Programmierkurs I (Inf., W-Info.), Übungsblatt 6

Claus/Weicker, Wintersemester 03/04 — Abgabetermin: 4.12.2003, 23:59 Uhr

Aufgabe 1: Verschlüsselung eines Zahlenpaars (leicht) — 6 Punkte

Ein Zahlenpaar $(a, b) \in \mathbb{N} \times \mathbb{N}$ lässt sich als eine einzelne Zahl verschlüsseln und darstellen, indem die Funktion

$$f(a, b) \quad = \quad 2^a 3^b$$

angewandt wird. Schreiben Sie zwei Prozeduren für die Ver- und Entschlüsselung (d.h. für f und die Umkehrfunktion f^{-1}). Und verwenden Sie diese Funktionen in einem Hauptprogramm `zahlenpaar.adb`, so dass der folgende Dialog entsteht.

```
Geben Sie das zu verschluesselnde Zahlenpaar ein: 3 7
Ergebnis: 17496
Geben Sie die zu entschluesselnde Zahl ein: 12
Ergebnis: 2 1
```

Aufgabe 2: Binomialkoeffizient (mittel) — 6(+4) Punkte

In der Mathematik ist der Binomialkoeffizient wie folgt definiert, wobei das Ausrufezeichen für die Fakultätsfunktion steht:

$$\binom{n}{k} = \begin{cases} \frac{n!}{k! \cdot (n-k)!}, & \text{für } 0 \leq k \leq n \\ 0, & \text{für } 0 \leq n < k \end{cases}$$

Der Binomialkoeffizient bezeichnet die Anzahl Der Möglichkeiten aus n verschiedenen Kugeln genau k Kugeln auszuwählen, wobei keine Kugel mehrmals ausgewählt wird. So bezeichnet etwa $\binom{49}{6} = 13983816$ die Anzahl der möglichen Ergebnisse beim Lotto.

a) Schreiben Sie ein Programm `binomial.adb`, welches zwei natürliche Zahlen n und k vom Benutzer einliest und die Zahl $\binom{n}{k}$ ausgibt. Benutzen Sie hierbei ausschließlich den Datentyp `Natural`.

```
Geben Sie n ein: 4
Geben Sie k ein: 3
Ergebnis: 4
```

b) **Zusatzaufgabe:** Sie werden feststellen, dass $\binom{n}{k}$ sehr schnell recht groß werden kann. Der Datentyp `Natural` kann in den vorliegenden Versionen von Ada nur Werte zwischen 0 und $2^{31}-1$ annehmen. Letzteren Wert erhalten Sie in Ada mit dem Ausdruck `Natural'Last`. Eine direkte Berechnung der obigen Formel ist daher nicht der beste Weg, um Binomialkoeffizienten zu erhalten, weil dabei Zwischenergebnisse entstehen, die größer als das Endergebnis sind. Für manche Werte von n und k schlägt dann die Berechnung fehl, weil die Zwischenergebnisse zu groß für den Datentyp `Natural` sind, obgleich sich das Endergebnis noch damit darstellen ließe.

Implementieren Sie eine bessere Berechnungsmethode für Binomialkoeffizienten, mit der man möglichst große Ergebnisse berechnen kann (z.B. durch frühzeitiges Kürzen). Als Erfolgskriterium sei festgelegt, dass das Ergebnis von $\binom{49}{6} = 13983816$ korrekt berechnet wird.

Aufgabe 3: Programmanalyse (schwer) — 9 Punkte

Der Student Exit hat bei unserer Tutorin Goto das folgende unkommentierte Programm abgegeben.

```
with text_io, ada.Integer_Text_Io;
use text_io, ada.Integer_Text_Io;
```

```
procedure Exit_Considered_Harmful is
   Zahl1, Zahl2, Zahl3 : Integer := 0;
begin
   put("Erste Zahl: ");
   get(Zahl1);
   Put("Zweite Zahl: ");
   Get(Zahl2);
   for I in 1..Zahl1 loop
      Zahl2 := Zahl2 + 3;
      exit when Zahl2 > 100;
      for J in Zahl1..Zahl1+10 loop
         exit when Zahl1 > Zahl2;
         Zahl2 := Zahl2 - Zahl1;
         if I+15 > J then
            Zahl3 := Zahl3 + 1;
            exit when Zahl3 > Zahl1;
         end if;
         Zahl2 := Zahl2 + Zahl1 + 1;
      end loop;
   end loop;
   Put("Ergebnis: ");
   Put(Zahl1-Zahl3);
   Put(Zahl2);
end Exit_Considered_Harmful;
```

