ALGODAT 1 / 3. VO-PRÜFUNG / 17.06.2003
GRUPPE A

Aufgabe 1.A: Ω/O/Θ-Notation **(10 Punkte)**

a) (6 Punkte)
Gegeben sei die folgende Funktion:

$$f(n) = \begin{cases} \frac{5\log n}{3n} + \frac{7+n}{n^2}, & 2^n < 10^6 \\ \frac{3n}{5\log n} + \frac{n^2}{7+n}, & 2^n \geq 10^6 \end{cases}$$

Kreuzen Sie in der folgenden Tabelle die zutreffenden Felder an:

$f(n)$ ist	$O(.)$	$\Omega(.)$	$\Theta(.)$
n^2			
n			
$\log n$			

(Anmerkung: Jede Zeile wird jeweils nur dann gewertet, wenn alle Felder der Zeile richtig ausgefüllt sind.)

b) (4 Punkte)
Schreiben Sie einen möglichst kurzen Algorithmus in Pseudocode, dessen Laufzeit $\Theta(n \log n)$ ist.

Aufgabe 2.A: Sortieren durch Fachverteilung **(10 Punkte)**

Gegeben sind folgende oktale Zahlen (d.h., Zahlen zur Basis 8):

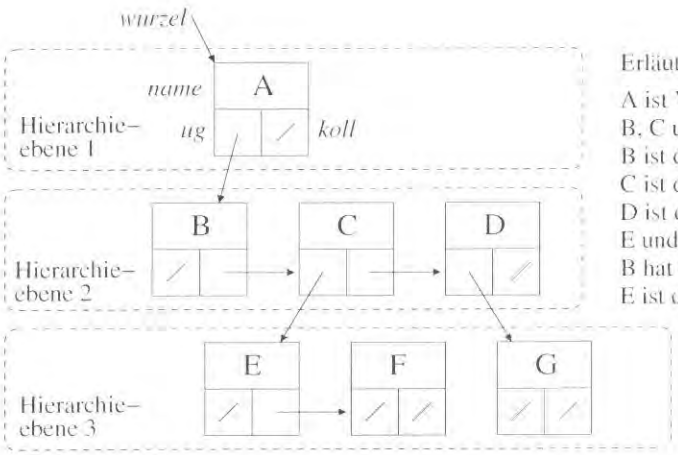
2350, 1250, 2742, 2222, 1700, 1240, 56, 317

Diese Zahlen sollen mit Hilfe des Verfahrens „Sortieren durch Fachverteilung" aufsteigend sortiert werden. Die Anzahl der Fächer entspricht der Basis 8, die Anzahl der Sammel- und Verteilungsphasen der maximalen Stellenanzahl.

Geben Sie die Inhalte aller Fächer nach jeder Verteilungsphase und die Sortierung der Zahlen nach jeder Sammelphase an.

Aufgabe 3.A: Bäume **(10 Punkte)**

a) (8 Punkte) Gegeben sei ein Organigramm einer Firma in Form eines binären Baumes. Jeder Knoten repräsentiert einen Mitarbeiter, welcher durch seinen Namen *name* eindeutig identifiziert wird (Namensgleichheiten werden hier ausgeschlossen). Der Wurzelknoten repräsentiert den Firmenleiter. Jeder Mitarbeiter kann beliebig viele direkte Untergebene haben. Dazu wird als linker Nachkomme *ug* immer der erste direkte Untergebene gespeichert. Als rechter Nachkomme *koll* wird der nächste Kollege (= nächster Mitarbeiter auf derselben Hierarchieebene, der denselben direkten Vorgesetzten hat) gespeichert. Die folgende Skizze verdeutlicht die Baumstruktur:



Erläuterungen:
A ist Vorgesetzter von B, C und D
B, C und D sind direkte Untergebene von A
B ist der erste direkte Untergebene von A
C ist der nächste Kollege von B
D ist der nächste Kollege von C
E und F sind Kollegen (Untergebene von C)
B hat keine direkten Untergebenen
E ist der erste direkte Untergebene von C

Ein Knoten enthält also folgende Informationen:

name: Name des Mitarbeiters als String
ug: Verweis auf ersten direkten Untergebenen
koll: Verweis auf nächsten Kollegen

Schreiben Sie unter Verwendung der gegebenen Datenstruktur einen effizienten Algorithmus Subordinates(m, *num*) in Pseudocode. Dieser soll für den Baum, auf dessen Wurzel m verweist, die Namen all jener Mitarbeiter ausgeben, die weniger als *num* direkte Untergebene haben.

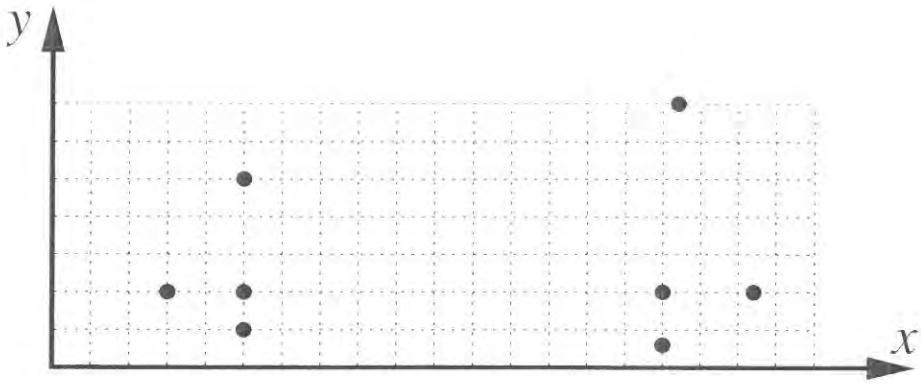
b) (2 Punkte)
Geben Sie den Aufwand Ihres Algorithmus in Θ-Notation an und begründen Sie Ihre Antwort.

