

Nachname (Druckschrift): _____

Vorname (Druckschrift): _____

Matrikelnummer: _____

Studiengang: _____

Bitte Hinweise beachten:

- Schreiben Sie Ihren Namen **nur auf dieses Titelblatt**.
- Merken oder notieren Sie sich Ihre Klausurnummer **0000**.
- Zusatzpapier abgeben? Vermerken Sie darauf unbedingt die Klausurnummer **0000**. Außerdem: Verweisen Sie unbedingt von der Aufgabe auf das Zusatzblatt!
- Nur **dokumentenechte Stifte** in den Farben blau und schwarz! Füller, Tipp-Ex, Tintenlöscher sind untersagt.
- Zugelassenes Hilfsmittel: 1 DIN A4 Blatt mit handschriftlichen Notizen (beidseitig).
- Alle elektronischen Geräte sind untersagt! Insbesondere also Handys, Smartwatches, Taschenrechner. **Ausschalten und wegpacken!** Nichtbeachtung stellt einen Täuschungsversuch dar und führt zum Nichtbestehen der Klausur.
- Zwei oder mehr Lösungen angegeben? Die Aufgabe gilt dann als nicht gelöst. Entscheiden Sie sich also immer für **eine** Lösung.
- Begründungen sind nur dann notwendig, wenn die Aufgabenformulierung dies verlangt.
- Die Klausur ist mit Sicherheit bestanden, wenn mindestens **50%** der Höchstpunktzahl erreicht wird. Die Klausur dauert 180 Minuten.



ALGORITHMEN UND DATENSTRUKTUREN 1

SoSe 2021 · Prof. Dr. H. Dell

6. Oktober 2021

Klausurnummer: **0000**



**Diese Seite ist für den internen Gebrauch.
Bitte leer lassen.**



ALGO1-21-0000

Seite 2 von 14

**Diese Seite ist für den internen Gebrauch.
Bitte leer lassen.**

Aufgabe	1	2	3	4	5	6	7
Erreichbar	16	10	12	10	12	9	11
Erreicht							

Klausur	Bonifikation	Summe	Note



a) Sei n eine natürliche Zahl. Wie viele Sterne (*) gibt der folgende Code aus?

```

for  $i = 1, 2, \dots, 2n$  do
  if  $i$  ist gerade then
    print "**"
  else
    print "***"
    
```

Es werden genau _____ Sterne ausgegeben. (Gesucht ist die genaue Anzahl in Abhängigkeit von n , nicht die asymptotische Anzahl in O- oder Θ -Notation.)

↑↑↑ _____ / 2 Punkte ↑↑↑

b) Geben Sie für die folgenden Algorithmen jeweils die Laufzeit in Θ -Notation abhängig von n an.

```

for  $i = 1, 2, \dots, n$  do
  if  $i == n$  then
    for  $j = 1, 2, \dots, i$ 
    do
      print "*"
    
```

```

 $s = 1$ 
while  $s \leq n$  do
   $s = 5s$ 
  for  $i = 1, 2, \dots, s$  do
    print "*"
  
```

```

 $s = 1$ 
while  $s \leq 5^{\sqrt{n}}$  do
   $s = 3s$ 
  
```

$\Theta(\text{_____})$ $\Theta(\text{_____})$ $\Theta(\text{_____})$

↑↑↑ _____ / 6 Punkte ↑↑↑

c) Sei $f(n) = 100n \log n + n^2 \log n + \frac{1}{2}n + n^3 / \log \log n$. Welches asymptotische Wachstum ist für diese Funktion richtig? Kreuzen Sie genau eine Box an.

- $\Theta(n^2 \log n)$ $\Theta(n^3 / \log \log n)$ $\Theta(n \log n)$ $\Theta(\sqrt{n})$ $\Theta(n)$

↑↑↑ _____ / 2 Punkte ↑↑↑

d) Sei $f(n) = \sqrt{n} \log n + n^{2/3} - 99\sqrt{n} - \log \log n$. Welches asymptotische Wachstum ist für diese Funktion richtig? Kreuzen Sie genau eine Box an.

- $\Theta(n^{2/3} - \sqrt{n} \log^{100} n)$ $\Theta(\sqrt{n})$ $\Theta(\log \log n)$ $o(n^{2/3})$ $\Theta(n^2)$

↑↑↑ _____ / 2 Punkte ↑↑↑

e) Sei $f(n) = (n^3 + n^{0.1}) \cdot (\sqrt{n} + \log n)$. Welches asymptotische Wachstum ist für diese Funktion richtig? Kreuzen Sie genau eine Box an.

