Vordiplom Wirtschaftswissenschaften
Allgemeine Informatik I
WS 2002/2003 – 27.2.2003
Beispiellösung – **Gruppe B**

Bearbeitungszeit: 120 Minuten

Name:	
Vorname:	
Matrikelnummer:	

Aufgabe	Punkte	Bewertung
1	**14**	
a)	2	
b)	2	
c)	2	
d)	2	
e)	3	
f)	3	
2	**16**	
a)	4	
b)	4	
c)	4	
3	**12**	
a)	4	
b)	4	
c)	4	
4	**6**	
5	**12**	
6	**8**	
7	**6**	
8	**16**	
a)	8	
b)	8	
Summe:	**90**	
	Note:	

Bitte benutzen Sie für die Lösungen den freigelassenen Platz nach der jeweiligen Angabe; sollte dieser nicht ausreichen, verwenden Sie bitte die Rückseite, wobei die Zuordnung zur jeweiligen Aufgabe deutlich erkennbar sein muss!
Viel Erfolg!!!

Aufgabe 1 **(14 Punkte)**

Sie möchten in Ihrem Heimatverzeichnis Ordnung schaffen.

(a) **2 Punkte**

Dazu legen Sie zunächst den Katalog **allgInfo** und in diesem den Unterkatalog **aufg1** an. Geben Sie die beiden Unix-Befehlszeilen an, mit denen Sie dies bewerkstelligen – Sie dürfen den Heimatkatalog dabei nicht verlassen!
Lösung:

```
thales$  mkdir allgInfo


thales$  mkdir allgInfo/aufg1
```

(b) **2 Punkte**

Sie befinden sich nach wie vor in Ihrem Heimatverzeichnis. Verschieben Sie nun alle Dateien, deren Name mit **a1** beginnt (also z. B. *blatt1.om*), in den oben angelegten Unterkatalog **aufg1**, ohne den Heimatkatalog zu verlassen!
Lösung:

```
thales$ mv a1* allgInfo/aufg1
```

(c) **2 Punkte**

Sie haben in Ihrem Heimatverzeichnis – noch immer Ihr Arbeitskatalog – jede Menge Dateien, deren Namen nur aus einem Zeichen (Buchstabe oder Ziffer) besteht; diese enthalten nur "Schrott" und Sie wollen diese daher alle löschen! Geben Sie die dazu notwendige Unix-Befehlszeile an!
Lösung:

```
thales$ rm ?
```

(d) **2 Punkte**

Weiterhin entdecken Sie eine Datei namens **vwl.ps**, die etwas mehr als 10MB groß ist. Da Sie diese Datei weiterhin benötigen, Sie aber kooperativ sparsam mit dem Plattenplatz umgehen wollen, beschließen Sie diese Datei zu komprimieren. Geben Sie exakt an, was Sie dazu machen!
Lösung:

```
thales$ gzip vwl.ps
```

(e) **3 Punkte**

In Ihrem Heimatkatalog hatten Sie vor langer Zeit einen Unterkatalog **temp** angelegt, in dem sich ebenfalls viel „Schrott" angesammelt hat, teilweise auch in Unterkatalogen von **temp**. Sie wechseln also in den Katalog **temp** – dies ist anzugeben – und löschen mit einem einzigen Befehl den gesamten Inhalt von **temp**! Sie können davon ausgehen, dass Dateinamen nur aus Buchstaben und Ziffern bestehen!
Lösung:

Katalog wechseln:
```
thales$ cd tmp
```

Löschen:
```
thales$ rm -r *
```

(f) **3 Punkte**

Sehr viele Unix-Kommandos sind als **Filter** konstruiert. Was charakterisiert diese? Nennen Sie ein gängiges UNIX-Kommando, auf das dieser Begriff zutrifft!
Lösung:
Sie lesen von der Standardeingabe, schreiben an die Standardausgabe, Fehler-/Diagnosemeldungen an die Diagnosausgabe
Kommandos wie wc, grep, sort, ...

Aufgabe 2 **(16 Punkte)**

Über dem Vokabular $V = \{0, 1\}$ ist folgende Sprache definiert:

> alle Zeichenfolgen über V, bei denen die Anzahl der unmittelbar nebeneinanderstehenden Einsen (**1** ein Vielfaches von 3 ist und die Anzahl der unmittelbar nebeneinanderstehenden Nullen (**0** ein Vielfaches von 2 ist

(a) **4 Punkte**

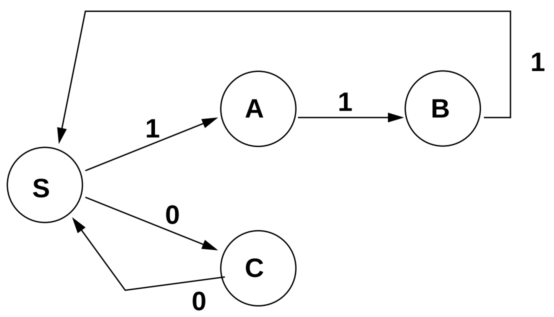
Beschreiben Sie diese Sprache durch einen regulären Ausdruck! Als Metasymbole dürfen die im Skripte im Zusammenhang mit Unix-Kommandos dargestellten Symbole verwendet werden!