Aufgabe 4.A: Grad-3-beschränkter minimaler Spannbaum **(10 Punkte)**

Gegeben sei ein vollständiger, ungerichteter Graph $G = (V, E)$. Jeder Kante $e \in E$ sei ein Gewicht $w_e \geq 0$ zugeordnet.
Ein *Grad-3-beschränkter Spannbaum* von G ist ein Spannbaum von G für den gilt, dass der Grad eines jeden Knoten maximal drei ist. Ein *minimaler* Grad-3-beschränkter Spannbaum $T \subseteq E$ ist ein Grad-3-beschränkter Spannbaum mit minimalem Gesamtgewicht $w(T) = \sum_{e \in T} w_e$.

a) (4 Punkte)
Beschreiben Sie in einfachem Pseudocode (ohne auf eventuelle Datenstrukturen genauer einzugehen) einen Algorithmus, der auf *Kruskals* Algorithmus zum Finden eines minimalen Spannbaums basiert und immer einen gültigen Grad-3-beschränkten Spannbaum mit möglichst geringem, aber nicht unbedingt minimalem, Gewicht zurückliefert.

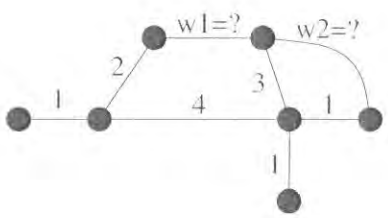
b) (3 Punkte)
Die in der folgenden Zeichnung gezeigten acht Punkte seien die Knoten eines Graphen G. Kanten existieren zwischen allen Paaren von Punkten, und die Euklidischen Abstände sind die Kantenkosten.



Wenden Sie Ihren Algorithmus für den Grad-3-beschränkten Spannbaum auf diesen Graphen an. Zeichnen Sie in die Zeichnung alle Kanten ein, die in den Spannbaum übernommen werden, und nummerieren Sie diese in der entsprechenden Reihenfolge.

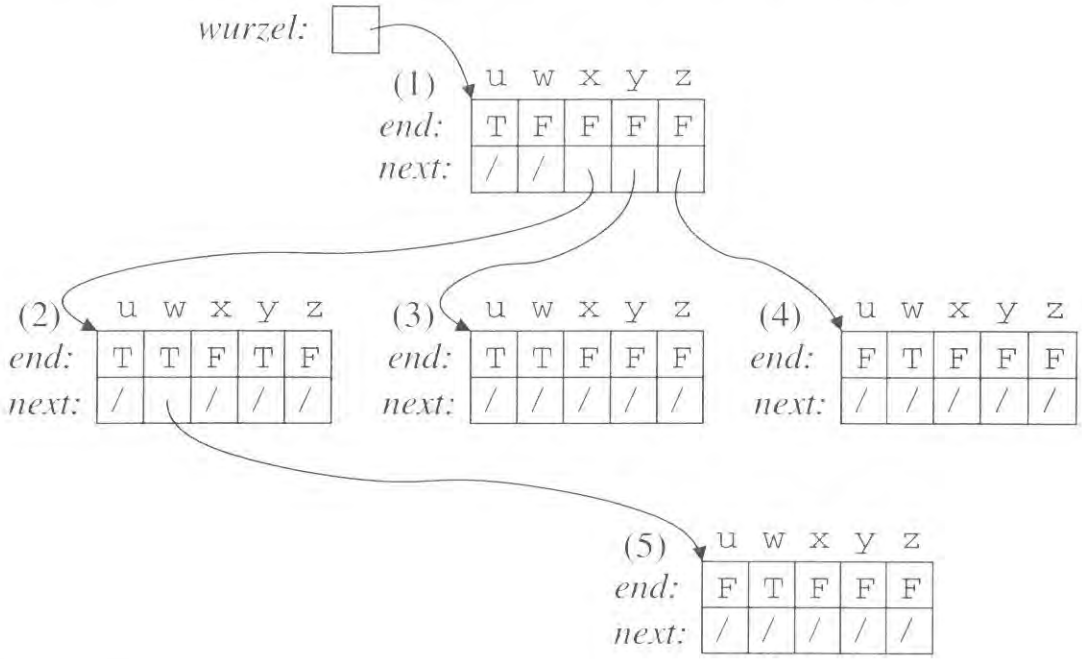
c) (3 Punkte)
Da das Finden eines Grad-3-beschränkten Spannbaums im Allgemeinen NP-schwierig ist, können Sie davon ausgehen, dass Ihr Algorithmus nicht immer einen minimalen Grad-3-beschränkten Spannbaum zurückliefert. Das sollen Sie nun mit einem Beispiel beweisen.

Nennen Sie für den folgenden nicht-Euklidischen Graphen konkrete Beispielwerte für die zwei fehlenden Kantengewichte $w1$ und $w2$, sodass Ihr Algorithmus sicher einen gültigen, aber nicht minimalen Grad-3-beschränkten Spannbaum zurückliefert. Erklären Sie die Situation auch mit einem Satz. (Um entsprechend der Definition von einem vollständigen Graphen auszugehen, nehmen Sie einfach an, dass alle nicht eingezeichneten Kanten Gewicht ∞ haben.)



