Lehrstuhl für Informatik I
Effiziente Algorithmen und wissensbasierte Systeme
Universität Würzburg

Würzburg, den 4. Februar 2016

Prof. Dr. Alexander Wolff
M.Sc. Krzysztof Fleszar

Probeklausur: Algorithmen und Datenstrukturen WS 2015/16

Vorname: ______________ Nachname: ______________ Matrikelnummer: __________

Aufgabe	1	2	3	4	5	Zusatzaufg.	**Gesamt**
mögliche Punkte	4	5	8	8	8	4	**33 + 4**
erreichte Punkte							

Aufgabe 1 **[1 + 2 + 1 = 4 Punkte]**

Gegeben sei folgender Algorithmus.

MyAlgorithm(Feld vom Typ int A, int $\ell = 1$, int $r = A.length$)

```
if ℓ < r then
    m = ⌊(ℓ + r)/2⌋
    MyAlgorithm(A, ℓ, m)
    for i = ℓ to r do
        Print(A[i]) // gibt A[i] aus
    MyAlgorithm(A, m + 1, r)
```

a) Sei A ein Feld der Länge n. Stellen Sie eine Rekursionsgleichung für die asymptotische Worst-Case-Laufzeit $T(n)$ von MyAlgorithm(A) auf.

$T(n) =$ ______________________________

b) Lösen Sie die Rekursionsgleichung aus Aufgabenteil a). Erläutern Sie Ihre Lösung.

____

____

____

____

____

$T(n) = \Theta($______________$)$

c) Was gibt MyAlgorithm(A) für $A = \langle 1, 2, 3, 4 \rangle$ aus?

____

____

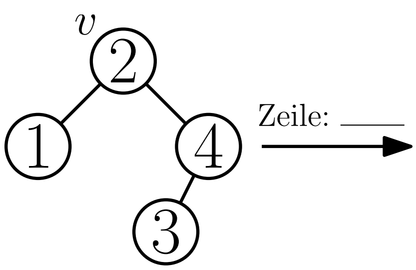
____

Aufgabe 2 **[2 + 2 + 1 = 5 Punkte]**

Gegeben sei ein Knoten v eines binären Suchbaums. Gesucht ist ein Algorithmus, der v durch Ausführung von Rotationen zu einem Blatt macht. Ihnen wird der Algorithmus MakeLeaf vorgeschlagen.

```
MakeLeaf(Node v)
1 while v.left ≠ nil do
2 |   RightRotate(v)
3 if v.right ≠ nil then
4 |   while v.right.left ≠ nil do
5 |   |   RightRotate(v.right)
6 |   LeftRotate(v)
```

a) Wenden Sie MakeLeaf(v) auf den gegebenen Baum an. Stellen Sie dafür das Ergebnis nach jeder Rotation dar. Geben Sie für jede Rotation an, in welcher Zeile von MakeLeaf die Methode RightRotate oder LeftRotate aufgerufen wurde.



b) Ist MakeLeaf korrekt? ☐ ja ☐ nein

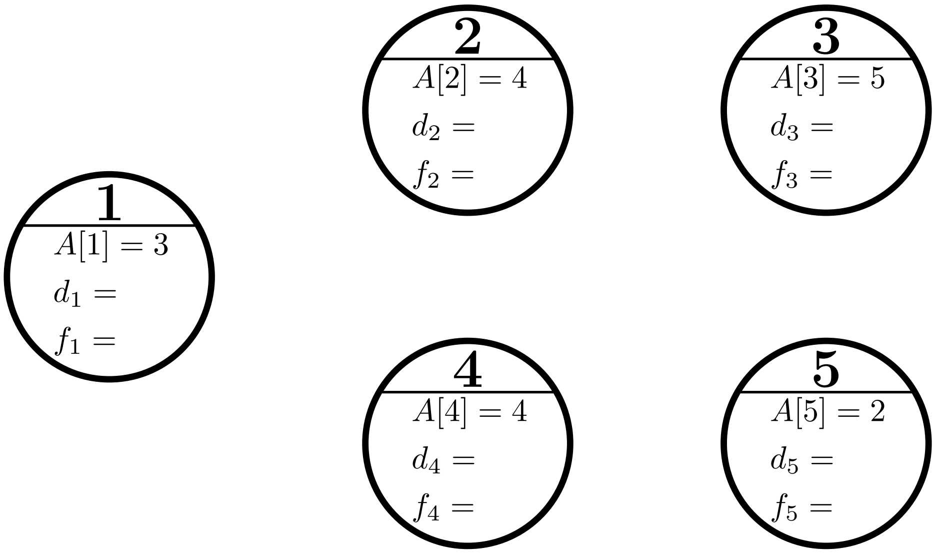
Begründung oder Gegenbeispiel:

c) Wofür werden die hier verwendeten Rotationen normalerweise eingesetzt?

