Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer
Kennzahl: _____ Kennwort: _____ Arbeitsplatz-Nr.: _____	HERBST 1989	66111

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

- Prüfungsaufgaben -

Fach: Informatik (nicht vertieft studiert)
Einzelprüfung: Betriebs/Datenbanksyst., Rechn.architektur
Anzahl der gestellten Themen (Aufgaben): 1
Anzahl der Druckseiten dieser Vorlage: 6

bitte wenden!

Sämtliche Teilaufgaben sind zu bearbeiten!

Teilaufgabe 1

Für einen endlichen nicht-leeren Zeichenvorrat Z sei eine Binärcodierung

$$c : Z \to \{O, L\}^*$$

gegeben.

1.1 Geben Sie eine Datenstruktur (durch Typdefinitionen) an, mit deren Hilfe ein beliebiger Binärcode c im Hinblick auf die folgende Teilaufgabe 1.2 geeignet dargestellt werden kann.

1.2 Schreiben Sie eine Funktionsprozedur *fano*, mit deren Hilfe festgestellt werden kann, ob ein Binärcode c der Fano-Bedingung genügt. Die Funktionsprozedur *fano* soll dabei folgender Spezifikation genügen:

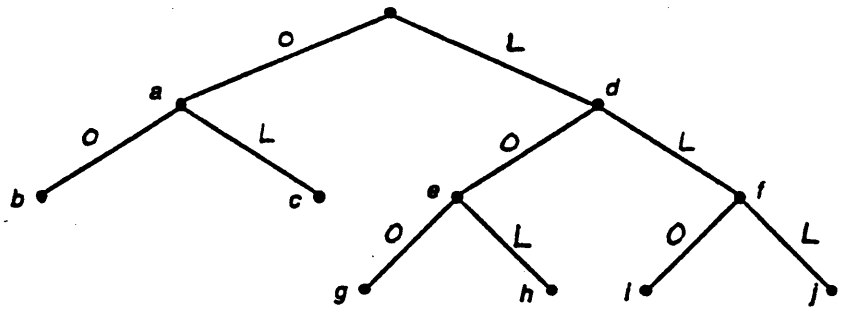
1.2.1 Eingabeparameter: c vom Typ *CODE*

1.2.2 Der Funktionswert ist vom Typ *BOOLEAN* und hat den Wert *TRUE* genau dann, wenn c der Fano-Bedingung genügt.

Hinweise: - Der Typ *CODE* muß in Ihrer Antwort zu 1.1 definiert werden.

- Ein Code erfüllt die Fano-Bedingung genau dann, wenn in ihm kein Codewort Anfang eines von ihm verschiedenen Codewortes ist.

1.3 Der Zeichenvorrat $Z = \{a, b, c, d, e, f, g, h, i, j\}$ läßt sich durch Wörter aus $\{O, L\}^*$ binär codieren, die höchstens die Länge 3 haben. Ein Beispiel dafür ist durch den folgenden Code-Baum gegeben:

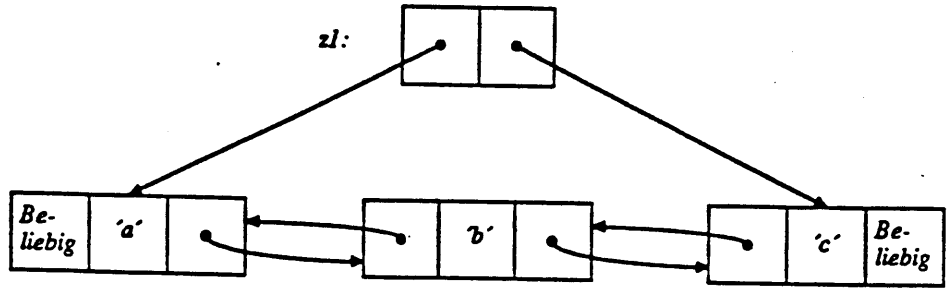


Zeigen Sie, daß es keine 3-stellige Binärcodierung (d.h. mit Codewortlänge ≤ 3) für Z gibt, die die Fano-Bedingung erfüllt.

