Universität Konstanz
Fachbereich Informatik & Informationswissenschaft
Dr. Barbara Pampel

Probeklausur
Konzepte der Informatik

19. Dezember 2023

Matrikelnummer.: ______________________ **Sitzplatz:** ______________________

PIC:

Hinweise: Es sind keine Hilfsmittel erlaubt. Schreiben Sie nicht in grüner oder roter Farbe und nicht mit Bleistift. Begründen Sie Ihre Aussagen und machen Sie deutlich, wenn Sie Sätze, Hilfssätze, Algorithmen oder Datenstrukturen aus der Vorlesung verwenden. Sie schreiben diese Klausur unter dem Vorbehalt, dass Sie zugelassen sind. Wenn Sie das Ergebnis dieser Klausur per Aushang erfahren wollen, merken Sie sich bitte Ihren persönlichen Identifizierungs-Code (PIC). Wir wünschen Ihnen viel Erfolg!

Hörsaal verlassen: __________ bis __________ Uhr, __________ bis __________ Uhr

Vorzeitige Abgabe: __________ Uhr

Aufgabe	**1**	**2**	**3**	**4**	**gesamt**
mögliche Punkte	10	10	10	10	40
erreichte Punkte					

Aufgabe 1: ***Zahlen*** 10 Punkte

(a) Rechnen Sie die Binärzahl $1011,11_2$ in eine Dezimalzahl um.

$1011,11_2 = 8 + 2 + 1 + 0,5 + 0,25 = 11,75_{10}$

Ein Punkt für korrekte Vor- und ein Punkt für korrekte Nachkommastellen.

(b) Rechnen Sie die Dezimalzahl $19,75_{10}$ in eine Binärzahl um.

Vorkommastellen: $n = 19, x_0 = 1; n = 9, x_1 = 1; n = 4, x_2 = 0;$
$n = 2, x_3 = 0; n = 1, x_4 = 1, n = 0$
Nachkommastellen: $n = 0,75, x_{-1} = 1; n = 0,5, x_{-2} = 1; n = 0$
Ergebnis: $19,75_{10} = 10011,11_2$

Ein Punkt für korrekte Vor- und ein Punkt für korrekte Nachkommastellen.

(c) Rechnen Sie die Binärzahl 101010111110100011_2 in eine Oktalzahl um.

253643_8
Ein Punkt für das richtige Ergebnis, bei kleinen Fehlern nur ein halber Punkt Abzug.

(d) Geben Sie die Zahl -13_{10} in (binär) 8-Bit Zweierkomplement-Darstellung an.

$13_{10} = 00001101_2$ => Invertieren: 11110010 => plus 1: 11110011
(2 Punkte, ein Punkt Abzug, falls nicht 8 Bit lang)

(e) Berechnen Sie die IEEE 754-Darstellung (32-bit Genauigkeit) zu folgender Dezimalzahl:

$$-11,25_{10}$$

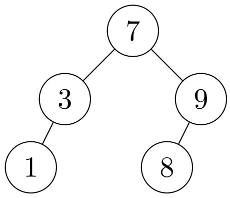
negative Zahl: Vorzeichenbit ist 1
$-11,25_{10} = 1011,01_2 = 1,01101_2 \cdot 2^3$
$E = 3 + 127 = 130_{10} = 10000010_2$

1 10000010 01101000000000000000000

Ein halber Punkt für korrektes Vorzeichen, ein Punkt für den korrekten Exponenten und 1,5 für die Mantisse.

Aufgabe 2: *Binäre Suchbäume* 10 Punkte

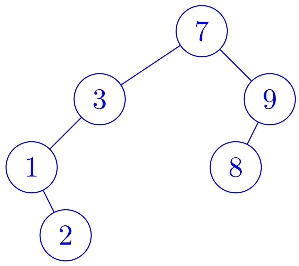
(a) Führen Sie die folgenden Operationen nacheinander für den unten stehenden, bzw. für den nach der vorherigen Operation entstandenen binären Suchbaum durch.



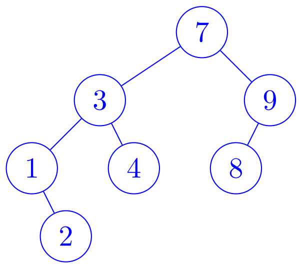
Zeichnen Sie den Baum nach jeder Operation.

