

FH Aachen, Standorte Jülich und Köln  
RWTH Aachen, ITC; FZ Jülich, JSC

Hans Joachim Pflug ( RWTH Aachen), Jörg Striegnitz (FH Aachen),

Klausur zum Modul „Algorithmen“ im Studiengang „Scientific  
Programming“ im SS 2021 am 19.7.2021

Klausur zum Modul „Algorithmen, Datenstrukturen und  
Theoretische Informatik“ im Studiengang „Angewandte Mathematik  
und Informatik“ im SS 2021 am 19.7.2021

Name:

Vorname:

Matrikelnummer:

Unterschrift:

				Klausurversion
Aufgabe	1)	<input type="text"/>	(10)	
Aufgabe	2)	<input type="text"/>	(8)	
Aufgabe	3)	<input type="text"/>	(8)	
Aufgabe	4)	<input type="text"/>	(8)	
Aufgabe	5)	<input type="text"/>	(8)	
Aufgabe	6)	<input type="text"/>	(14)	
Aufgabe	7)	<input type="text"/>	(14)	
Aufgabe	8)	<input type="text"/>	(15)	A <input type="checkbox"/> B <input type="checkbox"/>
Aufgabe	9)	<input type="text"/>	(15)	A <input type="checkbox"/> B <input type="checkbox"/>
maximale Summe:			100	
Gesamtpunkte:		Note:		

Hinweis: Die Versionen A und B unterscheiden sich hinsichtlich der Aufgaben 8 und 9. Sie müssen sich pro Aufgabe für eine Klausurversion entscheiden. Kreuzen Sie Ihre Wahl auf dem Deckblatt an. Bei fehlendem Eintrag wird Version A bewertet.

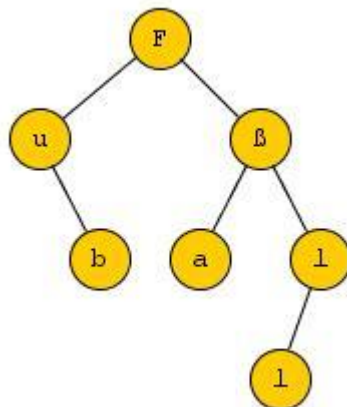
**Aufgabe 1:**

Notieren Sie die Antworten in ein oder mehreren Stichworten oder einem kurzen Satz:

- a) Kreuzen Sie an, welche der nachfolgenden Zeichenketten vom regulären Ausdruck  $(a^*|ab)^*(a|b)$  erkannt wird.

Zeichenkette	wird erkannt
aaaaaa	<input type="checkbox"/>
aabbab	<input type="checkbox"/>
baaabb	<input type="checkbox"/>
aababb	<input type="checkbox"/>

- b) Welche Durchlauf-Strategie erzeugt für den nachfolgenden Baum die Ausgabe Fußball?



- c) Welche zwei wichtigen Operationen besitzt die Datenstruktur „Prioritätswarteschlange“?
- d) Welche Datenstruktur benötigt man beim Durchlauf eines Baums in Level-Order?
- e) Was bedeutet das „doppelt“ in der doppelt verketteten Liste?
- f) Wie ist der AVL-Index definiert?

- g) Zeichnen Sie einen linksvollständigen Binärbaum mit Höhe 3.
  
  
  
  
  
  
  
  
  
  
- h) Nennen Sie ein Verfahren, mit dem man einen minimalen Spannbaum bestimmen kann.
  
  
  
  
  
  
  
  
  
  
- i) Was sind die drei Grundoperationen für reguläre Ausdrücke?
  
  
  
  
  
  
  
  
  
  
- j) Was ist beim doppelten Hashing der Sinn der beiden Hashfunktionen?

Suchen Sie in dem Text  
alle vorkommenden Muster

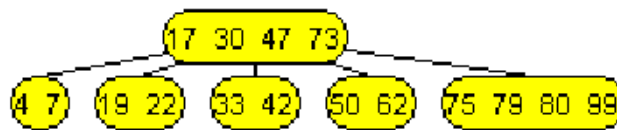
**Achtung:** Suchen Sie alle vorkommenden Muster, nicht nur das erste Muster.