Aufgabe 5.A: Suchen in Texten – Tries **(10 Punkte)**

Gegeben seien ein Alphabet Σ={'u', 'w', 'x', 'y', 'z'} und folgender Indexed Trie:



a) (4 Punkte)
Geben Sie alle Wörter an, die der oben dargestellte Indexed Trie enthält.

b) (2 Punkte)
Führen Sie Suffix Compression im oben dargestellten Indexed Trie durch. Kennzeichnen Sie die Änderungen deutlich!

c) (4 Punkte)
Aus dem resultierenden Indexed Trie mit Suffix Compression soll nun ein Packed Trie erstellt werden. Verwenden Sie dazu die Greedy-Heuristik aus der Vorlesung bzw. aus dem Skriptum. Zeigen Sie dabei mit Hilfe einer kleinen Graphik (wie in der Vorlesung bzw. im Skriptum), auf welche Weise die Knoten gepackt werden, und zeichnen Sie den Packed Trie.

ALGODAT VO/UE 27.06.2003 / GKUPPE B

Aufgabe 1.B: Ω/O/Θ-Notation **(10 Punkte)**

a) (6 Punkte)
Gegeben sei die folgende Funktion:

$$f(n) = \begin{cases} \frac{\log n}{6n^2} + \frac{8+n}{2n^3}, & 4n < 10^4 \\ \frac{6n^2}{\log n} + \frac{2n^3}{8+n}, & 4n \geq 10^4 \end{cases}$$

Kreuzen Sie in der folgenden Tabelle die zutreffenden Felder an:

$f(n)$ ist	$\Omega(.)$	$\Theta(.)$	$O(.)$
n^3			
$n \log n$			
n^2			

(Anmerkung: Jede Zeile wird jeweils nur dann gewertet, wenn alle Felder der Zeile richtig ausgefüllt sind.)

b) (4 Punkte)
Schreiben Sie einen möglichst kurzen Algorithmus in Pseudocode, dessen Laufzeit $\Theta(n^2)$ ist.

Aufgabe 2.B: Sortieren durch Fachverteilung **(10 Punkte)**

Gegeben sind folgende oktale Zahlen (d.h., Zahlen zur Basis 8):

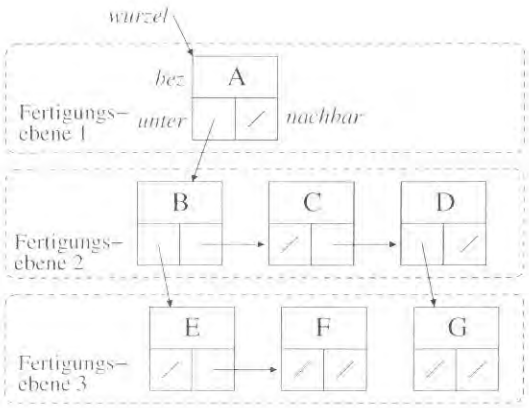
1722, 1042, 56, 2317, 2352, 1052, 740, 0

Diese Zahlen sollen mit Hilfe des Verfahrens „Sortieren durch Fachverteilung“ aufsteigend sortiert werden. Die Anzahl der Fächer entspricht der Basis 8, die Anzahl der Sammel- und Verteilungsphasen der maximalen Stellenanzahl.

Geben Sie die Inhalte aller Fächer nach jeder Verteilungsphase und die Sortierung der Zahlen nach jeder Sammelphase an.

Aufgabe 3.B: Bäume **(10 Punkte)**

a) (8 Punkte)
Gegeben sei eine hierarchische Stückliste eines Produktes in Form eines binären Baumes. Jeder Knoten repräsentiert eine Komponente, welche durch ihre Bezeichnung *bez* eindeutig identifiziert wird (Namensgleichheiten werden hier ausgeschlossen). Der Wurzelknoten repräsentiert das Fertigprodukt. Jede Komponente kann aus beliebig vielen Unterkomponenten bestehen. Dazu wird als linker Nachkomme *unter* immer die erste direkte Unterkomponente gespeichert. Als rechter Nachkomme *nachbar* wird die nächste Nachbarkomponente (= nächste Komponente auf derselben Fertigungsebene, die in derselben übergeordneten Komponente enthalten ist) gespeichert. Die folgende Skizze verdeutlicht die Baumstruktur:



Erläuterungen:
A enthält die Unterkomponenten B, C und D
B, C und D sind direkte Unterkomponenten von A
B ist die erste direkte Unterkomponente von A
C ist die nächste Nachbarkomponente von B
D ist die nächste Nachbarkomponente von C
E und F sind Nachbarkomponenten (sind in B enthalten)
C hat keine direkten Unterkomponenten
E ist die erste direkte Unterkomponente von B

Ein Knoten enthält also folgende Informationen:

bez:	Produktbezeichnung als String
unter:	Verweis auf erste direkte Unterkomponente
nachbar:	Verweis auf nächste Nachbarkomponente

Schreiben Sie unter Verwendung der gegebenen Datenstruktur einen effizienten Algorithmus Komponenten(k, *zaehl*) in Pseudocode. Dieser soll für den Baum, auf dessen Wurzel k verweist, die Namen all jener Komponenten ausgeben, die aus weniger als *zaehl* direkten Unterkomponenten bestehen.

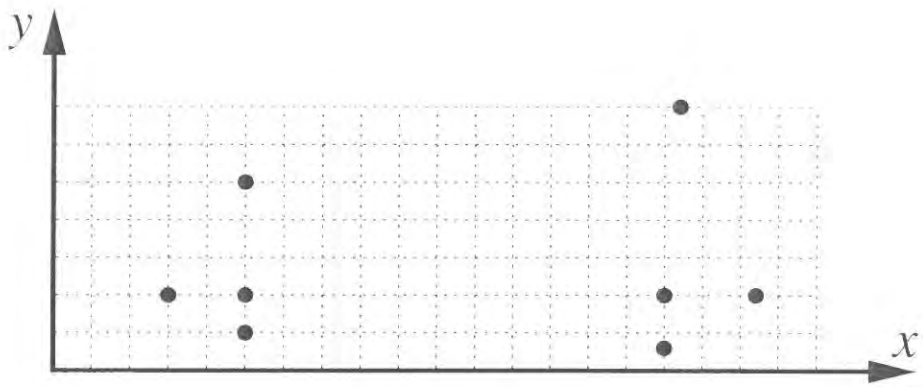
b) (2 Punkte)
Geben Sie den Aufwand Ihres Algorithmus in Θ-Notation an und begründen Sie Ihre Antwort.