1 Punkt für jede korrekte insert-Operation, 2 für jede korrekte remove-Operation

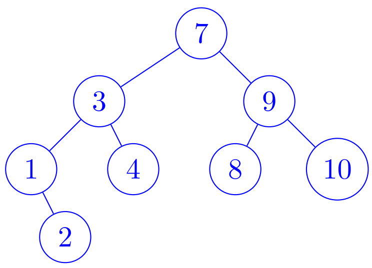
i. insert(2)



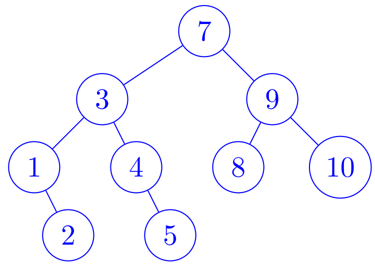
ii. insert(4)



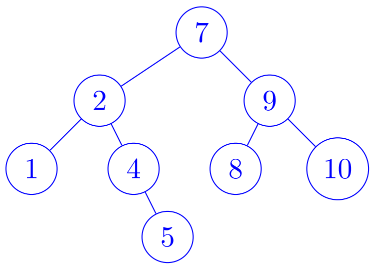
iii. insert(10)

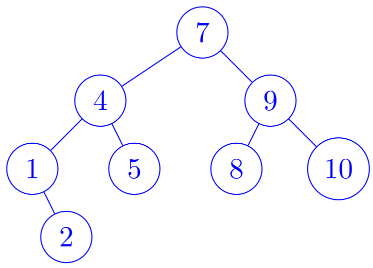


iv. insert(5)

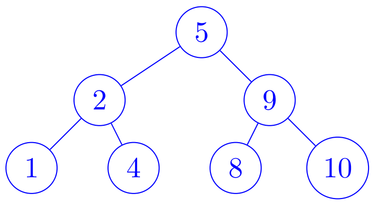


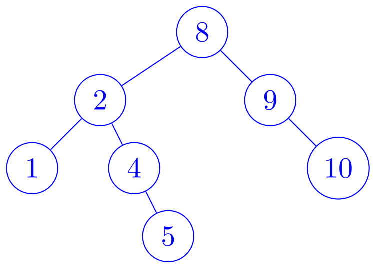
v. remove(3) zwei Varianten



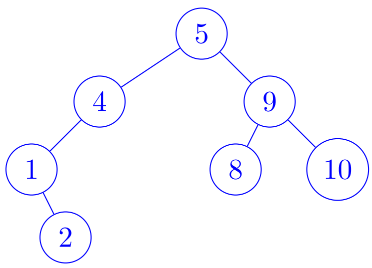


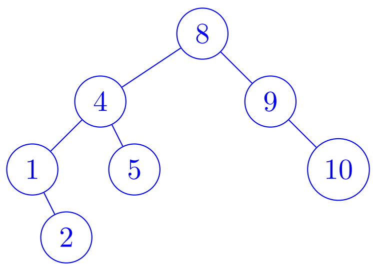
vi. remove(7) für jede der oberen Varianten zwei Varianten
für die linke:





für die rechte:





(b) Wie viele Knoten muss man in einem binären Suchbaum mit n Knoten maximal besuchen, um einen gegebenen Schlüssel zu finden bzw. zu wissen, dass er nicht im Baum enthalten ist. Begründen Sie!

Für zwei Punkte: Bis zu n Knoten.
Begründung: Ein binärer Suchbaum kann entarten, da keinerlei Balancierungs-Operationen vorgenommen werden. Ein Knoten wird IMMER als Blatt eingefügt.

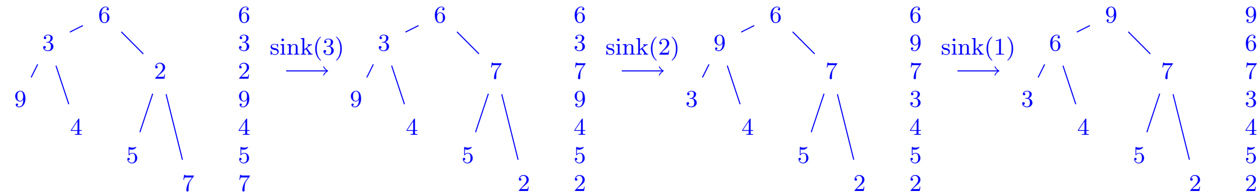
Aufgabe 3: ***Sortieren*** 10 Punkte

Gegeben sei folgendes Array M:

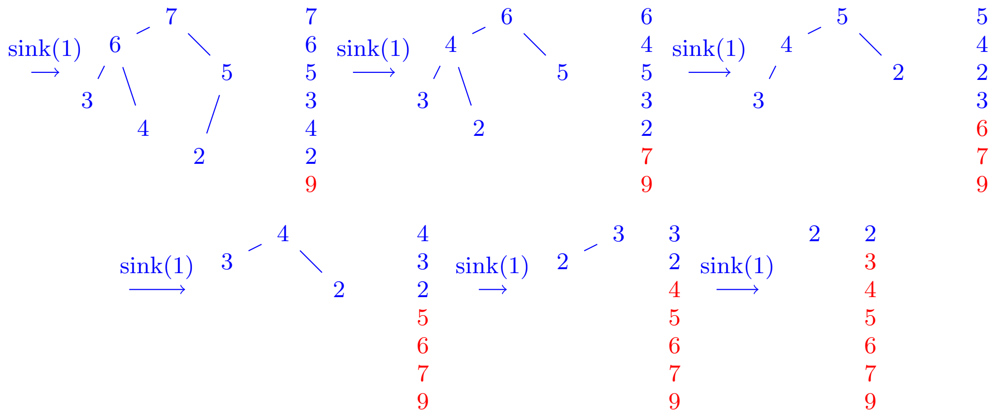
6	3	2	9	4	5	7

Sortieren Sie M nicht-absteigend mit HeapSort. Zeichnen Sie Baum **und** Array de nach **jedem** Aufruf von `sink`.

Heapaufbau (zwingend in dieser Reihenfolge!)



Heapabbau



Ein Punkt für jeden korrekten Schritt.

Aufgabe 4: *Analyse* 10 Punkte

Gegeben folgender Algorithmus:

Algorithm 1: Sort

```
Input: Array A[1, n]
begin
    for i = n − 1, ..., 1 do
        j ← i;
        while j < n and A[j] ≥ A[j + 1] do
            vertausche A[j] und A[j + 1];
            j ← j + 1;
        print(A);
```

(a) Wenden Sie den Algorithmus auf folgende Eingabe an:

5	6	5	2	3

Dokumentieren Sie den Verlauf, indem Sie A **nur** bei jedem Aufruf des `print`-Befehls ausgeben.

Für zwei Punkte:

5	6	5	2	3
5	6	2	3	5
5	2	3	5	6
2	3	5	5	6

(b) Arbeitet der Algorithmus stabil? Begründen Sie!

Nein, das tut er nicht, denn bei Vergleich mit dem rechten Nachbarn, wird ein Element auch getauscht, wenn es gleich groß ist!
(Ein Punkt.)

(c) Geben Sie seine (worst-case) Laufzeit in $\mathcal{O}$-Notation an. Begründen Sie!

Die Laufzeit setzt sich als Summe der Durchläufe der inneren Schleifen zusammen. Diese wird (im worst case) immer länger von einem bis maximal $n-1$ Vergleichen.

$$\sum_{k=1}^{n-1} k = \frac{(n-1)\cdot n}{2} = \frac{1}{2}(n^2 - n) \in \mathcal{O}(n^2)$$

Je ein Punkt für Begründung, Formel und O-Notation. Noch zwei Punkte, wenn einfach nur Zahl der Schleifendurchläufe multipliziert wurde.

(d) Beweisen Sie die Korrektheit des Algorithmus (nach Floyd).

i. **Schleifeninvariante (INV):**
Nach jedem Schleifendurchlauf (DL) für i gilt: $A[i, n]$ ist nicht-absteigend sortiert.
Induktionsanfang (IA): $i = n - 1$: Im ersten Schleifendurchlaufs werden $A[n-1]$ und $A[n]$ verglichen. $A[n-1]$ wird vor $A[n]$ behalten, falls es echt kleiner ist und danach einsortiert, falls es grösser oder gleich ist. Damit ist $A[n-1, n]$ nicht-absteigend sortiert.
Induktionsschritt (IS): $i + 1 \to i$: $A[i + 1, n]$ ist nicht-absteigend sortiert. Das Element $A[i]$ wird vor dem ersten Element einsortiert, welches echt grösser ist als $A[i]$. Damit bleibt $A[i, n]$ auch nach dem Einfügen und damit nach dem Schleifendurchlauf nicht-absteigend sortiert.

ii. vor dem ersten Schleifendurchlauf ist $A[n]$ sortiert, da einelementig.

iii. nach dem letzten Schleifendurchlauf ist nach INV $A[1, n]$ und damit das ganze Array sortiert.

iv. Die while-Schleife bricht spätestens ab, wenn $j = n$, die äussere for-Schleife nach $n - 1$ Durchläufen. Damit terminiert der Algorithmus.

Ein Punkt für INV, je ein halber Punkt für IA und IS, jeweils einen halben für ii und iii und einen für iv.

Diese Seite ist für Nebenrechnungen. Bitte zugehörige Aufgabe kennzeichnen, hier und auf ihrer Seite.