- b) Tragen Sie die wesentlichen Schritte des Suchvorgangs in folgendes Diagramm ein:

[illegible]

**Aufgabe 3:**

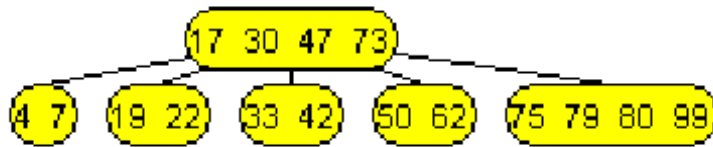
Betrachten Sie den folgenden B-Baum der Ordnung 2:



**Wichtig!** Die Aufgabenteile a-d) verwenden als Ausgangspunkt immer diesen Baum!

- Entfernen Sie aus dem B-Baum den Wert 73 und zeichnen Sie den entstehenden B-Baum..
- Entfernen Sie aus dem **ursprünglichen** B-Baum den Wert 62 und zeichnen Sie den entstehenden B-Baum.

Betrachten Sie den folgenden B-Baum der Ordnung 2:



**Wichtig!** Die Aufgabenteile a-d) verwenden als Ausgangspunkt immer diesen Baum!

- c) Entfernen Sie aus dem **ursprünglichen** B-Baum den Wert 33 und zeichnen Sie den entstehenden B-Baum.
  
- d) Fügen Sie in den **ursprünglichen** B-Baum den Wert 88 ein und zeichnen Sie den entstehenden B-Baum.

Sortieren Sie das folgende Feld mit Quick-Sort in aufsteigende Reihenfolge. Dabei sollte als Pivot-Element immer das am weitesten rechts liegende Element gewählt werden. Teilfelder von 2 Elementen brauchen Sie nicht mehr mit Quick-Sort sortieren - hier dürfen Sie die beiden Elemente gegebenenfalls einfach tauschen. Bitte markieren Sie in jedem Schritt das Pivot-Element, die Vertauschungen und die Grenzen.

[illegible]

**Aufgabe 5:**

d	A	B	C	D
A				
B				
C				
D				

p	A	B	C	D
A				
B				
C				
D				

d	A	B	C	D
A				
B				
C				
D				

p	A	B	C	D
A				
B				
C				
D				

d	A	B	C	D
A				
B				
C				
D				

p	A	B	C	D
A				
B				
C				
D				

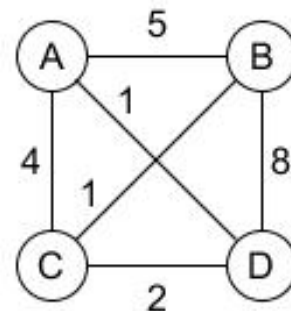
d	A	B	C	D
A				
B				
C				
D				

p	A	B	C	D
A				
B				
C				
D				

d	A	B	C	D
A				
B				
C				
D				

p	A	B	C	D
A				
B				
C				
D				

Betrachten Sie den folgenden Graphen G:



Berechnen Sie mit Hilfe des Floyd-Warshall-Algorithmus die Längen aller kürzesten Pfade in G. Geben Sie hierzu die Werte der Distanz-Matrix und der Vorgänger-Matrix jeweils nach jedem Durchlauf der Hauptschleife an.

Die Diagonal-Elemente haben zu Beginn den Wert unendlich.



**Aufgabe 6:**

Gegeben sei das Code-Fragment eines vermeintlichen Sortier-Algorithmus:

```
public static void foo(int[] a) {
    int i = 0;
    while (i < a.length - 1) {
        int s = 0;
        for (int j = i + 1; j < a.length; j++) {
            //Schlüsselvergleich
            if (a[j] < a[i]) {
                s++;
            }
        }
        if (s != 0) {
            //Tausche a[i] und a[s+i]
            int v = a[i];
            a[i] = a[s + i];
            a[s + i] = v;
        } else {
            i++;
        }
    }
}
```

- a) Wenden Sie die Funktion `foo` auf das folgende Feld an und geben Sie den Inhalt von `a` nach jeder Vertauschung an.