- $O(n^{3.1})$ $o(n^{3.1})$ $\Theta(n^{3.1})$ $\Omega(n^9)$ $\omega(n^{3.1})$

↑↑↑ _____ / 2 Punkte ↑↑↑

f) Sei $f(n) = \frac{\log^{99} n}{n^{0.1}}$. Welches asymptotische Wachstum ist für diese Funktion richtig? Kreuzen Sie genau eine Box an.

- $\Omega(\log^{99} n)$ $O(1)$ $\omega(\log^{99} n)$ $\Theta(\log^{99} n)$ $\Omega(1)$

↑↑↑ _____ / 2 Punkte ↑↑↑



a) Betrachten Sie die folgende Funktion FOO.

```
function FOO(n)
  if n ≤ 0 then
    print "**"
  else
    FOO(n - 1)
    print "*"
    FOO(n - 1)
    print "**"
```

Zum Beispiel gibt FOO(0) zwei Sterne (*) aus. Wie viele Sterne gibt FOO(1) aus?

Antwort: _____.

Geben Sie außerdem eine Rekursionsgleichung für die genaue Anzahl $A(n)$ von Sternen an, die ein Aufruf von FOO(n) insgesamt ausgibt. Der Basisfall ist $A(0) = 2$.

Für alle $n \geq 1$ gilt: $A(n) =$ _____

↑↑↑ _____ / 4 Punkte ↑↑↑

b) Geben Sie für folgende Rekursionsgleichungen eine geschlossene Form an.

Sie können bei B und C davon ausgehen, dass $n = 2^k$ für eine natürliche Zahl $k > 0$ gilt, und bei D , dass $n = 2k$ für eine natürliche Zahl $k > 0$ gilt. Geben Sie das Ergebnis in Θ -Notation abhängig von n an.

- $B(n) = B(\frac{n}{2}) + n^2, \quad B(1) = 1.$

$$B(n) = \Theta(\text{_____})$$

- $C(n) = 8 \cdot C(\frac{n}{2}) + n^2, \quad C(1) = 1.$

$$C(n) = \Theta(\text{_____})$$

- $D(n) = D(n - 2) + n^3, \quad D(0) = 1.$

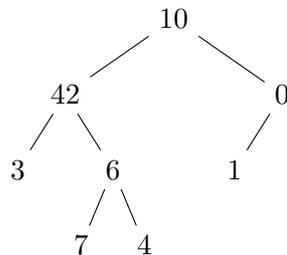
$$D(n) = \Theta(\text{_____})$$

↑↑↑ _____ / 6 Punkte ↑↑↑



Sei T ein Binärbaum. Jeder Knoten x von T hat die Eigenschaften $x.parent$, $x.left$ und $x.right$, welche auf den Elternknoten sowie auf das linke und rechte Kind von x verweisen. Wenn der Knoten kein Kind oder keinen Elternknoten hat, ist der jeweilige Wert auf `null` gesetzt (zum Beispiel $x.right = null$). Des Weiteren hat jeder Knoten x eine Eigenschaft $x.number$, die eine einzelne Zahl speichert.

In diesem Beispielbaum B ist jeder Knoten mit $x.number$ markiert:



Die Wurzel von B ist der Knoten v mit $v.number = 10$.

a) Betrachten Sie die folgende Funktion:

```

function FUNCA( $x$ )
  if  $x \neq \text{null}$  then
    return FUNCA( $x.left$ ) + FUNCA( $x.right$ )
  else
    return 1
  
```

Welche Zahl liefert $\text{FUNCA}(v)$ über die **return** Anweisung als Rückgabewert, wenn v die Wurzel des Beispielbaums B ist?

Antwort: _____

↑↑↑ _____ / 2 Punkte ↑↑↑

b) Betrachten Sie die folgende Funktion:

```

function FUNCB( $x, i$ )
  if  $x \neq \text{null}$  und  $i == 0$  then
    print  $x.number$ 
    FUNCB( $x.left, 1$ )
    FUNCB( $x.right, 1$ )
  else if  $x \neq \text{null}$  und  $i == 1$  then
    FUNCB( $x.left, 0$ )
    print  $x.number$ 
  
```

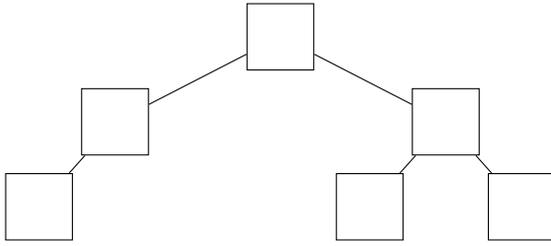
Welche Zahlen gibt $\text{FUNCB}(v, 1)$ über die **print** Anweisungen aus, wenn v die Wurzel des Beispielbaums B ist? Geben Sie die Zahlen in derselben Reihenfolge an, in der sie ausgegeben werden.

