Allgemeine Informatik I

WS 1999/2000

19. Februar 2000

Name ______________________________

Vorname ______________________________

Matr.Nr. ______________________________

Studiengang ______________________________

Aufgabe		Punkte	Bewertung
1	a)	6	
	b)	4	
	c)	4	
2	a)	2	
	b)	2	
	c)	8	
3	a)	2	
	b)	2	
	c)	4	
4		6	
5	a)	4	
	b)	4	
	c)	12	
Summe		**60**	

☞ Bearbeitungszeit: 90 Min.

Viel Erfolg!

Aufgabe 1 (14 Punkte)

Die formale Sprache **L** über dem Vokabular **V = {0,1}** sei verbal wie folgt definiert:

L = { alle nichtleeren mit einer ‘0’ beginnenden Zeichenfolgen
aus V^*, bei denen die Anzahl der unmittelbar
nebeneinanderstehenden gleichen Symbole ungerade ist }

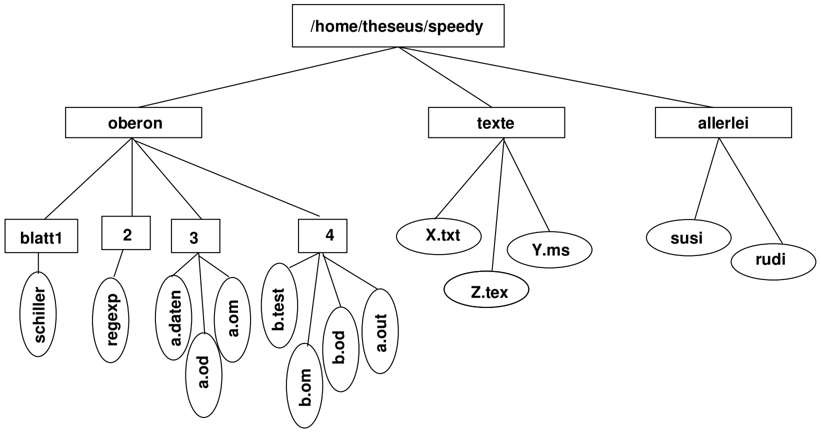
(*a*) 6 Punkte
Beschreiben Sie L mittels einer regulären Grammatik!

(*b*) 4 Punkte
Beschreiben Sie L mittels eines regulären Ausdrucks!

(*c*) 4 Punkte
Beschreiben Sie L mittels eines endlichen deterministischen Automaten!

Aufgabe 2 (12 Punkte)

Gegeben sei in grafischer Darstellung die Verzeichnisstruktur des Hörers der Vorlesung “Allgemeine Informatik I” mit dem Login-Namen *speedy* (Die Rechtecke stellen Katalogdateien, die Ovale reguläre Dateien dar):



(*a*) 2 Punkte
speedy hat sich angemeldet und befindet sich in seinem Heimatkatalog. Er hat für seine Übungsaufgaben im Unterkatalog *oberon* für jedes Blatt einen eigenen Unterkatalog angelegt. Als Name des Unterkatalogs für das erste Blatt hat er *blatt1* gewählt, als Name des Unterkatalogs für das zweite, dritte und vierte Blatt hat er nur die Blattnummer gewählt. Er möchte nun den Katalog *blatt1* in *1* umbenennen. Geben Sie das notwendige Kommando an (Sie stehen immer noch im Heimatkatatlog und dürfen diesen nicht verlassen!):

`theseus$` ________________________________

(*b*) 2 Punkte
Er möchte nun die Datei *a.out* löschen. Geben Sie ihm die notwendigen Kommandos (Sie dürfen es machen, wie Sie wollen) - die Zahl der Kommandozeilen, die im Folgenden angegeben sind, hat nichts mit der Zahl der notwendigen Kommandos zu tun!

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

(*c*) 8 Punkte
Das im Rahmen der 4. Übungsaufgabe erstellte Programm ist für ihn sehr nützlich; daher legt es sich in seinem Heimatkatalog einen Unterkatalog namens *src* und einen weiteren Unterkatalog *bin* an. Die Quellprogramme werden in den Katalog *src* kopiert, dort nocheinmal übersetzt und das ausführbare Programm kommt in den Katalog *bin*. Alles, was nach der Übersetzung in *src* nicht mehr benötigt wird, soll wieder entfernt werden. Geben Sie die notwendigen Kommandos an (Sie können dabei beliebig die Kataloge wechseln)! Die Zahl der Kommandozeilen, die im Folgenden angegeben sind, hat wieder nichts mit der Zahl der notwendigen Kommandos zu tun!

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

`theseus$` ________________________________

Aufgabe 3 (8 Punkte)

(*a*) 2 Punkte
Wie lautet die Oktaldarstellung der Dualzahl 100100111011?

Lösung: __________

(*b*) 2 Punkte
Geben Sie obige Dualzahl als Hexadezimalzahl an!

