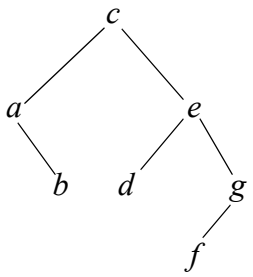
Aufgabe 1 (Binäre Bäume)

40 Punkte



(a) Geben Sie zu dem oben abgebildeten binären Baum die Ergebnisse von *inorder*-, *postorder*- und *preorder*-Durchlauf an. *3 Punkte*

(b) Gegeben sei ein binärer Baum, in dessen Knoten Buchstaben gespeichert sind. Sie benötigen den exakten Aufbau des Baumes, es stehen Ihnen aber nur drei Prozeduren *inorderproc*, *preorderproc* und *postorderproc* zur Verfügung, die den Inhalt besagten Baums in einer ihrem Namen gemäßen Weise ausgeben. Die Prozedur *inorderproc* liefert *12 Punkte*

rwsyzvxntpuo

Das Ergebnis eines *preorderproc*-Durchlaufs ist

zywrsxvutnpo

Schließlich erhält man von *postorderproc* die Ausgabe

rswyvnptouxz

Rekonstruieren Sie aus diesen Angaben den Aufbau und Inhalt des Binärbaums. Beschreiben Sie kurz Ihre Überlegungen zur Konstruktion des Baumes.

(c) Geben Sie einen kommentierten Algorithmus an, der die Anzahl aller Knoten eines beliebigen binären Baums ausgibt. *5 Punkte*

(d) Geben Sie einen kommentierten Algorithmus an, der die Anzahl aller Blätter eines binären Baums ausgibt. *5 Punkte*

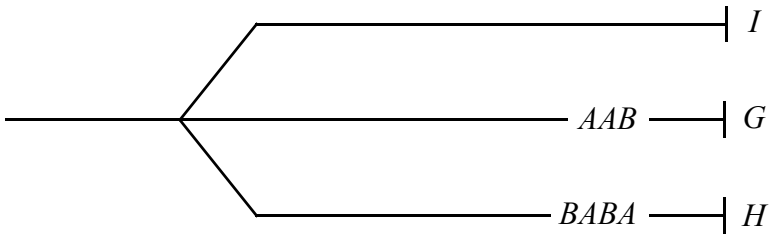
(e) Geben Sie einen kommentierten Algorithmus an, der die Anzahl der inneren Knoten eines binären Baums ausgibt. *5 Punkte*

(f) In den Knoten eines Binärbaums seien Daten des Typs *elem* gespeichert. Geben Sie einen Algorithmus an, der bei Eingabe eines Wertes vom Typ *elem* die Tiefe desjenigen Knotens, in dem dieser Wert gespeichert ist, bestimmt. Sie können davon ausgehen, dass der zu betrachtende Binärbaum nicht leer ist und dass Elemente nicht mehrfach im Baum gespeichert sind. *10 Punkte*

30 Punkte

Aufgabe 2 (Rangierbahnhof)

In einem Güterbahnhof befindet sich eine Weiche zu drei Gleisen, die Sackgassen sind. Auf den Gleisen *G* und *H* befinden sich zwei Züge mit Waggons für die Zielbahnhöfe *A* und *B*. Gleis *I* sei leer. Diese Situation wird in folgender Abbildung dargestellt:



20Punkte

(a) Betrachten Sie im folgenden *G*, *H* und *I* als Stacks. Auf *G* und *H* stehen zwei beliebige Züge aus insgesamt *n* Waggons für die Zielbahnhöfe *A* und *B*. Geben Sie einen Algorithmus an, der diese Züge derart umordnet, daß anschließend auf *G* alle Waggons für *A* und auf *H* alle Waggons für *B* stehen. Verwenden Sie außer Kontrollstrukturen lediglich die Stackoperationen *push*, *pop*, *top* und *isEmpty*. Beachten Sie, daß ihr Algorithmus nicht mehr als $2n$ Verschiebungen durchführen darf.

10 Punkte

(b) Wieviele Verschiebe-Operationen benötigt der Algorithmus, wenn a_G (bzw. b_G) Waggons auf dem Gleis *G* für den Zielbahnhof *A* (bzw. *B*) stehen und a_H (bzw. b_H) Waggons auf dem Gleis *H* für den Zielbahnhof *A* (bzw. *B*) stehen?

30 Punkte

Aufgabe 3 (Der Datentyp Vector)

In vielen Programmiersprachen wird über Bibliotheken eine dynamische Datenstruktur Vector angeboten, die den Zugriff auf darin gespeicherte Elemente wie in einem Array über einen Index ermöglicht. Der Zugriff auf eine beliebige Indexposition erfordert Laufzeit O(1). Dies wird meistens dadurch erreicht, daß dieser Typ intern ein Array der initialen Größe *m* benutzt, dessen Inhalt in ein Array doppelter Größe kopiert wird, sobald die Kapazität ausgeschöpft ist. Nach *k* Verdoppelungen beträgt die Größe also $2^k m$. Zusätzlich merkt sich ein Vector die nächste freie Indexposition und bietet eine Operation *append* an, die ein Element auf dieser Position ablegt.

15 Punkte

(a) Welche Kosten entstehen im worst case, wenn *N* Elemente mittels *append* in einen anfangs leeren Vector abgelegt werden sollen, und wieviel reservierter Speicherplatz bleibt ungenutzt? Bitte geben Sie eine genaue Kostenfunktion,

in der die Kosten für das Auslesen oder Zuweisen eines Array-Elementes mit *S* bezeichnet werden, an.

(b) Überlegen Sie sich eine alternative Implementierung, die den Direktzugriff in O(1) unterstützt, aber im worst case weniger ungenutzten Speicherplatz belegt. Falls Objekte mit hohem Speicherplatzbedarf gespeichert werden, sollen außerdem geringere Kosten für die Zuweisung von *N* Elementen mittels *append* in einen anfangs leeren Vector entstehen. Belegen Sie die geforderten Eigenschaften anhand einer möglichst genauen Kostenfunktion. *15 Punkte*