Klausur zu Programmieren I

Wintersemster 94/95

11. Februar 95

Name:

Vorname:

Martikelnummer:

Die Aufgaben 1 - 4 bilden den Theorie-Teil, die Aufgaben 5 - 8 den Praxis-Teil. In jedem Teil ist mindestens die Hälfte der Punkte zu erreichen!

Bearbeitungszeit: 120 min.

Viel Erfolg!

Aufgabe	Punkte	
1	5	
2	5	
3	5	
4		
a)	5	
b)	8	
Σ	28	
5	10	
6	6	
7	6	
8		
a)	6	
b)	10	
Σ	38	

Aufgabe 1:

Telefonnummern werden im internationalen Gebrauch häufig wie folgt angegeben:

’(+’ L ’)’ O ’-’ A ’-’ D

Dabei stehen die Buchstaben für Ziffernfolgen: L für Land, O für Ortsnetz, A für Anschluß, D für Durchwahl (Nebenstelle); der letzte Teil "’-’ D" ist optional. Jede Folge soll mindestens aus einer Ziffer bestehen. Die Telefonnummer besteht ausschließlich aus Ziffern und den in ’ gesetzten Zeichen. Schreiben Sie für *egrep* einen regulären Ausdruck, so daß aus einer Datei, die am Ende jeder Zeile (also vor dem abschließenden *newline*) eine solche Telefonnummer enthalten soll, alle "fehlerhaften" Zeilen herausgefiltert werden. Das Kommando sieht also wie folgt aus:

egrep -v ’<ihr Ausdruck>’ file

Aufgabe 2:

Gegeben sei folgendes Programmfragment:

```
VAR a,b,c : BOOLEAN;
 ...
IF a OR NOT (b AND c) THEN
        ...
ELSIF NOT a OR NOT b OR NOT c THEN
        ...
ELSIF NOT c THEN
        ...
ELSE (*wie hierher*)
END
```

Geben Sie den logischen Ausdruck an, der die Bedingung, unter der der ELSE-Teil erreicht wird, beschreibt. Der Ausdruck soll unter Angabe der Zwischenschritte mittels der Regeln von de Morgan möglichst stark vereinfacht werden!

Aufgabe 3:

Gegeben sei folgende (reguläre) Grammatik:

NT = { Z,R,S }; (Non-Terminalsymbole)

Startsymbol: Z

T = { 0, 1 }; (Terminalsymbole)

P = {

$Z \rightarrow 0$

$Z \rightarrow 1S$

$Z \rightarrow 0Z$

$S \rightarrow 1$

$S \rightarrow 0R$

$S \rightarrow 1Z$

$R \rightarrow 1R$

$R \rightarrow 0S$

} (Menge der Produktionsregeln)

Zeigen Sie, daß das Wort 11011 zu der durch diese Grammatik definierten Sprache gehört!

Aufgabe 4:

Gegeben sei der reguläre Ausdruck (1|(0|1)*11).

a) Geben Sie einen endlichen Automaten an, der die durch den regulären Ausdruck definierte Sprache erkennt.

b) Geben Sie eine Grammatik an, die dieselbe Sprache beschreibt.

Aufgabe 5:

Gegeben seien folgende Vereinbarungen:

```
TYPE        StatSet = SET OF [0..31];
VAR         set, partSet: StatSet;
            n,m: [0..31];
```

Eine Relation $PartialEquality_{n,m}$ sei über *StatSet* wie folgt definiert:

(partSet,set) ∈ $\text{PartialEquality}_{n,m}$ genau dann, wenn gilt:

a) $\forall i \in \text{partSet}: n \leq i \leq m$

b) $\forall i \in [n,m]$ gilt: $i \in \text{partSet}$ g.d.,w. $i \in \text{set}$

Realisieren sie eine Prozedur `PartialEquality`, die feststellt, ob zwei Mengen in dieser Relation stehen.

```
PROCEDURE PartialEquality(partSet,set: StatSet;
                          n,m:CARDINAL):BOOLEAN;
(*es wird vorausgesetzt, dass gilt: 0 <= n <= m <= 31*)
```

Aufgabe 6:

Gegeben sei ein absteigend sortiertes Feld von CARDINAL-Zahlen. Also feld[i] ≥ feld[j] für $0 \leq i < j \leq$ HIGH(feld). Schreiben Sie folgende Prozedur:

```
PROCEDURE enthalten(feld: ARRAY OF CARDINAL; x:CARDINAL):BOOLEAN;
(*feld ist absteigend sortiert; wenn x in feld enthalten, so TRUE,
  andernfalls FALSE; zu verwenden ist ein binaeres Suchverfahren.
 *)
```

Aufgabe 7:

Eine Prozedur soll die Fläche verschiedener geometrischer Figuren berechnen:

```
PROCEDURE flaeche(figur: Figur):REAL;
```

Als geometrische Figuren sind vorzusehen:

Kreis: beschrieben durch den Radius r;

Rechteck: beschrieben durch Länge l und Breite b;

Trapez: beschrieben durch die beiden parallelen Seiten g1, g2 und deren Abstand a;

Dreieck: beschrieben durch Grundlinie g und Höhe h;

Formulieren Sie eine gemeinsame Typ-Beschreibung `Figur` für die oben beschriebenen geometrischen Figuren zur Verwendung in der Prozedur `flaeche` !

Aufgabe 8:

Verzichten Sie bei beiden Teilaufgaben auf die Verwendung von Bibliotheksfunktionen. Sie dürfen in b) jedoch die in a) definierte Prozedur benutzen!

a) Realisieren Sie folgende Prozedur:

```
PROCEDURE Length(str: ARRAY OF CHAR):CARDINAL;
(* gibt die Laenge des Strings zurueck, es gilt also:
   Length(str) = i g.d.w. i <= HIGH(str) und str[i] = 0C
                          und fuer alle 0 <= j < i gilt:
                          str[j] # 0C
                   oder   i = HIGH(str) +1 und fuer alle
                          0 <= j < i gilt: str[j] # 0C
 *)
```

b) Auf der Menge der Strings (*ARRAY OF CHAR*) sei folgende Relation *isSubstring* definiert:

substr, str seien Strings

(substr,str) $\in$ isSubstring, genau dann wenn gilt

a) Length(substr) $\leq$ Length(str) und

b) $\exists$ i, $0 \leq i \leq$ Length(str) - Length(substr) mit:
$\forall\ 0 \leq j \leq$ Length(substr) -1: substr[j] = str[i+j].

Realisieren Sie diese Relation durch folgende Prozedur:

```
PROCEDURE isSubstring(sub,str: ARRAY OF CHAR): BOOLEAN;
```