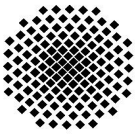
Universität Stuttgart
Fakultät 5, Institut FMI
EfidI 1, Professor Claus



WS 2002/2003
Bertele, Lewandowski
24. Oktober 2002

Aufgabenblatt 2

Abgabe bis 31.10.2002 15:00 Uhr – Besprechung in der Woche ab dem 4.11.2002

Aufgabe 1 Grenzen (Votieraufgabe, leicht) 1,5+0,5 Punkte

In dieser Aufgabe lernen Sie die Mächtigkeit einfacher Operationen einzuschätzen und sich Gedanken über den benötigten Speicherplatz zu machen.

Einmal vor langer, langer Zeit hatte sich ein Bauer um das Wohl seines Königreichs verdient gemacht. Der König sagte ihm, er habe einen Wunsch frei. „Mein Wunsch ist nicht groß“, sprach der Bauer. „Nehmen Sie ein Schachbrett und legen Sie mir auf das erste Feld ein Reiskorn, auf das zweite Feld zwei Reiskörner, auf das dritte Feld vier und so weiter – immer das Doppelte. Das ist alles was ich haben will.“
Nehmen Sie an, Sie wollen ein Ada95-Programm zur Berechnung der Anzahl der Reiskörner schreiben, die der Bauer erhält.

a) Wie viele Reiskörner erhält der Bauer? Wie groß muss der Datentyp sein, um das Ergebnis speichern zu können?
b) Was versteht man bei einer Programmiersprache unter dem Begriff „Bereichsgrenze“?

Aufgabe 2 Programmablauf in Ada95 (Votieraufgabe, mittel) 4 Punkte

Auf dem letzten Aufgabenblatt haben wir uns mit dem Ablauf von Programmen beschäftigt. Das Programm war jedoch nicht in Ada95 erstellt. In dieser Aufgabe bekommen Sie ein „richtiges“ Ada95-Programm zum Analysieren. Falls Sie nicht weiterkommen, können Sie im Referenzhandbuch von Ada die Semantik der Befehle nachschlagen.

Geben Sie ein Ablaufprotokoll für das folgende Ada95 Programm an. Welche Funktion berechnet dieses Programm? Testen Sie das Programm mit zehn Beispielwerten. Stellen Sie eine Vermutung auf, welcher Wert *N* am Ende ausgegeben wird. Erstellen Sie Ablaufprotokolle für die Werte *N=7* und *N=27*. Beschränken Sie die Ablaufprotokolle auf maximal 20 Schleifendurchläufe.

```
1. with Ada.Integer_Text_Io;
2. use  Ada.Integer_Text_Io;
3.
4. procedure Was_Macht_Es is
5.    M,N:Integer;
6. begin   M:=0;
7.    Get(N);
8.    while N > 1 loop
9.       if N mod 2 = 0 then
10.         N:=N / 2;
11.      else
12.         N:=(3*N)+1;
13.      end if;
14.      M:=M+1;
15.   end loop;
16.   Put(M);
17.   Put(N);
18. end Was_Macht_Es;
```

Aufgabe 3 Römische Zahlen (schriftlich, schwer) 8 Punkte

Im römischen Reich wurde ein anderes Zahlsystem verwendet. Die alten Römer verwendeten als Zahlzeichen I=1, V=5, X=10, L=50, C=100, D=500 und M=1000. Begonnen wird links mit dem Symbol der größten Zahl. Dabei werden die Symbole I, X, C, M höchstens drei mal hintereinander geschrieben. Die Symbole V, L, D kommen nur einzeln vor. Symbole einer kleineren Zahl, die vor dem einer größeren stehen, werden von diesem subtrahiert, wobei allerdings nur höchstens ein kleineres Symbol einem größeren vorangestellt werden darf. Folgende Regeln sind weiterhin zu beachten: Einer können nur von Fünfern und Zehnern abgezogen werden. Zehner können nur von Fünfzigern und Hundertern abgezogen werden. Hunderter können nur von Fünfhundertern und Tausendern abgezogen werden.

Geben Sie ein Verfahren an, um die Addition von 1 durchzuführen. Konstruieren Sie hieraus einen Algorithmus, der zu einer römischen Zahl (<4000) die Zahl 1 hinzuaddiert.

Aufgabe 4 Zahldarstellungen I (Votieraufgabe, leicht) 2 Punkte

Die Konvertierung von einer Darstellung in eine andere, quasi das Übersetzen von Zahlen ist eine wichtige Aufgabe eines Rechners. Man sollte jedoch die Dinge, die man programmiert, auch selbst verstanden haben.

Stellen Sie die folgenden Zahlen in den Basen 2, -2, 7 und 16 dar. Wie sieht bei Basis 2 das 2-Komplement aus?

- 19
- 33
- –123
- 394

Aufgabe 5 Zahldarstellungen II (schriftlich, mittel) 2+2 Punkte

In der vorigen Aufgabe haben wir uns mit ganzen Zahlen beschäftigt. Jetzt betrachten wir reelle Zahlen, dargestellt zur Basis 2.

a) Stellen Sie die folgenden Zahlen in Gleitkommaschreibweise mit 16 Stellen zur Basis 2 dar. (Mantissenlänge=*11*, Exponentenlänge=*5*)
   - 4
   - 2,5
   - –2,5
   - –1

b) Geben Sie die kleinste und die größte darstellbare Zahl, sowie auch die betragsmäßig kleinste und größte Zahl an.

Aufgabe 6 Rechenvorschriften (Zusatzaufgabe, recht schwer) 10 Punkte

In der Vorlesung wurde erwähnt, dass die Algorithmen zur Durchführung der mathematischen Operationen im Dualsystem noch relativ einfach sind, jedoch bezüglich der Basis –2 bereits recht ungewöhnlich erscheinen.

a) Schreiben Sie einen Algorithmus, der eine ganze Zahl (dargestellt als Folge von 0 und 1) negiert.
b) Geben Sie einen Algorithmus an, der 1 addiert.
c) Geben Sie einen Algorithmus für die Addition zweier Zahlen an. (Aber bitte nicht: *a+b = addiere b-mal 1 zu a*)

Allgemeine Hinweise:

- Es sind auf diesem Aufgabenblatt 30 Punkte erreichbar. Davon werden für den Übungsschein maximal 20 Punkte angerechnet. Die Zusatzaufgabe wird eventuell in den Übungen nicht besprochen. Die Bearbeitung ist nicht verpflichtend, dient jedoch zum eigenen Training.
- Hinter jeder Aufgabe ist die Art der Aufgabe, der Schwierigkeitsgrad, sowie die erreichbare Punktzahl angegeben.
- Die schriftlichen Aufgaben (22 Punkte) geben Sie bitte zum Abgabezeitpunkt im eClaus-System ab. Bitte votieren Sie bitte ebenfalls bis zum Abgabezeitpunkt im eClaus-System.
- Bei weiteren Fragen, wenden Sie sich bitte an das Forum (http://fachschaft.informatik.uni-stuttgart.de/forum/), Ihren Tutor, oder per Mail direkt an J. Bertele (inf@studbs.informatik.uni-stuttgart.de).
- Weitere Hinweise finden Sie auf unserer Veranstaltungswebseite unter: http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ws02-03/info_I_0203.html