Antwort: _____

↑↑↑ _____ / 3 Punkte ↑↑↑



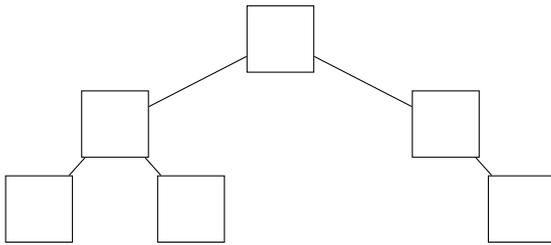
c) Hier ist die Struktur eines Baumes, die Zahlen $x.number$ sind nicht dargestellt:



Die Präorder-Traversierung (*preorder traversal*) dieses Baumes gibt die Zahlen 19, 16, 12, 51, 17, 11 in dieser Reihenfolge aus. Füllen Sie die Zahlen richtig in die Knoten ein.

↑↑↑ _____ / 2 Punkte ↑↑↑

d) Hier ist die Struktur eines Baumes, die Zahlen $x.number$ sind nicht dargestellt:



Die Inorder-Traversierung (*inorder traversal*) dieses Baumes gibt die Zahlen 19, 16, 12, 51, 17, 11 in dieser Reihenfolge aus. Füllen Sie die Zahlen richtig in die Knoten ein.

↑↑↑ _____ / 2 Punkte ↑↑↑

e) Wir erinnern uns an die Funktion `SIZE` aus der Vorlesung, die bei Aufruf von `SIZE(x)` die Zahl aller Knoten im Unterbaum mit Wurzel x zurückliefert.

Wir wollen jetzt eine Funktion `BLUBB` schreiben, die die Anzahl aller Knoten zurückliefert, die einen Vorfahren y mit $y.number == 42$ haben. Zum Beispiel: Wenn v die Wurzel des Beispielbaums B ist, dann soll `BLUBB(v)` die Zahl 5 zurückliefern (der Knoten 42 zählt also mit).

Vervollständigen Sie in der folgenden Funktion die drei **return**-Anweisungen. Sie dürfen die Funktion `SIZE` benutzen, aber keine weiteren Zeilen einfügen!

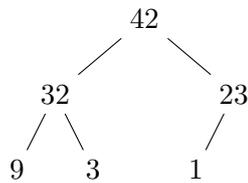
```

function BLUBB(x)
  if x == null then
    return _____
  else if x.number == 42 then
    return _____
  else
    return _____
  
```

↑↑↑ _____ / 3 Punkte ↑↑↑



- a) Der **Max-Heap** ist eine konkrete Datenstruktur, die eine Prioritätswarteschlange implementiert. Hier ist ein **Max-Heap** mit sechs Elementen abgebildet:



Endergebnis hier zeichnen:

Wir rufen jetzt `EXTRACTMAX()` und `INSERT(2)` in dieser Reihenfolge auf. Zeichnen Sie im selben Stil den Max-Heap, der dadurch am Ende entsteht.

↑↑↑ _____ / 2 Punkte ↑↑↑

- b) Hier ist ein Max-Heap als Feld dargestellt:

0	1	2	3	4	5	6	7	8
–	99	42	7	23	9	6	1	0

Zeichnen Sie den Max-Heap als Baum im selben Stil wie in Aufgabenteil a).

Endergebnis hier zeichnen:

↑↑↑ _____ / 2 Punkte ↑↑↑

- c) **Hashing mit linearem Sondieren** ist eine konkrete Datenstruktur, die eine dynamische Menge von Elementen verwaltet. Als Hashfunktion benutzen wir die Funktion $h: \mathbb{N} \rightarrow \{0, 1, \dots, 12\}$ mit

$$h(x) = (2x) \bmod 13.$$

Die Hash-Tabelle hat Platz für dreizehn Einträge und ist derzeit wie folgt gefüllt:

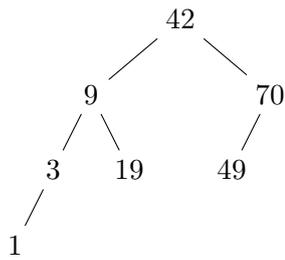
0	1	2	3	4	5	6	7	8	9	10	11	12
	33					42		30				

Fügen Sie nacheinander 4, 7, 6, 19 in dieser Reihenfolge in die Hashtabelle ein.