Lösung: `((00)|(111))*`

(b) **4 Punkte**

Beschreiben Sie diese Sprache mit einem grafisch dargestellten endlichen, deterministischen Automaten! Start- und Endzustand (oder Endzustände) sind zu kennzeichnen!

Lösung:



Startzustand: S

Endzustand: S

(c) **4 Punkte**

Geben Sie eine reguläre Grammatik in EBNF an, die dieselbe Sprache beschreibt!

Lösung: $S \rightarrow \epsilon, S \rightarrow 1A, A \rightarrow 1B, B \rightarrow 1S, S \rightarrow 0C, C \rightarrow 0S$

Aufgabe 3 **(12 Punkte)**

In der Datei *noten* werden die Ergebnisse einer Klausur geführt. Die Einträge haben die Form *login-name***:***Punktezahl***:***note* (jeder Eintrag kommt in eine neue Zeile). Für die folgenden Aufgaben habe die Datei *noten* den folgenden Inhalt:

```
susi:90:1.0
hugo:45:4.0
hansi::
ivo:0:5.0
theo:86:1.0
gigi:3:
gerda:82:1.3
babs:78:1.7
hanna:4:5.0
ingo:74:2.0
```

(a) **4 Punkte**
Welche Ausgabe liefert folgendes Kommando?

```
cat noten | egrep '^[^:]*:[89].:'
```

Lösung:

```
susi:90:1.0
theo:86:1.0
gerda:82:1.3
```

(b) **4 Punkte**
Welche Ausgabe liefert folgendes Kommando?

```
egrep ':.:.*$' < noten
```

Lösung:

```
ivo:0:5.0
gigi:3:
hanna:4:5.0
```

(c) **4 Punkte**
Welche Ausgabe liefert folgendes Kommando? Zusätzlich ist hier zu beschreiben, **was** diese Ausgabe beschreibt!

```
egrep -c ':1\.[03]$'  noten
```

Lösung:

```
3
```

Die Anzahl derer, die mit “sehr gut” (1.0, 1.3) bestanden haben!

Aufgabe 4 **(6 Punkte)**

Schreiben Sie eine Oberon-Prozedur **ToLower**, die als Parameter eine Folge von Zeichen erhält, die darin enthaltenen Großbuchstaben in die entsprechenden Kleinbuchstaben wandelt und die sonstigen Zeichen unverändert lässt.

Beispiel für die Verwendung:

```
VAR s: ARRAY 80 OF CHAR;

s := "Ursula Engelen-Kiefer";
ToLower(s);
Write.String(s) (*liefert: ursula engelen-kiefer *)
```

Lösung:

```
PROCEDURE ToLower(VAR s: ARRAY OF CHAR);
  VAR i: INTEGER;
BEGIN
  i := 0;
  WHILE (i < LEN(s)) & (s[i] # 0X) DO
      IF (s[i] >= "A") & (s[i] <= "Z") THEN
         s[i] := CHR(ORD("a") + ORD(s[i]) - ORD("A"));
      END;
      INC(i);
  END;
END ToLower;
```

Aufgabe 5 **(12 Punkte)**

Schreiben Sie ein komplettes Oberon-Programm (nur das **.om**-File), welches von der Standardeingabe einen Text einliest und die Häufigkeiten der darin enthalten Buchstaben ermittelt. Zwischen Groß- / Kleinschreibung soll nicht unterschieden werden! In der Ausgabe sollen zeilenweise aufsteigend der Buchstabe (als Kleinbuchstabe) und dahinter seine Häufigkeit erscheinen!
Hinweis: Um die Häufigkeit der 26 Buchstaben zu zählen, sollen natürlich nicht 26 verschiedene Zählvariable verwendet werden, ein ARRAY ist hier angebracht!

Lösung:

```
MODULE Histo;
   IMPORT Read, Write, Streams;
   VAR ch: CHAR;
       h: ARRAY 26 OF INTEGER;
       i: INTEGER;
BEGIN
   i := 0;
   WHILE i < 26 DO
     h[i] := 0;
     INC(i);
   END;
   Read.Char(ch);
   WHILE ~ Streams.stdin.eof DO
      IF (ch >= "a") & (ch <= "z") THEN
          INC(h[ORD(ch)-ORD("a")]);
      END;
      IF (ch >= "A") & (ch <= "Z") THEN
          INC(h[ORD(ch)-ORD("A")]);
      END;
      Read.Char(ch);
    END;
   i := 0;
   WHILE i < 26 DO
      Write.Char(CHR(ORD("a")+i)); Write.String(": ");
      Write.Int(h[i],0); Write.Ln;
      INC(i);
   END;
END Histo.
```