Aufgabe 3 **[1 + 2 + 1 + 1 + 1 + 1 + 1 = 8 Punkte]**

Ein Sortieralgorithmus GraphSort nimmt ein Feld A mit n ganzen Zahlen entgegen und erzeugt zunächst einen gerichteten Graphen $G = (V, E)$ mit $V = \{1, 2, \ldots, n\}$. Zwischen zwei Knoten $i, j \in V$ wird genau dann eine gerichtete Kante (i, j) eingefügt, wenn $A[i] < A[j]$. Nach der Initialisierung des Graphen wird der Algorithmus TopologicalSort(G) aufgerufen, der eine Liste $L = (j_1, j_2, \ldots, j_n)$ liefert. Der Algorithmus GraphSort gibt schließlich das Feld $\langle A[j_1], A[j_2], \ldots, A[j_n]\rangle$ zurück.

a) Ergänzen Sie die Zeichnung um gerichtete Kanten, so dass sie den Graphen G für das Feld $A = \langle 3, 4, 5, 4, 2\rangle$ darstellt.



b) Führen Sie eine Tiefensuche auf G ausgehend vom Knoten 1 durch. Tragen Sie für jeden Knoten i die Entdeckungszeit d_i und die Abschlusszeit f_i in Ihre Zeichnung aus Aufgabenteil a) ein.

c) Wie hängt das Ergebnis der Tiefensuche auf G mit dem Ergebnis von TopologicalSort(G) zusammen?

d) Die asymptotische Worst-Case-Laufzeit von GraphSort in Abhängigkeit von n ist $\Theta\Big(\underline{\qquad\qquad\qquad}\Big)$.

e) Arbeitet GraphSort in situ? ☐ ja ☐ nein

Begründung:

f) Was bedeutet es, wenn ein Sortieralgorithmus stabil ist?

g) Beschreiben Sie, mit welcher einfachen Änderung GraphSort stabil wird.

Aufgabe 4 **[4 + 2 + 2 = 8 Punkte]**

Gegeben sei ein Feld A, das eine gerade Anzahl n von ganzen Zahlen enthält. Jede Zahl in A kommt genau zweimal vor. Es gibt also $n/2$ verschiedene Zahlen in A.
Gesucht ist ein Algorithmus, der den größten Abstand $d_{\max}$ bestimmt, den zwei gleiche Zahlen in A voneinander haben.

a) Entwerfen Sie den gesuchten Algorithmus in Pseudocode. Sie erhalten zwei Punkte, wenn Ihr Algorithmus korrekt ist, und zwei weitere Punkte, wenn er Laufzeit $O(nd_{\max})$ hat.

MaxDistance(Feld vom Typ int A)

b) Wie kann man das Problem in $O(n \log n)$ Zeit lösen? Skizzieren Sie Ihre Idee in Worten.

c) Sei $d_{\min}$ der *kleinste* Abstand zwischen zwei gleichen Zahlen in A. Wie groß kann $d_{\min}$ höchstens sein? Zeigen Sie, wie der von Ihnen genannte Wert für eine Instanz der Größe n auftreten kann, und begründen Sie, warum es keine Instanz der Größe n gibt, bei der $d_{\min}$ größer ist.