Fortsetzung nächste Seite!

Teilaufgabe 2

Zeichenreihen über dem endlichen Alphabet $A = ('a','b',\ldots,'z')$ sollen durch vorwärts und rückwärts verkettete Listen dargestellt werden, für deren Anfang und Ende Anker (Verweise) vorgesehen sind. Die Darstellung der Zeichenreihe $z1$='*abc*' läßt sich also beispielsweise folgendermaßen graphisch veranschaulichen:



2.1 Geben Sie eine Datenstruktur mit Hilfe geeigneter Typdefinitionen an, durch die sich die o.a. Darstellung von Zeichenreihen realisieren läßt.

2.2 Wie kann die leere Zeichenreihe ϵ dargestellt werden?

2.3 Schreiben Sie eine Funktionsprozedur *istepsilon*, die feststellt, ob eine Zeichenreihe gleich ϵ ist, also folgender Spezifikation genügt:

2.3.1 Eingabeparameter: $x1$ vom Typ *ZEICHENREIHE*

Hinweis: *ZEICHENREIHE* muß als Typ in Ihrer Antwort zu 2.1 definiert sein.

2.3.2 Der Funktionswert ist vom Typ *BOOLEAN* und ist genau dann gleich *TRUE*, wenn $x1 = \epsilon$ ist.

Fortsetzung nächste Seite!

2.4 Schreiben Sie zwei Funktionsprozeduren *anfang* und *teil*, die feststellen, ob x Anfangs- bzw. Teilzeichenreihe einer Zeichenreihe y ist. Die entsprechenden Spezifikationen lauten:

2.4.1 Eingabeparameter (für beide Funktionsprozeduren): x, y vom Typ *ZEICHENREIHE*

2.4.2 Der Funktionswert ist jeweils vom Typ *BOOLEAN*. Er hat den Wert *TRUE* für die Funktion *anfang* bzw. *teil* genau dann, wenn x Anfangs- bzw. Teilzeichenreihe von y ist.

Hinweise:
- x ist genau dann Anfangszeichenreihe von y, wenn es ein y' so gibt, daß $y = xy'$ ist.
- x ist genau dann Teilzeichenreihe von y, wenn es ein y' und ein y'' so gibt, daß $y = y'xy''$ ist.
- Es ist empfehlenswert, erst die Funktionsprozedur *anfang* zu schreiben und dann in der Funktionsprozedur *teil* die Funktion *anfang* zu verwenden. Dabei darf auch die in 2.3 spezifizierte Funktion *istepsilon* verwendet werden.

2.5 Ein Palindrom ist eine Zeichenreihe, die mit ihrer rückwärts gelesenen identisch ist. Beispiele sind

'*otto*', '*reliefpfeiler*', ϵ (leere Zeichenreihe).

Schreiben Sie eine Funktionsprozedur *palin*, die feststellt, ob eine Zeichenreihe ein Palindrom ist, also folgender Spezifikation genügt:

2.5.1 Eingabeparameter: $x1$ vom Typ *ZEICHENREIHE*

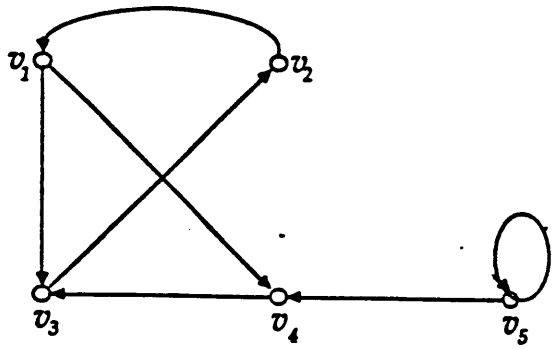
2.5.2 Der Funktionswert ist vom Typ *BOOLEAN* und ist genau dann gleich *TRUE*, wenn $x1$ ein Palindrom ist.