Aufgabe 6 **(8 Punkte)**

Gegeben ist das folgende Programm:

```
MODULE Obscure;
  IMPORT Write;

  VAR u,v,w: CHAR;

  PROCEDURE Magic(VAR x: CHAR; y: CHAR; VAR z: CHAR);
  BEGIN
   y:=x; x:=z; z:=y
  END Magic;

BEGIN
  u :="h"; v:="u"; w:="q";
  Magic(w,v,u);
  Write.Char(u); Write.Char(v); Write.Char(w);Write.Ln;
  Magic(v,u,w);
  Write.Char(u); Write.Char(v); Write.Char(w);Write.Ln;
  Magic(u,w,v);
  Write.Char(u); Write.Char(v); Write.Char(w);Write.Ln;
  Magic(w,v,u);
  Write.Char(u); Write.Char(v); Write.Char(w);Write.Ln;
END Obscure.
```

Geben Sie an, was dieses Programm an die Standardausgabe schreibt!

Lösung:

```
quh
qhu
hqu
uqh
```

Aufgabe 7 **(6 Punkte)**

Gegeben sei folgendes Programm:

```
MODULE Logic;
  IMPORT Write;

  PROCEDURE WhichNumber(a,b: INTEGER): INTEGER;
  BEGIN
     IF (a <= 0) & (b <= 0) THEN
        IF (a <= -2) OR (b >= 2 ) THEN
            RETURN  1
        ELSE
            RETURN 2
        END
     ELSE
        RETURN 3
     END
   END WhichNumber;
BEGIN
  Write.Int(WhichNumber(1,-2),0); Write.Ln;
  Write.Int(WhichNumber(-2,-2),0); Write.Ln;
  Write.Int(WhichNumber(-1,-2),0); Write.Ln;

END Logic.
```

Geben Sie an, was dieses Programm ausgibt!
Lösung:

```
3
1
2
```

Aufgabe 8 **(16 Punkte)**

In einem Unternehmen werden monatlich Verkaufserlös und Aufwendungen erfasst; dazu hat sich ein Programmierer folgende Oberon-Datenstruktur überlegt:

```
CONST AnzahlWerte = 12;

TYPE BetrDaten = RECORD (*Betriebsdaten*)
                  monat, jahr: INTEGER;
                  erloes, aufwand: REAL;
                 END;
VAR betrDaten: ARRAY AnzahlWerte OF BetrDaten;
```

(a) **8 Punkte**

Schreiben Sie eine Oberon-Prozedur **ReadData**, die von der Standardeingabe solche Betriebsdaten in ein ARRAY (oben: **betrDaten**) einliest und die Anzahl der eingelesenen Datensätze als Rückgabewert liefert.
Beispieleingabe:

```
1 2000  900.00 1000.00
2 2000 1400.00  900.00
3 2000 2500.00 2100.00
4 2000 2300.00  900.00
```

Sie können davon ausgehen, dass die Eingabedaten genau in dieser Form korrekt und vollständig kommen!

Lösung:

```
PROCEDURE ReadData(VAR daten: ARRAY OF BetrDaten):INTEGER;
  VAR i: INTEGER;
BEGIN
  i := 0;
  Read.Int(daten[i].monat); Read.Int(daten[i].jahr);
  Read.Real(daten[i].erloes); Read.Real(daten[i].aufwand);
  WHILE (~ Streams.stdin.eof) & (i < LEN(daten)-1) DO
    INC(i);
    Read.Int(daten[i].monat); Read.Int(daten[i].jahr);
    Read.Real(daten[i].erloes); Read.Real(daten[i].aufwand);
  END;
  IF Streams.stdin.eof THEN RETURN i ELSE RETURN i + 1 END;
END ReadData;
```

(b) **8 Punkte**

Schreiben Sie eine Oberon-Prozedur, die ein ARRAY mit solchen Betriebsdaten erhält und den Index bestimmt, in dem der maximale Monats-Gewinn erzielt wurde (Gewinn := Erloes - Aufwand). Durch einen weiteren Parameter wird angegeben, wieviele Positionen von Anfang her tatsächlich belegt sind!

Lösung:

```
PROCEDURE MaxGewinn(u: ARRAY OF BetrDaten; anz: INTEGER):INTEGER;
    VAR i, maxInd: INTEGER; maxGew, g: REAL;
BEGIN
    IF (anz <= 0) OR (anz >= LEN(u)) THEN RETURN -1 END;
    maxInd := 0;
    maxGew := u[0].erloes - u[0].aufwand;
    i := 1;
    WHILE i < anz DO
      g := u[i].erloes - u[i].aufwand;
      IF g > maxGew THEN
           maxGew := g; maxInd := i;
      END;
      INC(i);
    END;
    RETURN maxInd;
 END MaxGewinn;
```