Programmierkurs I (Inf., W-Info.), Zusatzblatt (Blatt 14)

Claus/Weicker, Wintersemester 03/04 Abgabetermin: 31.03.2004, 23:59 Uhr

Da Studierende der Wirtschaftsinformatik bedingt durch Prüfungen am Ende der Vorlesungszeit nicht mehr alle Aufgaben bearbeiten konnten, haben sie hier die Möglichkeit die Punkte der letzten 3 Blätter auf 60 Punkte aufzufüllen.

Aufgabe 1: Nachkommen (leicht) — 5 Punkte

In der Familie Recursi bekommt jedes Familienmitglied im Laufe seines Lebens zwei Kinder, und zwar stets das erste im Alter von 19 und das zweite im Alter von 23 Jahren. Implementieren Sie die Funktion `Familiengroesse`, die das Alter des Stammvaters Adamo Recursi als Parameterwert übergeben bekommt und davon ausgehend die Anzahl der Familienmitglieder berechnet. Gibt man beispielsweise das Alter 42 an, ist die Familiengröße 6. Die Spezifikationsdatei `recursi.ads` ist auf der Web-Seite erhältlich. Geben Sie die Datei `recursi.adb` ab.

Aufgabe 2: Stack II (mittel) — 9 Punkte

Schreiben Sie ein generisches Paket, welches einen Datentyp `Stack_Typ` für einen Stack (Keller, LIFO-Liste) implementiert. Die Elemente im Stack sollen dabei durch den generischen Parameter `Basistyp` angegeben werden. Die Funktionalität des Pakets (einschließlich der Fehlerbehandlung) ist für einen Spezialfall in der Aufgabe 2 von Blatt 12 beschrieben. Schreiben Sie sowohl `stack.ads` als auch `stack.adb` und geben Sie beide Dateien ab. Beachten Sie, dass Ihre Implementation mit der zur Verfügung gestellten Datei `teststack.adb` übersetzt.

Aufgabe 3: Turing-Maschine (mittel) — 18 Punkte

Implementieren Sie eine einfache Version einer Turing-Maschine. Sie soll aus einem internen Zustand, der Übergangsrelation sowie dem Bandinhalt bestehen. Der Bandinhalt rechts bzw. links des Lesekopfs kann dabei durch jeweils einen Stack (siehe vorherige Aufgabe) dargestellt werden. Für die Übergangsrelation gehen wir vereinfachend von einem Alphabet $\{0, 1\}$ sowie maximal 20 unterschiedlichen Zuständen aus, so dass die Übergangsrelation in einem fest definierten Array dargestellt werden kann. Folgende Funktionen und Prozeduren sind zu implementieren:

- Initialisieren des Turing-Maschine (2 Punkte)
- Initialisieren des Bands (3 Punkte)
- Ausgabe des Bandinhalts (3 Punkte)
- Definieren eines Übergangs für die Übergangsrelation (3 Punkte)
- Einen oder mehrere Schritt(e) durchführen (7 Punkte)

Geben Sie Ihre Implementation in der Datei `turing.adb` ab. Die Spezifikation befindet sich in der Datei `turing.ads` auf dem Web-Server. Ergänzen Sie in der Spezifikationsdatei im privaten Teil die Typdeklaration.
Hinweis: Beachten Sie bei Ihrer Implementation, dass die Turing-Maschine auch Teile des Bandes außerhalb des Eingabestrings benutzen kann. D.h. falls Sie den Lesekopf bewegen und entsprechend ein Element von einem der beiden Stacks nehmen möchten, kann es sein, dass dieser Stack leer ist. Dann müssen Sie eine weitere leere Zelle hinzufügen.

Aufgabe 4: Stammbaum II (schwer) 28 Punkte

In Aufgabe 2 von Blatt 10 wurde ein Programm zur Verwaltung von Stammbauminformationen geschrieben, welches in dieser Aufgabe neu implementiert werden soll. Die Daten sollen dabei objektorientiert als Graph organisiert werden. Jedes Objekt (vom privaten Typ `Person`) entspricht einem Familienmitglied und jede als Zeiger realisierte Kante einer direkten Verwandschaftsbeziehung. Beachten Sie auch, dass das Sie eventuell eine zusätzliche Struktur für das Auffinden jedes einzelnen Knotens benötigen. Erforderliche Funktionen und Prozeduren:

- Datenimport, um die Daten aus einer Datei einzulesen – dabei gehen wir von einem einfacheren Datenformat aus: jedes Familienmitglied wird durch 4 Zeilen dargestellt: eindeutige Nummer in Zeile 1, Name in Zeile 2, Nummer der Mutter in Zeile 3 und Nummer des Vaters in Zeile 4. Es können auch leere Zeilen auftreten. (3 Punkte)
- Datenexport (3 Punkte)
- Hinzufügen eines Familienmitglieds (2 Punkte)
- Identifikation eines Knotens zu einem Schlüssel (4 Punkte)
- Ausgabe eines Schlüssels zu einem Knoten (1 Punkt)
- Ausgabe eines Namens zu einem Knoten (1 Punkt)
- Identifikation des Mutterknotens zu einem Knoten (1 Punkt)
- Identifikation des Vaterknotens zu einem Knoten (1 Punkt)
- Anzahl der Kinder zu einem Knoten (1 Punkt)
- Ausgabe eines Kindknotens zu einem Knoten (1 Punkt)
- Berechnung, ob ein gemeinsamer Vorfahr existiert (5 Punkte)
- Berechnung, ob ein gemeinsamer Nachkomme existiert (5 Punkte)

Insbesondere sollen in der Neuimplementation auch beliebig große Datensätze unterstützt werden. Ebenso ist die Anzahl der Kinder nicht beschränkt – auch hier ist eine Lösung, z.B. in der Form einer verketteten Liste der Kinder für eine Person, zu finden.
Beachten Sie die Spezifikation in der Datei `stammbaum2.ads` auf dem Web-Server. Geben Sie die Implementation in der Datei `stammbaum2.adb` ab. Ergänzen Sie in der Spezifikationsdatei im privaten Teil die Typdeklaration.

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/`