Hinweis: In der Funktionsprozedur *palin* darf die unter 2.3 spezifizierte Funktion *istepsilon* verwendet werden.

Fortsetzung nächste Seite!

Teilaufgabe 3

Ein endlicher gerichteter Graph $G = (V, E)$ mit der Knotenmenge $V = \{v_1, v_2, \ldots, v_n\}$ und der Menge gerichteter Kanten $E \subseteq V \times V$ kann u.a. durch seine direkte graphische Darstellung und durch seine Adjazenzmatrix dargestellt werden. Ein Beispiel:



Dieser direkt dargestellte Graph mit $V = \{v_1, v_2, v_3, v_4, v_5\}$ und $E = \{(v_1, v_3), (v_1, v_4), (v_2, v_1), (v_3, v_2), (v_4, v_3), (v_5, v_4), (v_5, v_5)\}$ läßt sich auch durch seine Adjazenzmatrix

$$A = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

darstellen. Allgemein gilt für die Komponenten a_{ij} der Adjazenzmatrix A:

$$a_{ij} = \begin{cases} 1 & \text{falls es eine Kante von } v_i \text{ nach } v_j \text{ gibt} \\ 0 & \text{sonst} \end{cases}$$

Hat ein gerichteter Graph die Eigenschaft, daß von wenigstens einem Knoten zwei gerichtete Kanten und von keinem Knoten mehr als zwei gerichtete Kanten ausgehen, so sagt man, daß er den Verzweigungsgrad 2 hat. Der angegebene Beispiel-Graph hat also den Verzweigungsgrad 2. Solche Graphen können in direkter Darstellung durch einen Verweis auf ein Geflecht von Verbunden der Art *knoten* entsprechend folgender Typ-Vereinbarung aufgebaut werden:

```
TYPE graph  = ↑ knoten;
     knoten = RECORD inhalt: INTEGER;
                     nachf1, nachf2: graph
              END;
```

Mit VAR *g: graph* sei nun eine Variable gegeben, die auf einen endlichen gerichteten Graphen mit n Knoten und mit Verzweigungsgrad 2 verweist.

Fortsetzung nächste Seite!

3.1 Geben Sie einen terminierenden Algorithmus an, der die Adjazenzmatrix von g berechnet.

Hinweis: Es wird nicht verlangt, daß dafür ein Programm geschrieben wird; der Algorithmus soll aber klar und korrekt beschrieben werden.

3.2 Als Wegematrix W zu einem endlichen gerichteten Graphen G bezeichnet man die Matrix, die durch ihre Komponente w_{ij} angibt, ob es in G einen gerichteten Kantenzug von v_i nach v_j gibt. $W = (w_{ij})_{1 \leq i,j \leq n}$ ist also folgendermaßen definiert:

$$w_{ij} = \begin{cases} 1 & \text{falls ein gerichteter Kantenzug von } v_i \text{ nach } v_j \text{ existiert} \\ 0 & \text{sonst} \end{cases}$$

Zu dem o.a. Graphen ist beispielsweise

$$W = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

die zugehörige Wegematrix.

Schreiben Sie eine Prozedur *weg*, die zur Adjazenzmatrix A eines endlichen gerichteten Graphen mit n Knoten und mit einem beliebigen Verzweigungsgrad ($\leq n$) die Wegematrix W berechnet. Die Prozedur *weg* soll also folgender Spezifikation genügen:

3.2.1 Eingabeparameter: Adjazenzmatrix a vom Typ `ARRAY[1..n,1..n]OF 0..1`

3.2.2 Ausgabeparameter: Wegematrix w vom Typ `ARRAY[1..n,1..n]OF 0..1`
Dabei soll w die Wegematrix zu dem durch a dargestellten gerichteten endlichen Graphen sein.