Aufgabe 4.B: Grad-3-beschränkter minimaler Spannbaum **(10 Punkte)**

Gegeben sei ein vollständiger, ungerichteter Graph $G = (V, E)$. Jeder Kante $e \in E$ sei ein Gewicht $w_e \geq 0$ zugeordnet.
Ein *Grad-3-beschränkter Spannbaum* von G ist ein Spannbaum von G für den gilt, dass der Grad eines jeden Knoten maximal drei ist. Ein *minimaler* Grad-3-beschränkter Spannbaum $T \subseteq E$ ist ein Grad-3-beschränkter Spannbaum mit minimalem Gesamtgewicht $w(T) = \sum_{e \in T} w_e$.

a) (4 Punkte)
Beschreiben Sie in einfachem Pseudocode (ohne auf eventuelle Datenstrukturen genauer einzugehen) einen Algorithmus, der auf *Prims* Algorithmus zum Finden eines minimalen Spannbaums basiert und immer einen gültigen Grad-3-beschränkten Spannbaum mit möglichst geringem, aber nicht unbedingt minimalem, Gewicht zurückliefert.

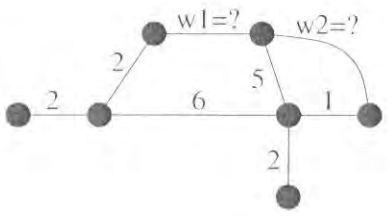
b) (3 Punkte)
Die in der folgenden Zeichnung gezeigten acht Punkte seien die Knoten eines Graphen G. Kanten existieren zwischen allen Paaren von Punkten, und die Euklidischen Abstände sind die Kantenkosten.



Wenden Sie Ihren Algorithmus für den Grad-3-beschränkten Spannbaum auf diesen Graphen an, wobei Sie mit dem Knoten mit kleinster x-Koordinate als Startknoten beginnen. Zeichnen Sie in die Zeichnung alle Kanten ein, die in den Spannbaum übernommen werden, und nummerieren Sie diese in der entsprechenden Reihenfolge.

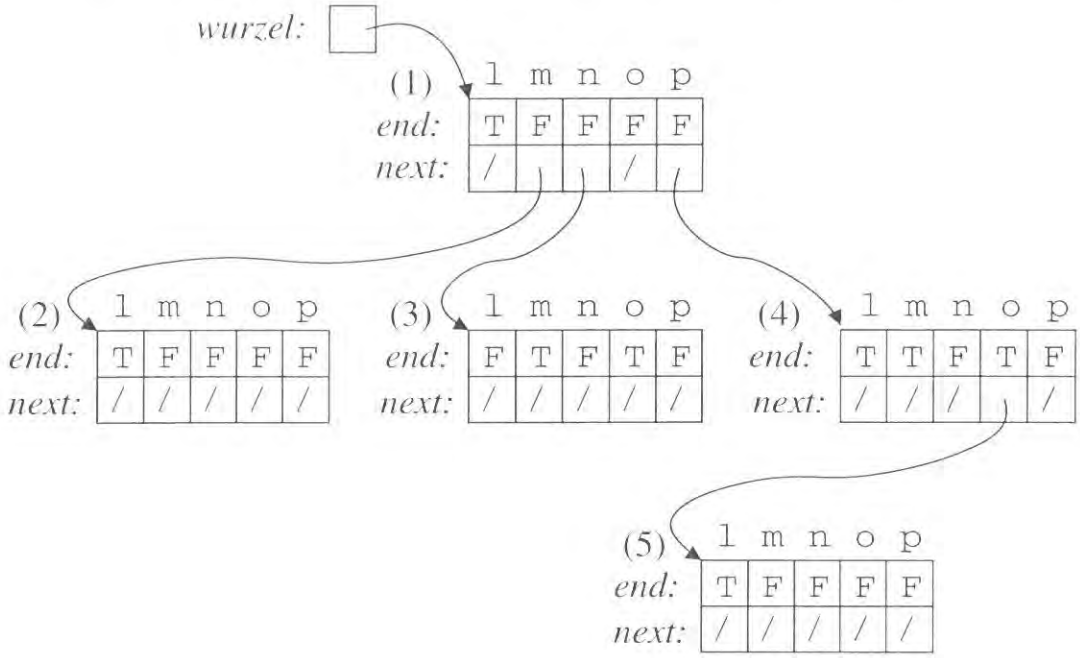
c) (3 Punkte)
Da das Finden eines Grad-3-beschränkten Spannbaums im Allgemeinen NP-schwierig ist, können Sie davon ausgehen, dass Ihr Algorithmus nicht immer einen minimalen Grad-3-beschränkten Spannbaum zurückliefert. Das sollen Sie nun mit einem Beispiel beweisen.

Nennen Sie für den folgenden nicht-Euklidischen Graphen konkrete Beispielwerte für die zwei fehlenden Kantengewichte $w1$ und $w2$, sodass Ihr Algorithmus sicher einen gültigen, aber nicht minimalen Grad-3-beschränkten Spannbaum zurückliefert, wenn der Knoten mit Grad 4 der Startknoten ist. Erklären Sie die Situation auch mit einem Satz. (Um entsprechend der Definition von einem vollständigen Graphen auszugehen, nehmen Sie einfach an, dass alle nicht eingezeichneten Kanten Gewicht ∞ haben.)



