

01. August 2008

Wiederholungs-Klausur zu Informatik I: Algorithmieren und Programmieren

Musterlösungen

Bitte gut lesbar ausfüllen !

Name: Bachmann	Vorname: Peter
Matrikel: 66666666	Studiengang: alles Informatik-nahe
Tutor: ich selber	Raum: EHS 211
Sitzreihe: 1	Sitzplatz: 1

Achtung:

- Alle Antworten tragen Sie bitte direkt hinter der Aufgabenstellung in die dafür vorgesehenen Freiräume bzw. Schemata ein.
- Nebenrechnungen machen Sie bitte auf Ihrem eigenen Papier, das aber nicht eingesammelt und deshalb nicht berücksichtigt wird!
- Sie können alle Hilfsmittel in schriftlicher Form benutzen, d.h. z.B. Ihre Aufzeichnungen, das Skript zur Vorlesung, Bücher usw., aber keine Computer und keine Handys.

Bewertung:

1	2	3	4	5	6	7	8	9	10	Summe
10	10	10	10	10	10	10	10	10	10	100

1. Informationsdarstellung (10 Punkte)

1.1 (5 Punkte) Gegeben sind Zahlen z der Darstellung zur Basis 10. Konvertieren Sie diese in Zahlen z' der Darstellung zur Basis b .

z	b	z'
1007	2	1111101111
203	3	21112
219	4	3123
1007	8	1757
1007	16	3EF

1.2 (5 Punkte) Gegeben sind Zahlen z der Darstellung zur Basis b . Konvertieren Sie diese in Zahlen z' der Darstellung zur Basis 10.

b	z	z'
2	1101	13
3	222	26
4	1233	111
8	1360	752
16	1ba	442

2. Induktion (10 Punkte)

2.1 (8 Punkte) Eine Folge s_n von natürlichen Zahlen sei induktiv definiert durch:

$$s_0 = s_1 = 0 \quad \text{und} \\ s_n = 3s_{n-1} - 2s_{n-2} + 2 \text{ für } n > 1.$$

2.1.1 (2 Punkte) Bestimmen Sie die Polynome $g(z)$ und $h(z)$.

$$g(z) = 1 - 3z + 2z^2$$

$$h(z) = 1 - z$$

2.1.2 (2 Punkte) Bestimmen Sie die Nullstellen des Polynoms $g(z) \cdot h(z)$ (einschließlich ihrer Vielfachheiten).

$$\omega_1 = \frac{1}{2}, \mu_1 = 1, \omega_2 = 1, \mu_2 = 2$$

2.1.3 (2 Punkte) Bestimmen Sie ein Gleichungssystem für die Berechnung der A_{ij} .

Allgemein gilt:

$$s_n = 2^n A_{10} + A_{20} + nA_{21},$$

also

$$n = 0: \quad 0 = A_{10} + A_{20}$$

$$n = 1: \quad 0 = 2A_{10} + A_{20} + A_{21}$$

$$n = 2: \quad 2 = 4A_{10} + A_{20} + 2A_{21}$$

2.1.4 (1 Punkt) Bestimmen Sie eine geschlossene Darstellung für s_n .

$$s_n = 2^{n+1} - 2(n+1)$$

2.1.5 (1 Punkt) Wie muss der Wert von s_1 gewählt werden, damit $s_2 = 5$ gilt?.

$$s_1 = 1$$

2.2 (2 Punkte) Die Funktion $f: \mathbb{N} \rightarrow \mathbb{N}$ sei induktiv definiert durch:

$$f(0) = 1 \quad \text{und} \quad f(n+1) = 2 * f(n) + n^2 - 2 * n - 1.$$

Beweisen Sie, dass gilt: $f(n) = 2^n - n^2$.

$$\text{Induktionsanfang: } f(0) \stackrel{D}{=} 2^0 - 0^2 = 1 - 0 = 1$$

$$\text{Induktionsschritt: } f(n+1) \stackrel{D}{=} 2 * f(n) + n^2 - 2 * n - 1 \stackrel{IA}{=} 2 * (2^n - n^2) + n^2 - 2 * n - 1 \\ = 2^{n+1} - n^2 - 2 * n - 1 = 2^{n+1} - (n+1)^2$$