Aufgabe 5 **[4 + 1 + 1 + 1 + 1 = 8 Punkte]**

Gegeben sei ein Feld A mit n verschiedenen ganzen Zahlen und eine ganze Zahl $1 \leq k \leq n$. Gesucht ist ein Algorithmus, der ein Feld B zurückgibt, das die k kleinsten Zahlen aus A enthält. Die folgenden vier Algorithmen lösen dieses Problem.

```
KSmallest1(Feld vom Typ int A, int k)
    MergeSort(A)
    Sei B[1..k] ein neues Feld vom Typ int.
    for i = 1 to k do
        B[i] = A[i]
    return B
```

```
KSmallest2(Feld vom Typ int A, int k)
    BuildMinHeap(A)
    Fasse A als Prioritätsschlange auf.
    Sei B[1..k] ein neues Feld vom Typ int.
    for i = 1 to k do
        B[i] = A.ExtractMin() // funktioniert analog zu ExtractMax aus Vorlesung
    return B
```

```
KSmallest3(Feld vom Typ int A, int k)
    p = Select(A, 1, A.length, k)
    Sei B[1..k] ein neues Feld vom Typ int.
    j = 1
    for i = 1 to A.length do
        if A[i] ≤ p then
            B[j] = A[i]
            j = j + 1
    return B
```

```
KSmallest4(Feld vom Typ int A, int k)
    Sei T ein neuer Rot-Schwarz-Baum.
    for i = 1 to k do
        T.Insert(A[i])
    for i = k + 1 to n do
        if A[i] < T.Maximum().key then
            T.Delete(T.Maximum())
            T.Insert(A[i])
    Sei B[1..k] ein neues Feld vom Typ int.
    ptr = T.Minimum()
    for i = 1 to k do
        B[i] = ptr.key
        ptr = ptr.Successor()
    return B
```

a) Geben Sie für jeden der vier Algorithmen die asymptotische Worst-Case-Laufzeit in Abhängigkeit von n und k an.

	KSmallest1	KSmallest2	KSmallest3	KSmallest4
Laufzeit	$\Theta\Big($ ________ $\Big)$	$\Theta\Big($ ________ $\Big)$	$\Theta\Big($ ________ $\Big)$	$\Theta\Big($ ________ $\Big)$

b) Welche der vier Algorithmen geben das Feld B aufsteigend geordnet zurück?

	KSmallest1	KSmallest2	KSmallest3	KSmallest4
liefert B aufsteigend geordnet	□	□	□	□

c) Welche der vier Algorithmen lassen die Ordnung der Zahlen in Feld A unverändert?

	KSmallest1	KSmallest2	KSmallest3	KSmallest4
lässt A unverändert	□	□	□	□

d) Was berechnet der Algorithmus Select in KSmallest3?

____

____

e) Geben Sie in Abhängigkeit von A, k und dem Laufindex i an, welche Zahlen der Rot-Schwarz-Baum T nach Ausführung des if-Blocks in KSmallest4 enthält.

____

____

____

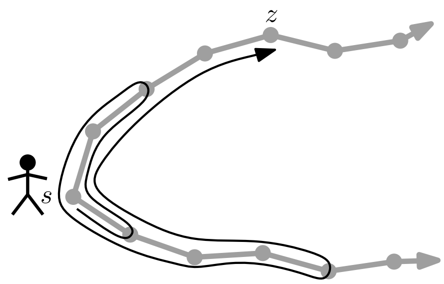
Zusatzaufgabe **[4 Zusatzpunkte]**

Ein Wanderer gelangt an eine Weggabelung s und weiß nicht, welcher von zwei Wegen A und B zu seinem Ziel z führt. Ob er den richtigen Weg gewählt hat, merkt er erst, wenn er das Ziel erreicht. Die Distanz von s nach z (entlang des richtigen Weges) sei $d \in \mathbb{Z}^+$. Der Wanderer kennt diese Distanz jedoch nicht.
Um das Ziel zu erreichen, exploriert der Wanderer abwechselnd die Wege A und B. Für jede Exploration legt er den Suchradius r fest, geht von s entlang des jeweiligen Weges, bis er Distanz r zurückgelegt hat, und kehrt nach s zurück.
Für die erste Exploration wählt der Wanderer $r = 1$. Nach jeder erfolglosen Exploration eines Weges vergrößert er den Suchradius, wobei er eine der folgenden Zuweisungen anwendet:

(1) $r = r + 1$

(2) $r = 2r$

Den neuen Suchradius wendet er nun für die Exploration des anderen Weges an.



Sei D_1 die Distanz, die der Wanderer mit Zuweisung (1) höchstens zurücklegt, bis er z erreicht.
Sei D_2 die Distanz, die der Wanderer mit Zuweisung (2) höchstens zurücklegt, bis er z erreicht.
Geben Sie D_1 und D_2 in Abhängigkeit von d genau an. Sie können davon ausgehen, dass d eine Zweierpotenz ist.

$D_1 =$

$D_2 =$

Geben Sie die Größenordnung von D_1 und D_2 in Abhängigkeit von d an.

D_1 ist in $\Theta\Big(\underline{\hspace{3cm}}\Big)$. D_2 ist in $\Theta\Big(\underline{\hspace{3cm}}\Big)$.