Klausur zu Allg. Informatik I

WS 98 / 99

19. Januar 1999

Name _____

Vorname _____

Matrikel-Nr. _____

Aufgabe		Punkte	Bewertung
1		10	
2	a)	4	
	b)	2	
	c)	4	
	d)	6	
3	a)	6	
	b)	6	
	c)	6	
4	a)	4	
	b)	4	
	c)	8	
	d)	12	
5	a)	6	
	b)	4	
6		18	
Summe		**100**	

Hinweis: Bitte benützen Sie den jeweils vorgesehenen Platz. Sollte dieser nicht ausreichen, verwenden Sie bitte die Rückseite und machen Sie eine eindeutige Markierung!

Aufgabe 1

Gegeben sei folgendes etwas obskures Modula-2-Programm:

```
MODULE Obscure;
     FROM InOut IMPORT WriteLn, WriteString, WriteInt;

PROCEDURE Ausgabe(s: ARRAY OF CHAR; i,x,y: INTEGER);
BEGIN
     INC(i);
     WriteString(s); WriteString(" i=");
     WriteInt(i,4); WriteString(" : x=");
     WriteInt(x,4); WriteString(" : y=");
     WriteInt(y,4); WriteLn;
END Ausgabe;

PROCEDURE Obscure1(x: INTEGER; VAR y: INTEGER): INTEGER;
BEGIN
     Ausgabe("Obscure1", i, x, y);
     IF (x < y)
     THEN y := y - x;
     ELSE
          y := y + x;
          y := Obscure2(y, x);
     END;
     Ausgabe("Obscure1", i, x, y);
     RETURN y
END Obscure1;

PROCEDURE Obscure2(VAR x,y: INTEGER):INTEGER;
     VAR z: INTEGER;
BEGIN
     z := x;
     Ausgabe("Obscure2", i, x, y);
     x := y;
     y := z;
     RETURN x-y
END Obscure2;

VAR i,x,y,z: INTEGER;

BEGIN (*main*)
     i := 0; x := 12; y := 4;
     Ausgabe("main     ", i, x, y);
     z := Obscure1(x,y);
     Ausgabe("main     ", i, x, y);
     WriteString("Ergebnis: "); WriteInt(z,4); WriteLn;
END Obscure.
```

Geben Sie die Ausgaben des Programmes (unter Zuhilfenahme der untenstehenden Tabelle) an!

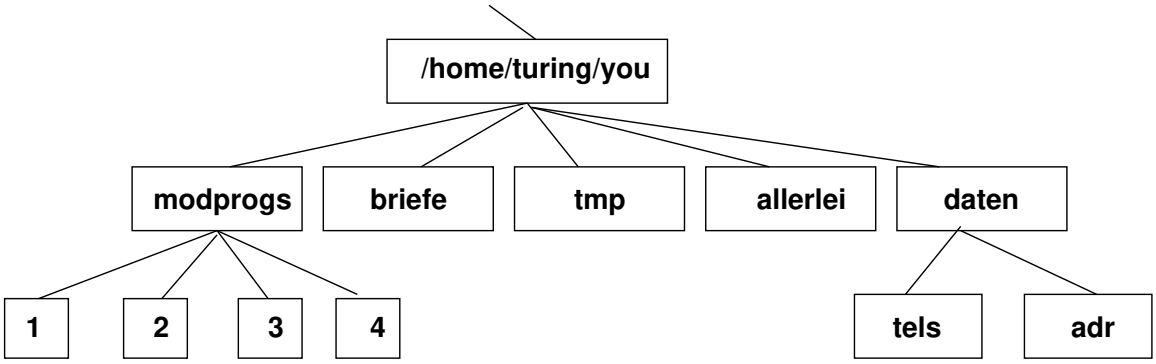
Text	(i)	(x)	(y)

• • •

Ergebnis:	

Aufgabe 2

In dieser Aufgabe seien Sie ein Benutzer unserer ‘‘turing’’ mit dem login-Namen ‘‘you’’. Das folgende Bild gibt Ihre Verzeichnisstruktur an:



(*a*) Nach dem Anmelden (ein-‘‘loggen’’) landen Sie wie immer in Ihrem Heimatverzeichnis. Geben Sie zwei vollständige Kommando’s an, die als Ausgabe den absoluten Pfadnamen Ihres Heimatkatalogs liefern!

(*b*) Sie wechseln in Ihren Katalog *modprogs* und wollen dort jetzt einen neuen Katalog namens ‘‘5’’ anlegen. Geben Sie das notwendige Kommando an!