Aufgabe 5.B: Suchen in Texten – Tries **(10 Punkte)**

Gegeben seien ein Alphabet Σ={'l', 'm', 'n', 'o', 'p'} und folgender Indexed Trie:



a) (4 Punkte)
Geben Sie alle Wörter an, die der oben dargestellte Indexed Trie enthält.

b) (2 Punkte)
Führen Sie Suffix Compression im oben dargestellten Indexed Trie durch. Kennzeichnen Sie die Änderungen deutlich!

c) (4 Punkte)
Aus dem resultierenden Indexed Trie mit Suffix Compression soll nun ein Packed Trie erstellt werden. Verwenden Sie dazu die Greedy-Heuristik aus der Vorlesung bzw. aus dem Skriptum. Zeigen Sie dabei mit Hilfe einer kleinen Graphik (wie in der Vorlesung bzw. im Skriptum), auf welche Weise die Knoten gepackt werden, und zeichnen Sie den Packed Trie.

AlgoDAT VO/UE 27.06.2003 / Gruppe C

Aufgabe 1.C: $\Omega/O/\Theta$-Notation **(10 Punkte)**

a) (6 Punkte)
Gegeben sei die folgende Funktion:

$$f(n) = \begin{cases} \frac{9\log n}{2n} + \frac{6+n}{n^2}, & 3^n < 10^5 \\ \frac{2n}{9\log n} + \frac{n^2}{6+n}, & 3^n \geq 10^5 \end{cases}$$

Kreuzen Sie in der folgenden Tabelle die zutreffenden Felder an:

$f(n)$ ist	$\Theta(.)$	$O(.)$	$\Omega(.)$
$n \log n$			
n			
$\log n$			

(Anmerkung: Jede Zeile wird jeweils nur dann gewertet, wenn alle Felder der Zeile richtig ausgefüllt sind.)

b) (4 Punkte)
Schreiben Sie einen möglichst kurzen Algorithmus in Pseudocode, dessen Laufzeit $\Theta(n \log n)$ ist.

Aufgabe 2.C: Sortieren durch Fachverteilung **(10 Punkte)**

Gegeben sind folgende oktale Zahlen (d.h., Zahlen zur Basis 8):

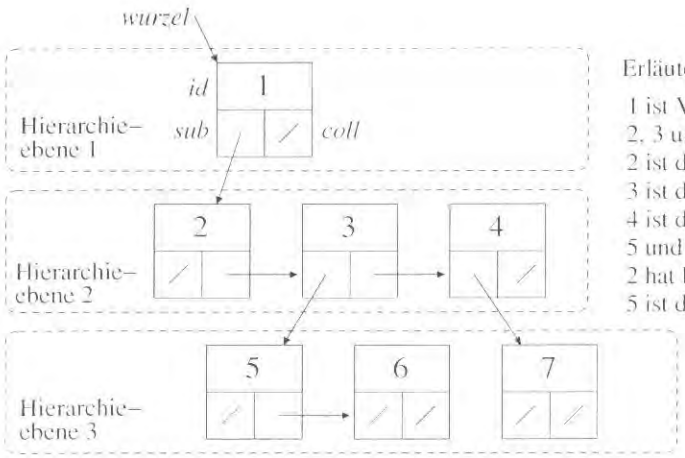
1356, 1250, 3174, 4444, 1744, 1240, 56, 317

Diese Zahlen sollen mit Hilfe des Verfahrens „Sortieren durch Fachverteilung“ aufsteigend sortiert werden. Die Anzahl der Fächer entspricht der Basis 8, die Anzahl der Sammel- und Verteilungsphasen der maximalen Stellenanzahl.

Geben Sie die Inhalte aller Fächer nach jeder Verteilungsphase und die Sortierung der Zahlen nach jeder Sammelphase an.

Aufgabe 3.C: Bäume **(10 Punkte)**

a) (8 Punkte)
Gegeben sei ein Organigramm einer Firma in Form eines binären Baumes. Jeder Knoten repräsentiert einen Mitarbeiter, welcher durch seine Personalnummer *id* eindeutig identifiziert wird. Der Wurzelknoten repräsentiert den Firmenleiter. Jeder Mitarbeiter kann beliebig viele direkte Untergebene haben. Dazu wird als linker Nachkomme *sub* immer der erste direkte Untergebene gespeichert. Als rechter Nachkomme *coll* wird der nächste Kollege (= nächster Mitarbeiter auf derselben Hierarchieebene, der denselben direkten Vorgesetzten hat) gespeichert. Die folgende Skizze verdeutlicht die Baumstruktur:



Erläuterungen:
1 ist Vorgesetzter von 2, 3 und 4
2, 3 und 4 sind direkte Untergebene von A
2 ist der erste direkte Untergebene von 1
3 ist der nächste Kollege von 2
4 ist der nächste Kollege von 3
5 und 6 sind Kollegen (Untergebene von 3)
2 hat keine direkten Untergebenen
5 ist der erste direkte Untergebene von 3

Ein Knoten enthält also folgende Informationen:

id: Personalnummer des Mitarbeiters
sub: Verweis auf ersten direkten Untergebenen
coll: Verweis auf nächsten Kollegen

Schreiben Sie unter Verwendung der gegebenen Datenstruktur einen effizienten Algorithmus Subordinates(s, nr). Dieser soll für den Baum, auf dessen Wurzel s verweist, die Namen all jener Mitarbeiter ausgeben, die mehr als nr direkte Untergebene haben.