Beide stehen vor einem Rätsel, was das Programm tatsächlich tut. Da sich unter Ihnen Spezialisten für die Transformation von bestehenden Programmen und die Reformulierung von Schleifen befinden, transformieren Sie das Programm in eine Version `ohneexit.adb`, die keine `exit`-Anweisungen mehr enthält und sich identisch zum Ursprungsprogramm verhält. Vereinfachen Sie das Programm und versehen Sie es mit sinnvollen Kommentaren, sodass die Verständlichkeit erhöht wird. Sie können davon ausgehen, dass die Eingaben für das Programm immer zwischen 0 und 200 liegen. Das obige Programm ist auf der Webseite des Programmierkurses erhältlich.

Hinweis 1: Die Überführung in ein Programm ohne `exit` ist relativ einfach. Um das Programm jedoch in eine leicht lesbar, klar verständliche Form zu bringen, ist vermutlich ein wesentlich größerer Aufwand notwendig. Wägen Sie ab, wieviel Zeit Sie investieren möchten.

Hinweis 2: Falls Sie die Anweisung `exit when` nicht kennen: Es handelt sich dabei um ein verstecktes `goto`, welches die aktuelle (innerste) Schleife sofort verlässt, falls die Bedingung hinter `when` erfüllt ist. Die Programmausführung wird direkt hinter der Schleife fortgesetzt.

Hinweise

- Zur Abgabe wird das eClaus-System verwendet:
  `http://eclaus.informatik.uni-stuttgart.de`
- Die Abgabe zu jeder Teilaufgabe besteht aus einem kompilierbaren Ada-Quelldatei. In jeder Aufgabe wird ein Dateiname vorgegeben. Verwenden Sie diesen bitte für das Hauptprogramm. Auch sind in der Aufgabe Angaben zu Ein- und Ausgabetexten sowie zur Formatierung der Ausgabe enthalten. Bitte folgen Sie diesen Angaben so genau wie möglich.
- Beachten Sie beim Programmieren bitte die folgenden Hinweise („kleine Programmierrichtlinie“).
  - Halten Sie einzelne Bestandteile überschaubar, z.B. indem Sie nur eine Anweisung pro Zeile schreiben, sowie pro Zeile höchstens 80 Zeichen, höchstens 40 Zeilen pro Prozedur/Funktion, nicht mehr als 800 Zeilen pro Datei und höchstens 5 Parameter bei Prozeduren/Funktionen benutzen.

  - Bezeichner sollen selbsterklärend und maximal 15 Zeichen lang sein. Bezeichner enthalten nur Buchstaben, den Bindestrich oder den Unterstrich.
  - Durch Einrückung um 2 Zeichen soll die logische Gliederung eines Programms verdeutlicht werden. Beispielsweise wird der Anweisungsteil einer IF-Verzweigung eingerückt, während „`end if;`“ nicht mehr eingerückt wird. Auch auf der nächsten Zeile fortgesetzte Zeilen werden um 2 Zeichen eingerückt.
  - Zu Beginn des Programms muss in Kommentaren das Konzept und die Lösungsidee des Programms ausführlich erläutert werden.
  - Auch im Programmtext sind Kommentare einzufügen, um den Code zu erläutern.

- Beachten Sie, dass jede Abgabe individuell vom jeweiligen Studierenden erstellt werden muss. Werden von den Tutoren Plagiate erkannt, d.h. exakte oder leicht modifizierte Kopien, werden für die Aufgabe keine Punkte vergeben. Falls Sie die Lösung der Aufgaben vor der Abgabe in Gruppen besprechen, achten Sie darauf, dort nur das generelle Konzept abzuklären und die Programmierung jedem selbst zu überlassen.

- Bei Fragen wenden Sie sich bitte an Ihren Tutor oder an die Übungsleitung:
  `Karsten.Weicker@fmi.uni-stuttgart.de` oder Tel. 7816-337

- Weitere Veranstaltungshinweise einschließlich der Übungsblätter finden Sie unter:
  `http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ws03-04/ada95/`