10	2	6	31	5	9	14	7

- b) Sortiert `foo` beliebige Felder? Begründen Sie Ihre Antwort.  
 c) Beschreiben Sie einen worst- und einen best case für

- die Zahl der Schlüsselvergleiche
- die Zahl der Vertauschungen

Verwenden Sie die O-Notation, um für beide Fälle (worst- und best-case) kleinste obere Schranken in Abhängigkeit von der Feldgröße anzugeben.

- d) Optimieren Sie `foo` so, dass die Anzahl der Schlüsselvergleiche (bei gleichbleibender O-Klasse) im worst case sinkt.

**Tipp:** Überlegen Sie sich eine zusätzliche Abbruchbedingung für die while-Schleife.

**Aufgabe 7:**

Gegeben sei ein binärer Suchbaum:

```
public class BinarySearchTree {  
  
    public class Node {  
        public int val;  
        public Node left;  
        public Node right;  
    }  
  
    private Node root;  
    ...  
}
```

Fügen Sie der Klasse BinarySearchTree eine Methode

```
public int indexOf(int val)
```

hinzu. Diese Methode gibt den Index des Werts `val` im Suchbaum zurück. Der kleinste Wert des Suchbaums hat den Index 0. Die restlichen Baumelemente sind nach aufsteigendem Wert bzw. in In-Order-Reihenfolge durchnummeriert.

Falls `val` im Baum nicht vorhanden ist, wird -1 zurückgegeben.

Falls `val` mehrfach vorhanden ist, wird der kleinste Index zurückgegeben.

**Aufgabe 8 (Version A):**

Eine natürliche Zahl sei eine *vollständige Primzahl*, wenn sowohl die erste Ziffer, als auch die ersten beiden Ziffern, die ersten drei Ziffern usw. Primzahlen sind.

Beispiel:

Die Zahl 37139 ist eine vollständige Primzahl, da die Zahlen 3, 37, 371, 3713 und 37139 Primzahlen sind. Die Anzahl der vollständigen Primzahlen ist endlich und alle sind kleiner als 100.000.000.

Schreiben Sie eine Funktion

```
public static ArrayList<Integer> getFullPrimeNumbers()
```

die alle vollständigen Primzahlen in aufsteigender Reihenfolge in einer ArrayList zurückgibt. Schreiben Sie einen möglichst performanten Code. Sie können dabei auf folgende Hilfsfunktion zurückgreifen:

```
public static boolean isPrime(int z) {  
    if (z == 1) {  
        return false;  
    }  
    for (int i = 2; i <= Math.sqrt(z); i++) {  
        if (z % i == 0) {  
            return false;  
        }  
    }  
    return true;  
}
```

**Aufgabe 9 (Version A):**

Gegeben sei ein ungewichteter Graph in Form einer Adjazenzliste. Schreiben Sie eine Funktion

```
public static boolean isReachable(ArrayList<Integer>[] list, int a, int b,  
int maxDist)
```

`isReachable` gibt zurück, ob im Graphen, der durch die Adjazenzliste `list` beschrieben wird, der Knoten `b` vom Knoten `a` aus in maximal `maxDist` Schritten erreicht werden kann. Der Knoten `a` kann von sich selbst aus in 0 Schritten erreicht werden. Fehlerüberprüfungen müssen Sie nicht durchführen.

**Aufgabe 8 (Version B):**

Gegeben sei die Rekursionsgleichung

$$T(n) = \begin{cases} 1 & \text{falls } n = 0 \\ 3 & \text{falls } n = 1 \\ 3 \cdot T(n-1) - 2 \cdot T(n-2) & \text{sonst} \end{cases}$$

Benutzen Sie die Technik der Erzeugenden Funktion um eine geschlossene Form für  $T(n)$  zu finden.

**Hinweis:** Zur geschlossenen Form von F: Die Nullstellen des Nennerpolynoms sind  $1/2$  und 1.

**Aufgabe 9 (Version B):**

Betrachten Sie einfache reguläre Ausdrücke (RA), wie sie in der Vorlesung behandelt wurden. Ihre Aufgabe besteht darin, einen Algorithmus zu entwickeln, der entscheidet, ob zwei Ausdrücke äquivalent sind. Wie gehen Sie vor? Ein Parser, der aus einem RA einen abstrakten Syntaxbaum erstellt, sei gegeben.

**Tipp:** Nutzen Sie Algorithmen, die Sie aus der Automatentheorie kennen.