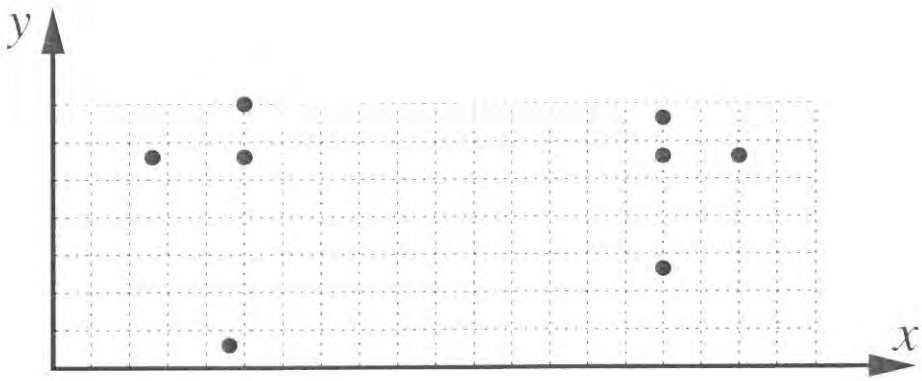
b) (2 Punkte)
Geben Sie den Aufwand Ihres Algorithmus in Θ-Notation an und begründen Sie Ihre Antwort.

Aufgabe 4.C: Grad-3-beschränkter minimaler Spannbaum **(10 Punkte)**

Gegeben sei ein vollständiger, ungerichteter Graph $G = (V, E)$. Jeder Kante $e \in E$ sei ein Gewicht $w_e \geq 0$ zugeordnet.
Ein *Grad-3-beschränkter Spannbaum* von G ist ein Spannbaum von G für den gilt, dass der Grad eines jeden Knoten maximal drei ist. Ein *minimaler* Grad-3-beschränkter Spannbaum $T \subseteq E$ ist ein Grad-3-beschränkter Spannbaum mit minimalem Gesamtgewicht $w(T) = \sum_{e \in T} w_e$.

a) (4 Punkte)
Beschreiben Sie in einfachem Pseudocode (ohne auf eventuelle Datenstrukturen genauer einzugehen) einen Algorithmus, der auf *Kruskals* Algorithmus zum Finden eines minimalen Spannbaums basiert und immer einen gültigen Grad-3-beschränkten Spannbaum mit möglichst geringem, aber nicht unbedingt minimalem, Gewicht zurückliefert.

b) (3 Punkte)
Die in der folgenden Zeichnung gezeigten acht Punkte seien die Knoten eines Graphen G. Kanten existieren zwischen allen Paaren von Punkten, und die Euklidischen Abstände sind die Kantenkosten.

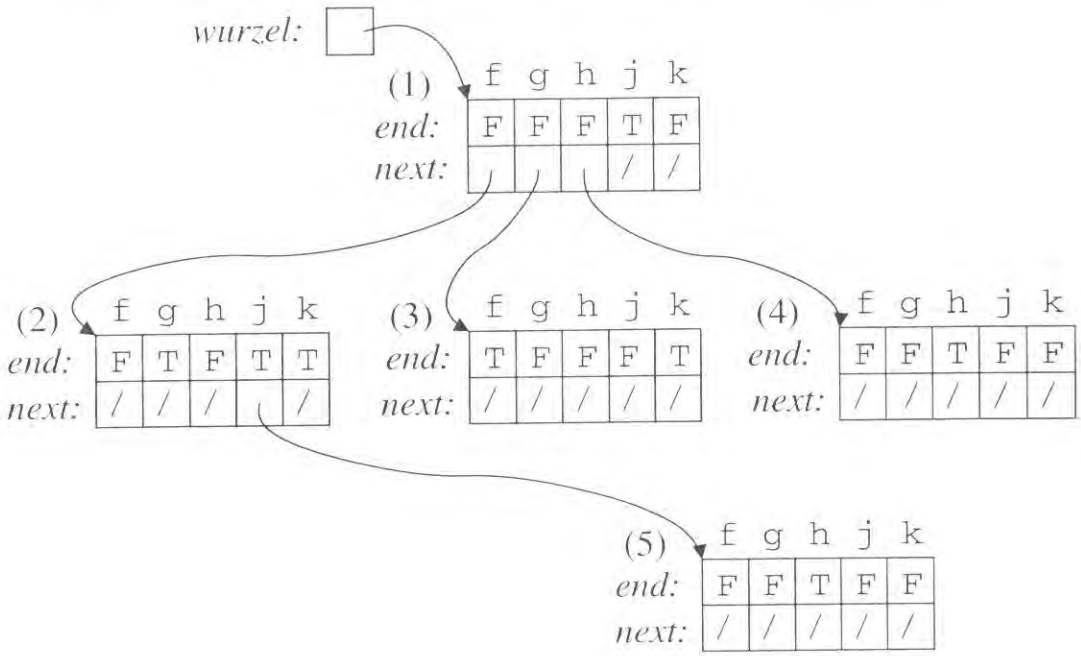


Wenden Sie Ihren Algorithmus für den Grad-3-beschränkten Spannbaum auf diesen Graphen an. Zeichnen Sie in die Zeichnung alle Kanten ein, die in den Spannbaum übernommen werden, und nummerieren Sie diese in der entsprechenden Reihenfolge.

c) (3 Punkte)
Erklären Sie an Hand eines möglichst einfachen Beispiels, warum Ihr Algorithmus für unvollständige Graphen G nicht immer einen gültigen Grad-3-beschränkten Spannbaum zurückliefern kann, obwohl ein solcher existiert.

