Prof. Dr. A. Schwill, Dr. Peter-Uwe Zettiér, Marco Thomas 12.11.97
Institut für Informatik

Übungen zur Vorlesung Algorithmen, Daten, Programme I Blatt 5

Abgabe: 18.11.97, 13.00 Uhr im Postfach von Herrn Zettiér bzw. Herrn Thomas in Raum 1.2.2.08

Aufgabe 1: (8 Punkte)

Sei X ein endliches Alphabet und f: $X^* \to X^*$ eine bijektive, totale, berechenbare Funktion.

Zeigen Sie: Die Umkehrfunktion f^{-1}: $X^* \to X^*$ ist berechenbar.

Aufgabe 2: (6 Punkte)

Erstellen Sie in ihrem home-Verzeichnis ein neues Verzeichnis mit dem Namen `bin` (Befehl: `mkdir bin`). Kopieren Sie sich die Datei `Aufg2` aus dem Verzeichnis `/home/SUNOS/ADP` in dieses Verzeichnis (Befehl: `cp /home/SUNOS/ADP/Aufg2 $HOME/bin/Aufg2`) und wechseln Sie in das Verzeichnis bin.

a) Geben Sie die axiomatische und denotationale Semantik des Shellskripts `Aufg2` an, indem Sie die Datei mit dem Texteditor betrachten.

b) Sie können das Skript wie ein Kommando verwenden, da die Shell das Programm während der Abarbeitung interpretiert. Leiten Sie die Ausgabe des Skripts vom Bildschirm in die Datei `Ergeb` um (Befehl: `Aufg2 > Ergeb`) und verwenden Sie als Eingabe die ersten 3 Ziffern ihrer Matrikelnummer.

c) Drucken Sie nun die Dateien Aufg2 und Ergeb aus (Befehl: `cat Aufg2 Ergeb | lpr`).

Aufgabe 3: (6 Punkte)

Schreiben Sie eine Funktion in PRO mit einer Zahlenfolge als Parameter, die die gespiegelte Version der Zahlenfolge als Ergebnis liefert, die also folgende Funktion f berechnet:

f: $\mathbb{Z}^* \to \mathbb{Z}^*$ mit

$f([x_1,x_2,...,x_n])=[x_n,x_{n-1},...,x_1]$ für $n \geq 0$.

Zeigen Sie anschließend möglichst präzise, daß Ihre Funktion tatsächlich das Gewünschte leistet.

Aufgabe 4: (3 Punkte)

Prüfen Sie die folgenden Programme auf korrekte Syntax und ermitteln Sie die Semantik der korrekten Programme, indem Sie die berechnete Funktion exakt angeben.

a)
```
def i,s: Zahl;
 i←1;
 s←0;
 solange n≥i tue
         s←s+i;
         i←i+1
 ende;
 zeige(s).
```

b)
```
def f: Zahlenfolge;
def z: Zahl;
lies(z);
lies(f);
solange z≤erstes(f) tue
        f←rest(f)
ende;
zeige(f).
```

c)
```
def f,g: Zahlenfolge;
def i,z: Zahl;
lies(z);
lies(f);
i←1;
g←[ ];
solange i≤z tue
                g←g•erstes(f);
                f←rest(f)
ende;
zeige(g).
```