichenkette "Banane" vor. (1 Punkt)

Einzelproblem

Alle Angaben zum sofortigen berechnen da.

Aufgabe 9 – Klassenvariablen und -methoden vs. Objektvariablen und -methoden (8 Punkte)

Betrachten Sie das folgende Programm. Es enthält sowohl Klassenvariablen und -methoden als auch Objektvariablen und -methoden. Manche Zugriffe und Methodenaufrufe im Programm sind nicht erlaubt. Notieren Sie auf den Linien neben dem Programmtext, ob der jeweilige Zugriff oder der jeweilige Methodenaufruf **erlaubt** oder **nicht erlaubt** ist. (Richtige Antwort: 0,5 Punkte, falsche Antwort: -0,5 Punkte, keine Antwort 0 Punkte; die Aufgabe gibt insgesamt mindestens 0 Punkte.)

```
1  class Kreditkarte {
2    private String inhaberName;
3    private int nummer;
4    private static int anzahlKunden;
5
6    public void gibInhaberAus() {
7      if (anzahlKunden > 0) {                          erlaubt
8       System.out.println(inhaberName);                erlaubt
9     }
10   }
11
12   public static int gibAnzahl() {
13     if (nummer > 0)                                   nicht erlaubt
14       return anzahlKunden;                            erlaubt
15     else return 0;
16   }
17
18   public static void main(String[] args) {
19     anzahlKunden = 0;                                 erlaubt
20     inhaberName = "Petra Mustermann";                 nicht erlaubt
21     int x = gibAnzahl();                              erlaubt
22     gibInhaberAus();                                  nicht erlaubt
23
24     Kreditkarte testKarte = new Kreditkarte();
25     testKarte.anzahlKunden += 1;                      erlaubt
26     testKarte.inhaberName = "Peter Mustermann";       erlaubt
27     System.out.println(testKarte.gibAnzahl());        erlaubt
28     testKarte.gibInhaberAus();                        erlaubt
29
30     x = Kreditkarte.anzahlKunden;                     erlaubt
31     Kreditkarte.nummer = 5621345;                     nicht erlaubt
32     System.out.println(Kreditkarte.gibAnzahl());      erlaubt
33     Kreditkarte.gibInhaberAus();                      nicht erlaubt
34   }
35 }
```

Aufgabe 10 – Prüfung (15 Punkte)

Es sollen drei öffentliche Klassen konzipiert werden: Eine Oberklasse „Pruefung“ und zwei Unterklassen „UnbenotetePruefung“ und „BenotetePruefung“.

a) Erstellen Sie die abstrakte Oberklasse 'Pruefung', sodass sie folgende Bedingungen erfüllt. (5 Punkte)

i) Alle Methoden (auch der Konstruktor) sind öffentlich.

ii) Zu einer Prüfung gehören die Attribute 'titel' (Name der Veranstaltung), 'versuch' (Nummer des Prüfungsversuchs) und 'bestanden' (wahr/falsch). Auf das Attribut 'titel' kann nur innerhalb der Oberklasse zugegriffen werden, die anderen Attribute sind innerhalb der Oberklasse und aller Unterklassen sichtbar.

iii) Dem Konstruktor wird der Name der Veranstaltung übergeben. Eine Prüfung beginnt mit dem ersten Versuch und ist zunächst nicht bestanden.

iv) Die Klasse hat eine Methode 'teilnahmeMoeglich', die wahr zurückgibt, wenn die Prüfung noch nicht bestanden wurde und der nächste Versuch höchstens der dritte ist.

v) Die Klasse hat eine Methode 'falleDurch', die die Anzahl der Versuch um eins erhöht, falls die Teilnahme möglich ist. Andernfalls ändert sich nichts.

vi) Die Klasse hat eine Methode 'ausgabe', die den Titel der Prüfung ausgibt und ob sie bereits bestanden oder noch nicht bestanden wurde.

Lösung:

```
public abstract class Pruefung {
   private String titel;
   protected int versuch;
   protected boolean bestanden;

   public Pruefung(String titel) {
      this.titel = titel;
      this.versuch = 1;
      this.bestanden = false;
   }

   public boolean teilnahmeMoeglich() {
      return !bestanden && versuch <= 3;
   }

   public void falleDurch() {
      if (teilnahmeMoeglich())
      versuch++;
   }

   public void ausgabe() {
      System.out.print("Prüfung " + titel + " ");
      if (bestanden)
      System.out.println("bestanden");
      else
      System.out.println("nicht bestanden");
   }
}
```

b) Erstellen Sie eine öffentliche Klasse 'UnbenotetePruefung', die von 'Pruefung' erbt. Diese Klasse hat keine zusätzlichen Attribute und soll die folgenden Bedingungen erfüllen. (3 Punkte)

i) Alle Methoden (auch der Konstruktor) sind öffentlich.

ii) Die Klasse hat eine zusätzliche Methode 'bestehe', die das Attribut 'bestanden' aktualisiert, falls die Teilnahme möglich ist. Andernfalls ändert sich nichts.

Lösung:

```
public class UnbenotetePruefung extends Pruefung {
   public UnbenotetePruefung(String titel) {
      super(titel);
   }

   public void bestehe() {
      if (teilnahmeMoeglich())
      bestanden = true;
   }
}
```

c) Erstellen Sie eine öffentliche Klasse 'BenotetePruefung', die ebenfalls von 'Pruefung' erbt. Diese Klasse hat ein zusätzliches Attribut 'note', indem die Note * 100 gespeichert wird. Beispielsweise entsprechen die Werte 270 und 500 den Noten 2,7 und 5,0. Die Klasse soll die folgenden Bedingungen erfüllen.

(4 Punkte)

i) Alle Methoden (auch der Konstruktor) sind öffentlich.

ii) Anfangs wird als Note 0 gesetzt.

iii) Die Klasse hat eine Methode 'bestehe', die den Notenwert übergeben bekommt. Falls die Teilnahme möglich ist, werden zwei Fälle unterschieden. Wenn der Notenwert kleiner als 500 ist, dann wird die Note gespeichert und die Prüfung als bestanden verzeichnet. Andernfalls wird nur der Versuchszähler um eins erhöht. Ist die Teilnahme nicht möglich, passiert nichts.

iv) Die Klasse hat eine Methode 'ausgabe', die zunächst die Ausgabe der Oberklasse erzeugt. Falls die Prüfung bereits bestanden wurde, wird außerdem die erreichte Note ausgegeben.

Lösung:

```
public class BenotetePruefung extends Pruefung {
   private int note;

   public BenotetePruefung(String titel) {
      super(titel);
      this.note = 0;
   }

   public void bestehe(int note) {
      if (teilnahmeMoeglich()) {
         if (note < 500) {
            this.bestanden = true;
            this.note = note;
         }
         else {
         this.versuch++;
         }
      }
   }

   public void ausgabe() {
      super.ausgabe();
      if (bestanden)
         System.out.println("Note: " + this.note);
      }
   }
}
```

d) Was wird durch die Ausführung des folgenden Quelltextes auf dem Bildschirm ausgegeben?

(3 Punkte)

```
public class Pruefungen {
   public static void main(String[] args) {
      UnbenotetePruefung p = new UnbenotetePruefung("Englischkurs");
      p.falleDurch();
      p.bestehe();
      p.ausgabe();

      BenotetePruefung k = new BenotetePruefung("EINI");
      k.bestehe(130);
      k.ausgabe();

      BenotetePruefung m = new BenotetePruefung("HöMa1");
      m.falleDurch();
      m.ausgabe();
   }
}
```