↑↑↑ _____ / 2 Punkte ↑↑↑



- d) Ein **binärer Suchbaum** ist eine konkrete Datenstruktur, die eine Menge von sortierten Daten verwaltet. Hier ist ein binärer Suchbaum mit sieben Elementen abgebildet:

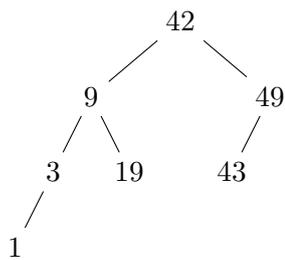


Endergebnis hier zeichnen:

Wir rufen jetzt DELETE(9) und INSERT(43) in dieser Reihenfolge auf. Zeichnen Sie im selben Stil den binären Suchbaum, der dadurch am Ende entsteht.

↑↑↑ _____ / 2 Punkte ↑↑↑

- e) Ein **AVL-Baum** ist eine konkrete Datenstruktur, die eine Menge von sortierten Daten verwaltet. Hier ist ein AVL-Baum mit sieben Elementen abgebildet:



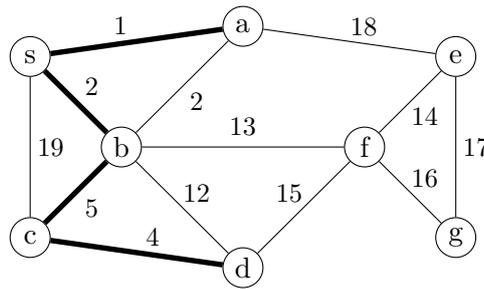
Endergebnis hier zeichnen:

Wir rufen jetzt DELETE(9) und INSERT(44) in dieser Reihenfolge auf. Zeichnen Sie im selben Stil den AVL-Baum, der dadurch am Ende entsteht.

↑↑↑ _____ / 2 Punkte ↑↑↑



Wir lassen Prim's, Kruskal's und Dijkstra's Algorithmus auf diesem ungerichteten, gewichteten Graphen laufen:



- a) Prim's Algorithmus wurde mit Wurzel s gestartet. Die Abbildung zeigt den Zustand von Prim's Algorithmus, nachdem vier Kanten in den Baum eingefügt wurden (fett gezeichnet). Welche Kante $\{u, v\}$ wird Prim's Algorithmus als Nächstes einfügen?

Antwort: _____ (gesucht ist die Kante, nicht ihr Gewicht)

↑ ↑ ↑ _____ / 2 Punkte ↑ ↑ ↑

- b) Dieselbe Abbildung zeigt den Zustand von Kruskal's Algorithmus, nachdem vier Kanten in den Wald eingefügt wurden (fett gezeichnet). Welche Kante $\{u, v\}$ wird Kruskal's Algorithmus als Nächstes einfügen?

Antwort: _____ (gesucht ist die Kante, nicht ihr Gewicht)

↑ ↑ ↑ _____ / 2 Punkte ↑ ↑ ↑

- c) Dijkstra's Algorithmus wurde mit Wurzel s gestartet. Dieselbe Abbildung zeigt jetzt den Zustand von Dijkstra's Algorithmus, nachdem vier Kanten in den Baum eingefügt wurden (fett gezeichnet). Welche Kante $\{u, v\}$ wird Dijkstra's Algorithmus als Nächstes einfügen?

Antwort: _____ (gesucht ist die Kante, nicht ihr Gewicht)

↑ ↑ ↑ _____ / 2 Punkte ↑ ↑ ↑

- d) Welches Gewicht hat der minimale Spannbaum?

Antwort: _____

↑ ↑ ↑ _____ / 2 Punkte ↑ ↑ ↑

- e) Gibt es einen minimalen Spannbaum, der die Kante $\{a, e\}$ enthält? Schreiben Sie "Ja" oder "Nein" und begründen Sie Ihre Antwort.

↑ ↑ ↑ _____ / 2 Punkte ↑ ↑ ↑

- f) Gibt es einen minimalen Spannbaum, der die Kante $\{b, c\}$ nicht enthält? Schreiben Sie "Ja" oder "Nein" und begründen Sie Ihre Antwort.