(*c*) Sie sind immer noch im Katalog *modprogs* und sollen nun alle Modula-2-Programme (und nur diese) aus den Katalogen 1 und 3 in den Katalog 5 kopieren, ohne den Arbeitskatalog zu wechseln. Geben Sie die notwendige Kommandozeile an!

(*d*) Sie sind immer noch im Katalog *modprogs*. Sie wollen jetzt wissen, wieviele Dateien mit der Endung ‘‘.m2’’ insgesamt in den Unterkatalogen ‘‘1’’-‘‘5’’ enthalten sind. Geben Sie eine Kommandozeile an, die als Ausgabe genau diese Zahl liefert.

Aufgabe 3

Gegeben sei das Vokabular V = { 0, 1 }; darüber sei die Sprache definiert, die aus allen Folgen besteht, bei denen die Anzahl der unmittelbar nebeneinanderstehenden gleichen Symbole stets geradzahlig ist.

Zulässig: 00001100, 00, <die leere Folge>
Nicht zulässig: 001 (nur eine 1), 0 (nur eine 0), 01, 1

Geben Sie eine Definition dieser Sprache mittels

(*a*) Produktionsregeln

(*b*) eines endlichen Automaten (in grafischer Form)

(*c*) eines regulären Ausdrucks an!

Aufgabe 4

(*a*) Gegeben sei folgende Dual-Zahl: 1011110.

- Wie lautet ihre Darstellung im Oktal-System?

- Wie lautet ihre Darstellung im Hexadezimal-System?

(*b*) Gegeben sei folgende Hexadezimalzahl: A1F.

- Wie lautet ihre Darstellung im Oktal-System?

- Wie lautet ihre Darstellung im Dezimal-System?

(*c*) Die größte auf einem Rechner darstellbare CARDINAL-Zahl sei hexadezimal FFFFFFFF. Die größte 16er Potenz ist demnach 7. Schreiben Sie eine Modula-2-Prozedur, die in einem 8-elementigen Array den Dezimalwert der Potenzen 16^i für i=0,1,2,3,4,5,6,7 zurückliefert. Vorgabe:

```
TYPE Potenzen = ARRAY[0..7] OF CARDINAL;

PROCEDURE HexPotenzen(VAR pot: Potenzen);
(* pot[i] = 16 hoch i, i=0,...,7 *)
```

(*d*) Unter Zuhilfenahme der Prozedur aus c) schreiben Sie jetzt eine Prozedur, die als ersten Parameter eine CARDINAL-Zahl bekommt und über den zweiten Parameter die Hexadezimal-Zahldarstellung des ersten Parameters als String liefert. Vorgabe:

```
(* globale Variable: *)
     VAR hexDigits: ARRAY[0..15] OF CHAR;
(* initialisiert als: hexDigits[0]='0',
 * hexDigits[1]='1', ..., hexDigits[15]='F'
 *)

TYPE HexString = ARRAY[0..8] OF CHAR;

PROCEDURE Num2Hex(n: CARDINAL; VAR hexString: HexString);
(* die erste Hex-Ziffer ungleich 0 steht in hexString[0]
 * ausser es ist n = 0
 *)
```

Aufgabe 5

Bool'sche Ausdrücke:

Gegeben sei folgende Prozedur:

```
PROCEDURE Bool(x,y,z: INTEGER): INTEGER;
BEGIN
      IF  (x>y) OR (z>y) THEN

     (*_____________________________________________________________*)
         RETURN 1
      ELSE

     (*_____________________________________________________________*)
         IF  (x > 0) & NOT( ( z > y) OR (y < 0) ) THEN

     (*_____________________________________________________________*)
              RETURN 2
         ELSE

     (*_____________________________________________________________*)
              IF x<0 THEN

     (*_____________________________________________________________*)
                   RETURN 3
              ELSE

     (*_____________________________________________________________*)
                   RETURN 4
              END
         END
     END
END Bool;
```

(*a*) Geben Sie auf allen im Programm durch Linien markierten Zeilen in möglichst einfacher Form den Bool'schen Ausdruck an, der gelten muß, damit diese Stelle erreicht wird.

(*b*) Geben Sie vier konkrete Parameterbelegungen an, so daß jeder der definierten RETURN-Werte einmal entsteht.

RETURNWERT	x	y	z
1			
2			
3			
4			

Aufgabe 6

Argumentverarbeitung:

Schreiben Sie ein Modula-2-Programm, das alle seine Argumente in umgekehrter Reihenfolge (also erst das letzte, dann das vorletzte,...) jeweils auf einer Zeile ausgibt. Die maximal zulässige Länge eines Arguments sei auf 31, die maximale Anzahl der Argumente auf 20 festgelegt!