3. Grammatiken (10 Punkte)

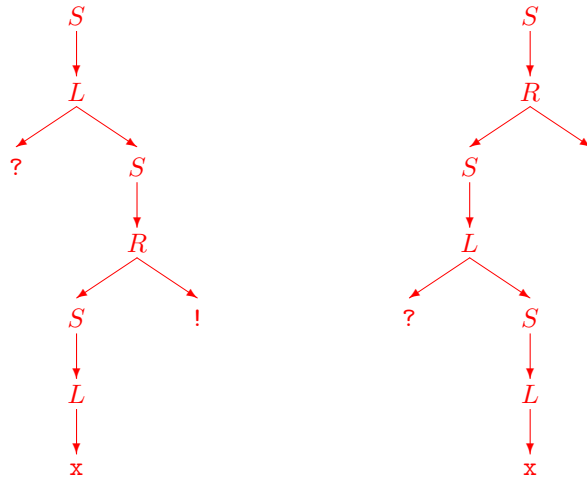
3.1 (6 Punkte) In der folgenden BNF sind die Metasymbole aus der Menge $\{S, L, R\}$ *kursiv* gesetzt, die Grundsymbole aus der Menge $\{!, ?, \%, x, y\}$ in **Schreibmaschinenschrift**:

$$\begin{aligned} S &::= L \mid R \mid L \% R \\ L &::= x \mid ? S \\ R &::= y \mid S ! \end{aligned}$$

3.1.1 (4 Punkte) Geben Sie alle kürzesten Texte an, die durch die BNF aus dem Satzsymbol S erzeugt werden und in denen jedes Grundsymbol mindestens einmal auftritt:

$$?x\%y! \quad ?y\%x! \quad ?x!\%y \quad x!\%?y!$$

3.1.2 (2 Punkte) Geben Sie alle Syntaxbäume für den Text $? x !$ an, wobei das Satzsymbol wieder S ist.



3.2 (4 Punkte) Es sei $W \subseteq \{a, b\}^*$ die Menge aller Worte, für die in jedem Suffix die Anzahl der Zeichen a größer gleich der Anzahl der Zeichen b ist. Geben Sie eine Grammatik für W an.

$$W ::= \varepsilon \mid W a \mid b W a \mid W W$$

4. Sortieren (10 Punkte)

4.1 (5 Punkte) Mittels **mergesort** soll die Folge 1 7 6 5 2 8 4 3 aufsteigend (kleinstes Element steht links!) sortiert werden.

4.1.1 (3 Punkte) Tragen Sie dabei die Folge, die jeweils bei einem Mischvorgang aus zwei Teilfolgen entsteht, in die unmittelbar unter den Teilfolgen liegende Zeile ein.

1	7	6	5	2	8	4	3
1	7	5	6	2	8	3	4
1	5	6	7	2	3	4	8
1	2	3	4	5	6	7	8

4.1.2 (1 Punkt) Wieviele Vergleiche sind insgesamt durchzuführen?

Antwort = 17

4.1.3 (1 Punkt) Wieviele Vergleiche sind beim Sortieren der umgekehrten Folge 3 4 8 2 5 6 7 1 durchzuführen?

Antwort = 17

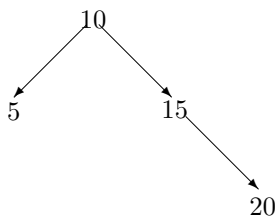
4.2 (5 Punkte) Wenn die Folge 2 4 3 6 10 11 1 9 7 mittels **heapsort** sortiert werden soll, so muss zunächst daraus mittels **heapify** ein **heap** erzeugt werden. Der **heap** sei so gestaltet, dass die Markierung des Vaters grösser als die Markierungen der beiden Söhne ist. Geben Sie in der folgenden Tabelle für jede notwendige Vertauschung bei **heapify** in der Folge eine neue Zeile an.