Aufgabe 5.C: Suchen in Texten – Tries **(10 Punkte)**

Gegeben seien ein Alphabet Σ={'f', 'g', 'h', 'j', 'k'} und folgender Indexed Trie:



a) (4 Punkte)
Geben Sie alle Wörter an, die der oben dargestellte Indexed Trie enthält.

b) (2 Punkte)
Führen Sie Suffix Compression im oben dargestellten Indexed Trie durch. Kennzeichnen Sie die Änderungen deutlich!

c) (4 Punkte)
Aus dem resultierenden Indexed Trie mit Suffix Compression soll nun ein Packed Trie erstellt werden. Verwenden Sie dazu die Greedy-Heuristik aus der Vorlesung bzw. aus dem Skriptum. Zeigen Sie dabei mit Hilfe einer kleinen Graphik (wie in der Vorlesung bzw. im Skriptum), auf welche Weise die Knoten gepackt werden, und zeichnen Sie den Packed Trie.

ALGOBAT VO/UE 27.06.2003 / Gruppe D

Aufgabe 1.D: $\Omega/O/\Theta$-Notation **(10 Punkte)**

a) (6 Punkte)
Gegeben sei die folgende Funktion:

$$f(n) = \begin{cases} \frac{\log n}{4n^2} + \frac{3+n}{7n^3}, & 3n < 10^4 \\ \frac{4n^2}{\log n} + \frac{7n^3}{3+n}, & 3n \geq 10^4 \end{cases}$$

Kreuzen Sie in der folgenden Tabelle die zutreffenden Felder an:

$f(n)$ ist	$\Omega(.)$	$O(.)$	$\Theta(.)$
n^2			
n			
$n \log n$			

(Anmerkung: Jede Zeile wird jeweils nur dann gewertet, wenn alle Felder der Zeile richtig ausgefüllt sind.)

b) (4 Punkte)
Schreiben Sie einen möglichst kurzen Algorithmus in Pseudocode, dessen Laufzeit $\Theta(n^2)$ ist.

Aufgabe 2.D: Sortieren durch Fachverteilung **(10 Punkte)**

Gegeben sind folgende oktale Zahlen (d.h., Zahlen zur Basis 8):

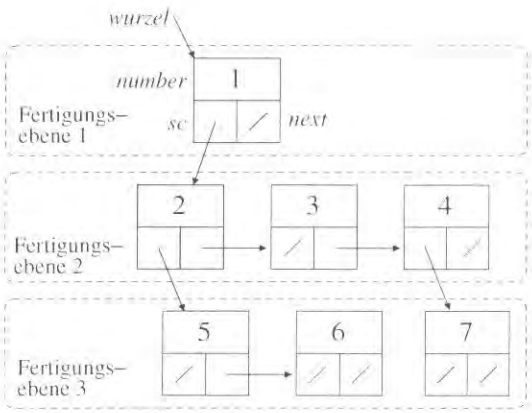
1533, 1523, 742, 22, 1747, 1253, 1156, 2313

Diese Zahlen sollen mit Hilfe des Verfahrens „Sortieren durch Fachverteilung“ aufsteigend sortiert werden. Die Anzahl der Fächer entspricht der Basis 8, die Anzahl der Sammel- und Verteilungsphasen der maximalen Stellenanzahl.

Geben Sie die Inhalte aller Fächer nach jeder Verteilungsphase und die Sortierung der Zahlen nach jeder Sammelphase an.

Aufgabe 3.D: Bäume **(10 Punkte)**

a) (8 Punkte)
Gegeben sei eine hierarchische Stückliste eines Produktes in Form eines binären Baumes. Jeder Knoten repräsentiert eine Komponente, welche durch ihre Produktnummer *number* eindeutig identifiziert wird. Der Wurzelknoten repräsentiert das Fertigprodukt. Jede Komponente kann aus beliebig vielen Unterkomponenten bestehen. Dazu wird als linker Nachkomme *sc* immer die erste direkte Unterkomponente gespeichert. Als rechter Nachkomme *next* wird die nächste Nachbarkomponente (= nächste Komponente auf derselben Fertigungsebene, die in derselben übergeordneten Komponente enthalten ist) gespeichert. Die folgende Skizze verdeutlicht die Baumstruktur:



Erläuterungen:
1 enthält die Unterkomponenten 2, 3 und 4
2, 3 und 4 sind direkte Unterkomponenten von 1
2 ist die erste direkte Unterkomponente von 1
3 ist die nächste Nachbarkomponente von 2
4 ist die nächste Nachbarkomponente von 3
5 und 6 sind Nachbarkomponenten (sind in 2 enthalten)
3 hat keine direkten Unterkomponenten
5 ist die erste direkte Unterkomponente von 3

Ein Knoten enthält also folgende Informationen:

number:	Produktbezeichnung als String
sc:	Verweis auf erste direkte Unterkomponente
next:	Verweis auf nächste Nachbarkomponente

Schreiben Sie unter Verwendung der gegebenen Datenstruktur einen effizienten Algorithmus Subcomponents(c, *count*). Dieser soll für den Baum, auf dessen Wurzel c verweist, die Namen all jener Komponenten ausgeben, die aus mehr als *count* direkten Unterkomponenten bestehen.

b) (2 Punkte)
Geben Sie den Aufwand Ihres Algorithmus in Θ-Notation an und begründen Sie Ihre Antwort.