↑ ↑ ↑ _____ / 2 Punkte ↑ ↑ ↑



- a) Sei $A[1..n]$ ein Feld der Länge $n \geq 3$, welches rationale Zahlen enthält. Und seien i, j ganze Zahlen mit $1 \leq i \leq j \leq n$. Hier ist ein rekursiver Algorithmus:

```

function RECFUNC( $A, i, j$ )
    if  $i == n$  then
        return  $A[j]$ 
    else if  $j == i$  then
        return  $A[i]$ 
    else
        return  $\max \{ A[i] \cdot \text{RECFUNC}(A, i + 1, j), (A[j])^2 \cdot \text{RECFUNC}(A, i, j - 1) \}$ 

```

Benutzen Sie dynamische Programmierung, um diesen rekursiven Algorithmus iterativ zu machen. Füllen Sie hierzu die folgende Funktion aus. Sie dürfen nur die unterstrichenen Teile vervollständigen und keine neuen Zeilen hinzufügen!

```

function ITERFUNC( $A$ )
    Initialisiere ein zweidimensionales Feld  $T[1..n][1..n]$ .
    for  $i$  from _____ to _____ do
        for  $j$  from _____ to _____ do
            if _____ then
                _____
            else if _____ then
                _____
            else
                _____
    return  $T[1][n]$ 

```

↑↑↑ _____ / 4 Punkte ↑↑↑

- b) Sei $S[1..n]$ eine Zeichenkette (*string*) der Länge n . Jedes Zeichen ist entweder **a** oder **b**. Geben Sie einen **rekursiven** Algorithmus COUNT an, der **True** oder **False** zurückgibt. Hierbei soll $\text{COUNT}(i, j, k)$ für $1 \leq i \leq j \leq n$ genau dann **True** zurückliefern, wenn der Teilstring $S[i..j]$ genau k mal das Zeichen **a** enthält. Abgesehen von rekursiven Aufrufen muss die Laufzeit **konstant** sein. Geben Sie Ihren Algorithmus in **Pseudocode** an:

```

function COUNT( $i, j, k$ )

```

↑↑↑ _____ / 5 Punkte ↑↑↑



Um wieder in Form zu kommen, haben Sie sich entschieden, im nun anstehenden Präsenzsemester jeden Tag von Zuhause zur Uni zu joggen. Es gibt viele Hügel in der Stadt. Sie wollen eine *ausgeglichene Joggingroute* finden, die erst anstrengend und dann lockerer ist, damit Sie gegen Ende abkühlen können. Das heißt, Ihre Joggingroute darf zuerst nur bergauf führen und dann nur bergab.

Sie haben eine detaillierte Straßenkarte mit insgesamt m **Straßenabschnitten** und n **Kreuzungen**. Jeder Straßenabschnitt verbindet zwei Kreuzungen, hat eine **positive, bekannte Länge** und hat eine feste Steigung. Jede Kreuzung hat eine **bekannte Höhe**. Ihre Joggingroute startet an Kreuzung s (Zuhause) und endet an Kreuzung t (Uni), wobei $s \neq t$. Eine Joggingroute ist **ausgeglichen**, wenn es für die Sequenz der Höhen h_1, h_2, \dots, h_ℓ entlang der besuchten Kreuzungen einen Index i gibt mit

$$h_1 \leq h_2 \leq \dots \leq h_{i-1} \leq h_i \geq h_{i+1} \geq h_{i+2} \geq \dots \geq h_\ell.$$

- a) Beschreiben Sie **kurz, präzise und in Stichpunkten** einen Algorithmus, der die kürzeste ausgeglichene Laufstrecke berechnet. Der Algorithmus soll die Antwort in Zeit $O(m + n \log n)$ berechnen; begründen Sie, warum Ihr Algorithmus die Zeit einhält.

↑↑↑ _____ / 5 Punkte ↑↑↑

- b) Wir nehmen an, dass alle Kreuzungen unterschiedliche Höhen haben. Beschreiben Sie **kurz, präzise und in Stichpunkten** einen Algorithmus, der die kürzeste ausgeglichene Laufstrecke berechnet. Der Algorithmus soll die Antwort in Zeit $O(m + n)$ berechnen; begründen Sie, warum Ihr Algorithmus die Zeit einhält.

↑↑↑ _____ / 6 Punkte ↑↑↑



Wichtig: Lösungen auf dieser Seite werden nur dann berücksichtigt, wenn bei der entsprechenden Aufgabe ein Verweis zu Seite 13 platziert wurde.



Wichtig: Lösungen auf dieser Seite werden nur dann berücksichtigt, wenn bei der entsprechenden Aufgabe ein Verweis zu Seite 14 platziert wurde.