2	4	3	6	10	11	1	9	7
2	4	3	9	10	11	1	6	7
2	4	11	9	10	3	1	6	7
2	10	11	9	4	3	1	6	7
11	10	2	9	4	3	1	6	7
11	10	3	9	4	2	1	6	7

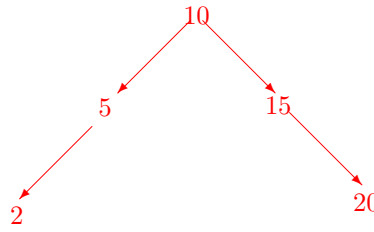
5. Suchbäume (10 Punkte)

5.1 (5 Punkte) Geben Sie jeweils den AVL-Baum an, der entsteht, nachdem in dem entsprechenden linken Baum die zugehörige Operation ausgeführt wurde:

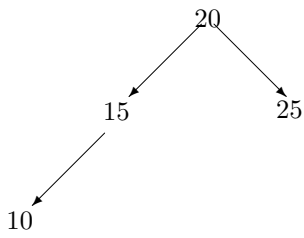
5.1.1 (1 Punkt):



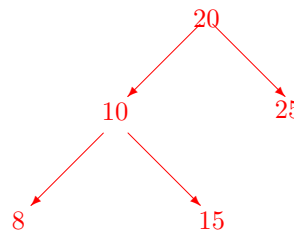
einfügen 2:



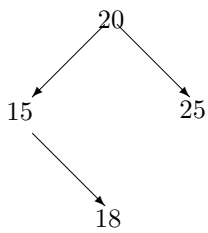
5.1.2 (1 Punkt):



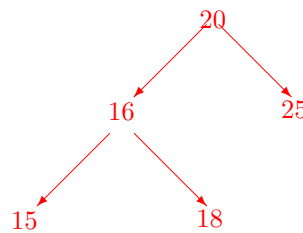
einfügen 8:



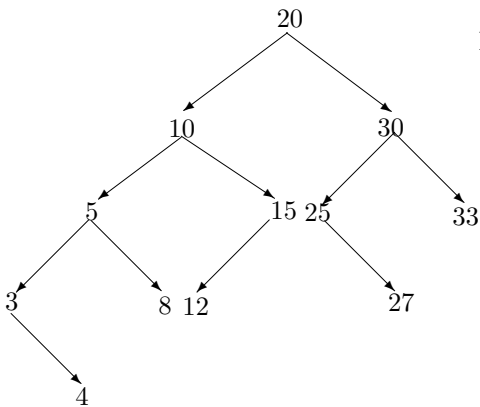
5.1.3 (1 Punkt):



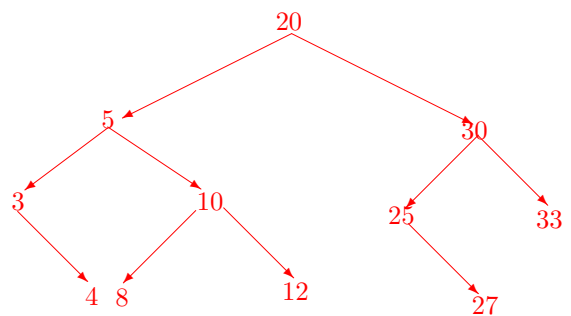
einfügen 16:



5.1.4 (2 Punkte):

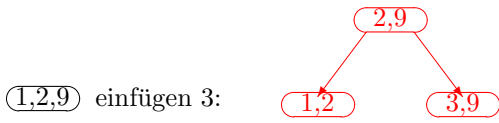


löschen 15:

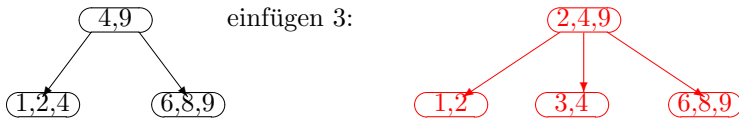


5.2 (5 Punkte) Geben Sie jeweils den (2,3)-Baum an, der entsteht, nachdem in dem entsprechenden linken Baum die zugehörige Operation ausgeführt wurde:

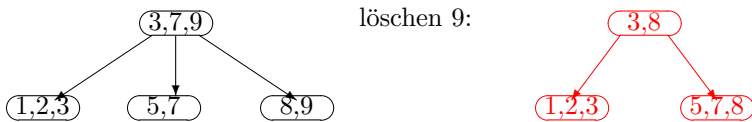
5.2.1 (1 Punkt):