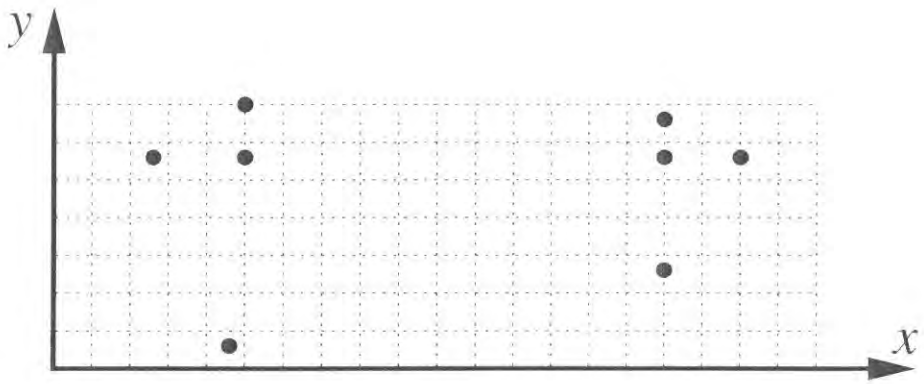
Aufgabe 4.D: Grad-3-beschränkter minimaler Spannbaum **(10 Punkte)**

Gegeben sei ein vollständiger, ungerichteter Graph $G = (V, E)$. Jeder Kante $e \in E$ sei ein Gewicht $w_e \geq 0$ zugeordnet.
Ein *Grad-3-beschränkter Spannbaum* von G ist ein Spannbaum von G für den gilt, dass der Grad eines jeden Knoten maximal drei ist. Ein *minimaler* Grad-3-beschränkter Spannbaum $T \subseteq E$ ist ein Grad-3-beschränkter Spannbaum mit minimalem Gesamtgewicht $w(T) = \sum_{e \in T} w_e$.

a) (4 Punkte)
Beschreiben Sie in einfachem Pseudocode (ohne auf eventuelle Datenstrukturen genauer einzugehen) einen Algorithmus, der auf *Prims* Algorithmus zum Finden eines minimalen Spannbaums basiert und immer einen gültigen Grad-3-beschränkten Spannbaum mit möglichst geringem, aber nicht unbedingt minimalem, Gewicht zurückliefert.

b) (3 Punkte)
Erklären Sie an Hand eines möglichst einfachen Beispiels, warum Ihr Algorithmus für unvollständige Graphen G nicht immer einen gültigen Grad-3-beschränkten Spannbaum zurückliefern kann, trotzdem ein solcher existiert.

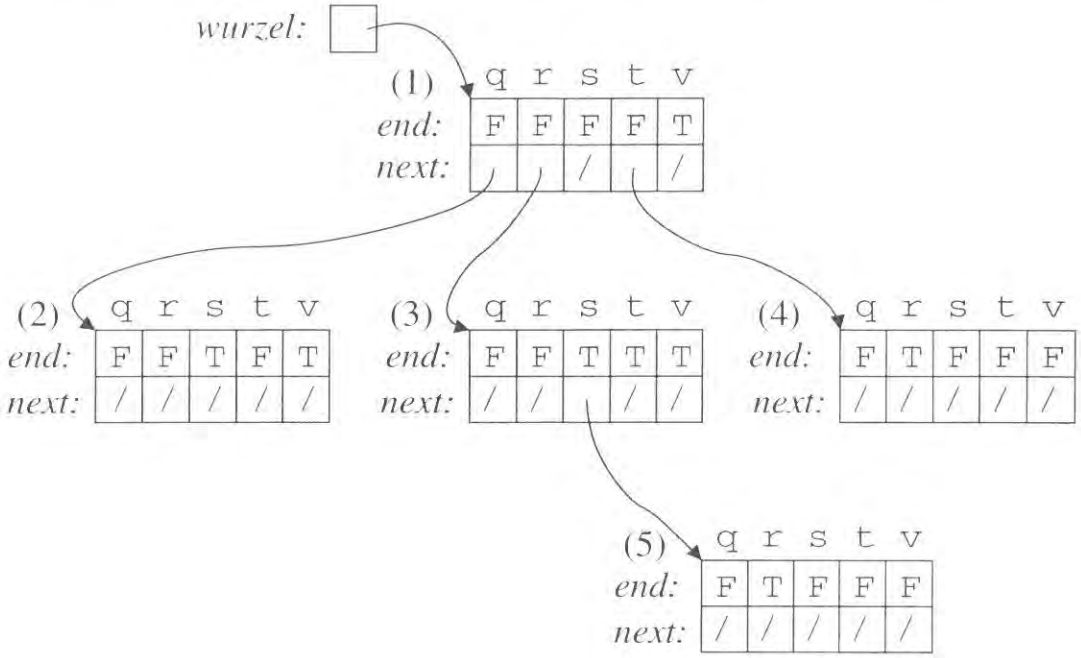
c) (3 Punkte)
Die in der folgenden Zeichnung gezeigten acht Punkte seien die Knoten eines Graphen G. Kanten existieren zwischen allen Paaren von Punkten, und die Kantenkosten sind die Euklidischen Abstände.



Wenden Sie Ihren Algorithmus für den Grad-3-beschränkten Spannbaum auf diesen Graphen an, wobei Sie mit dem Knoten mit kleinster x-Koordinate als Startknoten beginnen. Zeichnen Sie in die Zeichnung alle Kanten ein, die in den Spannbaum übernommen werden, und nummerieren Sie diese in der entsprechenden Reihenfolge.

Aufgabe 5.D: Suchen in Texten – Tries **(10 Punkte)**

Gegeben seien ein Alphabet Σ={'q', 'r', 's', 't', 'v'} und folgender Indexed Trie:



a) (4 Punkte)
Geben Sie alle Wörter an, die der oben dargestellte Indexed Trie enthält.

b) (2 Punkte)
Führen Sie Suffix Compression im oben dargestellten Indexed Trie durch. Kennzeichnen Sie die Änderungen deutlich!

c) (4 Punkte)
Aus dem resultierenden Indexed Trie mit Suffix Compression soll nun ein Packed Trie erstellt werden. Verwenden Sie dazu die Greedy-Heuristik aus der Vorlesung bzw. aus dem Skriptum. Zeigen Sie dabei mit Hilfe einer kleinen Graphik (wie in der Vorlesung bzw. im Skriptum), auf welche Weise die Knoten gepackt werden, und zeichnen Sie den Packed Trie.