Lösung: __________

(*c*) 4 Punkte
Gegeben ist folgendes Programm:

```
MODULE PrintZahl;
   IMPORT Write, ASCII;
   TYPE
      Hex = ARRAY 9 OF CHAR;
   VAR
      hexNumber: Hex;
   PROCEDURE PrintDec(z: ARRAY OF CHAR): BOOLEAN;
      VAR i,d,dec: INTEGER;
   BEGIN
      dec := 0;
      i := 0;
      WHILE (i < SHORT(LEN(z))) & (z[i] # ASCII.nul) DO
        CASE z[i] OF
        | "0" .. "9" : d := ORD(z[i])-ORD("0");
        | "A"        : d := 10;
        | "B"        : d := 11;
        | "C"        : d := 12;
        | "D"        : d := 13;
        | "E"        : d := 14;
        | "F"        : d := 15;
        ELSE           RETURN FALSE;
        END;
        dec := dec * 10 + d;
        INC(i);
      END;
      Write.Int(dec,0);
      RETURN TRUE
   END PrintDec;
BEGIN
   hexNumber := "1F";
   IF ~ PrintDec(hexNumber) THEN
      Write.String("ERROR!");
   END;
   Write.Ln
END PrintZahl.
```

Dieses Programm enthält einen gravierenden Fehler: statt **31** liefert es **25** - bei der Korrektur dieses Fehlers sollten Sie das Programm gleichzeitig so modifizieren, daß die Groß-/Kleinschreibung der Hex-Ziffern “A”-“F” erlaubt ist!

Aufgabe 4 (6 Punkte)

Gegeben ist folgendes Programm:

```
MODULE Obscure;
  IMPORT Write;
  VAR
     x,y,z: INTEGER;
  PROCEDURE Obscure(VAR u,v: INTEGER);
     VAR w: INTEGER;
  BEGIN
     w := u;
     u := v;
     v := w;
  END Obscure;
BEGIN
   x := 9; y := 1; z := 6;
   Write.String("  x  |  y  |  z"); Write.Ln;
   Write.String("_____|_____|___"); Write.Ln;

   Write.Int(x,3); Write.String("  |");
   Write.Int(y,3); Write.String("  |");
   Write.Int(z,3); Write.Ln;

   IF x > y THEN
     Obscure(x,y);
        Write.Int(x,3); Write.String("  |");
        Write.Int(y,3); Write.String("  |");
        Write.Int(z,3); Write.Ln;
   END;
   IF x > z THEN
     Obscure(x,z);
        Write.Int(x,3); Write.String("  |");
        Write.Int(y,3); Write.String("  |");
        Write.Int(z,3); Write.Ln;
   END;
   IF y > z THEN
     Obscure(y,z);
        Write.Int(x,3); Write.String("  |");
        Write.Int(y,3); Write.String("  |");
        Write.Int(z,3); Write.Ln;
   END;
END Obscure.
```

Tragen Sie in folgender Tabelle, die letztlich auch vom Programm erzeugt würde, die vom Programm erzeugten Ausgaben für die Werte der Variablen x,y,z ein!

x	**y**	**z**

Aufgabe 5 (20 Punkte)

Sie sollen ein Oberon-Programm schreiben, das von der Standardeingabe einen Text liest und darin die Häufigkeiten der verschiedenen Kleinbuchstaben ermittelt.

(*a*) 4 Punkte
Die Buchstaben "a" .. "z" sollen zunächst auf das Intervall 0 .. 25 abgebildet werden, also "a"→0, "b"→1, usw.
Enthält die Variable *ch* (vom Typ *CHAR*) einen Kleinbuchstaben, so muss dazu die entsprechende Zahl *i* ermittelt werden.
Geben Sie eine entsprechende Oberon-Anweisung an:

`i := ________________________________;`

(*b*) 4 Punkte
Damit können die Häufigkeiten der Kleinbuchstaben wie folgt gezählt werden: in einem Integer-Array mit 26 Elementen ist der Index die dem Kleinbuchstaben (in der Variablen *ch*) nach Teilaufgabe a) zugeordnete Zahl, der Wert ist die festgestellte Häufigkeit.
Für die Ausgabe muß aus dem Index aber wieder der zughörige Kleinbuchstabe (als *CHAR*) ermittelt werden; sei also *i* eine Zahl zwischen 0 und 25; geben Sie an, wie daraus der Buchstabe (in der Variablen *ch*) bestimmt werden kann:

`ch := ________________________________;`

(*c*) 12 Punkte
Das zu erstellende Programm liest also den Text von der Standardeingabe, stellt für jedes Zeichen fest, ob es ein Kleinbuchstabe ist oder nicht; wenn ja, so wird der Zähler für diesen Buchstaben um 1 erhöht. Die Ausgabe soll eine Tabelle der Form <buchstabe> : <haeufigkeit> sein.
Der Programmrahmen sei wie folgt vorgegeben:

```
MODULE Count;
  IMPORT Write, Read, Streams;
  CONST
    Number = 26;
  TYPE
    CountField = ARRAY Number OF INTEGER;
  VAR
    ch: CHAR;
    i: INTEGER;
    countField: CountField;
BEGIN
  (* Initialisierung: *)
  i := 0;
  WHILE i < Number DO
    countField[i] := 0;
    INC(i);
  END;
(* ab hier zu ``fuellen'' -> naechstes Blatt*)
```



```
END Count.
```