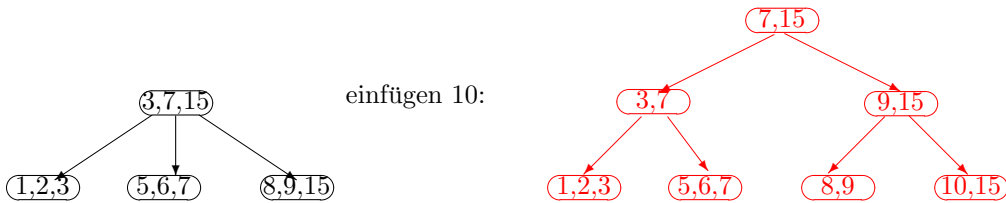
5.2.2 (1 Punkt):



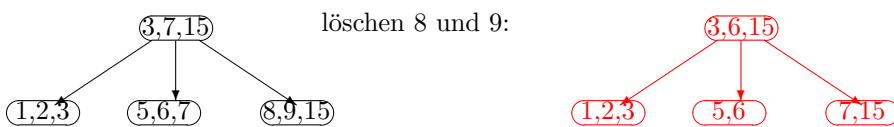
5.2.3 (1 Punkt):



5.2.4 (1 Punkt):



5.2.5 (1 Punkt):



6. Kürzeste Wege (10 Punkte)

Gegeben ist der gerichtete Graph mit

- der Knotenmenge $N = \{1, 2, 3, 4, 5, 6, 7\}$,
- der Kantenmenge $E = \{(1, 2), (1, 3), (2, 4), (2, 5), (3, 2), (3, 5), (4, 7), (5, 4), (5, 6), (5, 7), (6, 4), (6, 7)\}$ sowie
- der Kostenfunktion $\phi : E \rightarrow \mathbb{N}$, die durch die folgende Tabelle 1 beschrieben ist:

Tabelle 1

	(1,2)	(1,3)	(2,4)	(2,5)	(3,2)	(3,5)	(4,7)	(5,4)	(5,6)	(5,7)	(6,4)	(6,7)
ϕ	1	5	7	2	2	2	2	4	1	4	3	2

6.1 (5 Punkte) Nach dem Verfahren von Dijkstra soll ein kürzester Weg vom Knoten 1 zum Knoten 7 bestimmt werden. Dabei werden, entsprechend der Induktion über die verbrauchte Zeit (bzw. Kosten), schrittweise Knoten erreicht, die dann in eine Umgebung aufgenommen werden. In der Tabelle 2 sind in der ersten Zeile die Induktionsschritte angegeben. In den weiteren Zeilen tragen Sie unter den jeweiligen Induktionsschritt den erreichten Knoten und die dabei verbrauchte Zeit (Kosten) ein.

Tabelle 2

Induktionsschritt	0	1	2	3	4	5
erreichter Knoten	1	2	5	6	3	7
verbrauchte Zeit (Kosten)	0	1	3	4	5	6

6.2 (5 Punkte) Nach dem Verfahren von Ford/Moore sollen alle kürzesten Wege vom Knoten 1 zu allen anderen Knoten bestimmt werden. Dabei werden, entsprechend der Induktion über die Anzahl der benutzten Kanten, schrittweise Knoten erreicht. In der Tabelle 3 sind in der ersten Zeile die Induktionsschritte i angegeben, in der ersten Spalte die Knoten n . In den Einträgen $T(n, i)$ tragen Sie jeweils die minimale Zeit (Kosten) ein, die gebraucht wird, um vom Knoten 1 zum Knoten n zu gelangen, wenn man i viele Kanten betreten darf. Ein leerer Eintrag bedeutet, dass man unter Benutzung von i vielen Kanten nicht vom Knoten 1 zum Knoten n gelangen kann.

Tabelle 3

$n \setminus i$	0	1	2	3	4	5
1	0	0	0	0	0	0
2		1	1	1	1	1
3		5	5	5	5	5
4			8	7	7	7
5			3	3	3	3
6				4	4	4
7				7	6	6

7. HASKELL (10 Punkte)

Gegeben ist folgende Haskell-Definition:

```
f :: [Int] -> Int
f(a:b:c:d) = 3*a + f(b:c:d)
f(a:b:c)   = 2*a + f(b:c)
f(a:b)     = 1*a + f b
f a        = 0
```

Bestimmen Sie

$$f [5] = 5$$

$$f [1,2] = 4$$

$$f [2,1] = 5$$

$$f [4,3,2,1] = 26$$

$$f [5,4,3,2,1] = 41$$

8. C++/Vereinbarung und Iteration (10 Punkte)

8.1 (5 Punkte) Gegeben sind die folgenden Deklarationen:

```
unsigned a[]={2,1,3,4};  
struct S {int a,b; float f; } s;  
S * ps;
```

Kreuzen Sie in der folgenden Tabelle an, ob die angegebenen Ausdrücke korrekt gebildet sind, oder nicht.

Achtung: Falsche Kreuze ergeben Punktabzug! Wenn Sie sich unsicher sind, kreuzen Sie besser nicht an!

	korrekt	nicht korrekt
<code>a[3]=a*a[1];</code>		×
<code>s.a=5;</code>	×	
<code>(*ps).a=a[0];</code>	×	
<code>s.b=ps.a;</code>		×
<code>s.a+=a[2];</code>	×	

8.2 (5 Punkte) Gegeben ist die folgende `main`-Funktion. Tragen Sie hinter den Kommentarzeilen die nach Abarbeitung der `for`-Anweisung erreichten Werte des Feldes `a` ein.

```
main()  
{  
    int a[]={0,1,2,3,4,5};  
    for(unsigned i=0;i<6;++i)  
        switch(i)  
        {  
            case 0: ++a[0];  
            case 1: ++a[1];  
            case 2: ++a[2]; break;  
            default: if(i==4) continue; a[i]+=i;  
        }  
  
    //a[1]= 3  
    //a[2]= 5  
    //a[3]= 6  
    //a[4]= 4  
    //a[5]= 10  
}
```

9. C++/Funktionen (10 Punkte)

In dem folgenden C++-Programm tragen Sie hinter dem Zeilenkommentar den beim entsprechenden Funktionsaufruf zurückgegebenen Wert ein.

```
unsigned f(unsigned * a, unsigned l, unsigned m=1)
{
    if(!l) return 0;
    m=(m<3?m+1:0);
    return m*a[0]+f(&a[1],l-1,m);
}
main()
{
    unsigned a1[]={9},
             a2[]={9,8},
             a3[]={8,9},
             a4[]={7,8,9},
             a5[]={7,8,9,9,9};

    f(a1,1);//Rückgabe= 18
    f(a2,2);//Rückgabe= 42
    f(a3,2);//Rückgabe= 43
    f(a4,3);//Rückgabe= 38
    f(a5,5);//Rückgabe= 65
}
```

10. C++/Programmierung (10 Punkte)

Bubblesort ist ein einfaches Sortierverfahren. Es lässt sich in zwei Phasen unterteilen:

Phase 1: In einem Durchlauf durch das zu sortierende Feld **a** werden alle benachbarten Feldelemente betrachtet und verglichen, ob $a[i] \leq a[i+1]$. Wenn das nicht der Fall ist, wird $a[i]$ mit $a[i+1]$ vertauscht.

Phase 2: Die Phase 1 wird solange wiederholt, bis keine Vertauschung mehr erfolgte.

10.1 (6 Punkte) Implementieren Sie eine Funktion `bubble`, die die Phase 1 realisiert und zugleich die Anzahl der beim Durchlauf vorgenommenen Vertauschungen zurückgibt. Geben Sie dazu für den unten angegebenen Rahmen der Funktion `bubble` einen entsprechenden Funktionskörper an. Es ist **a** das Feld und **l** dessen Länge.

```
unsigned bubble(unsigned * a, unsigned l)
```

```
{
    unsigned x=a[0];
    if(l<2) return 0;
    if(x<a[1]) return bubble(&a[1],l-1);
    a[0]=a[1];a[1]=x;
    return bubble(&a[1],l-1)+1;
}
```

10.2 (4 Punkte) Geben Sie unter Nutzung der Funktion `bubble` für den unten angegebenen Rahmen der Funktion `bubblesort` einen Funktionskörper an, wobei `bubblesort` die Phase 2 realisieren soll und die Anzahl der vorgenommenen Vertauschungen in allen Durchlaufungen zurückgibt.

```
unsigned bubblesort(unsigned * a, unsigned l)
```

```
{
    unsigned anz=0,ba;
    while(ba=bubble(a,l))anz+=ba;
    return anz;
}
```