Ada-Kompaktkurs (Wirtschaftsinformatiker), Übungsblatt 4
Claus/N.Weicker, WS 02/03

Bitte beachten Sie die allgemeinen Hinweise am Ende des Blattes!

Aufgabe 1: Palindrom (leicht) 2 Punkte

Ein Palindrom ist ein Wort oder ein Satz, welches bzw. welcher vorwärts und rückwärts gelesen identisch ist, z.B. `neben` oder `stets`. Schreiben Sie ein Programm, welches eine vom Benutzer eingegebene Zeichenkette (beliebiger Länge) darauf überprüft, ob sie diese Eigenschaft hat. (Zum Einlesen von Zeichenketten siehe Aufgabe 3.3.)

Aufgabe 2: Sortieren (mittel) 5 Punkte

Schreiben Sie ein Programm, welches den Benutzer eine Sequenz von Zahlen eingeben lässt und diese dann in sortierter Reihenfolge wieder ausgibt, wobei die Ausgabe mit der kleinsten eingegebenen Zahl anfangen und mit der größten enden soll.
Hinweis 1: Um die Sache einfacher zu machen, kann Ihr Programm am Anfang zunächst abfragen, wie viele Zahlen die Sequenz hat, damit Sie ein Feld mit entsprechender Größe anlegen können, in das Sie die Zahlen anschließend einlesen.

Hinweis 2: Auf welche Weise Sie die Zahlen sortieren, ist Ihnen überlassen. Wenn Ihnen nichts Besseres einfällt, benutzen Sie folgende Methode: Suchen Sie zunächst das Minimum unter den eingegebenen Zahlen und geben es aus, dann das Minimum unter den restlichen Zahlen usw., bis Sie alle Zahlen haben.

Aufgabe 3: Matrixmultiplikation (mittel) 5 Punkte

Im folgenden bezeichnen wir eine Matrix als $m \times n$-Matrix, wenn sie m Zeilen und n Spalten hat. Das Produkt einer $m \times n$-Matrix A mit einer $n \times k$-Matrix B ist bekanntermaßen eine $m \times k$-Matrix C mit

$$c_{ij} = \sum_{p=1}^{n} a_{ip} b_{pj},$$

wobei c_{ij} den Eintrag von C bezeichnet, der in der i-ten Zeile und j-ten Spalte steht.
Schreiben Sie ein Programm, welches eine Matrixmultiplikation zweier gegebener Matrizen A und B durchführt, deren Einträge reelle Zahlen sind. Das Programm soll zuerst die Werte von m, n und k (in dieser Reihenfolge) abfragen, danach die Einträge der Matrizen A und dann B, und zwar jeweils Zeile für Zeile von links nach rechts. C, die Ergebnismatrix, soll als Tabelle am Bildschirm ausgegeben werden.
Auf der Webseite des Programmierkurses (die URL steht in den Hinweisen am Ende des Blattes) finden Sie bei den Übungsblättern einige Dateien mit Beispielmatrizen, an denen Sie Ihr Programm testen sollten. Wenn Ihr Programm die Daten in der oben angegebenen Reihenfolge einliest, können Sie diese Dateien direkt als Eingabe für Ihr Programm benutzen, und zwar wie folgt:

- Unter Linux geben Sie auf der Kommandozeile z.B. Folgendes ein:

  `meinprogramm < matrix1`

  Dabei steht `meinprogramm` für den Befehl, mit dem Sie ihr Programm normalerweise starten, und `matrix1` für eine der Dateien mit den Beispieldaten.

- Unter Windows, sofern Sie die Ada-Entwicklungsumgebung (AdaGIDE) benutzen, drücken Sie die Taste F5. In der dann erscheinenden Dialogbox tragen Sie im Feld 'Input file' den Namen der Datei ein (etwa `matrix1`) und setzen ein Häkchen in das Feld 'Redirect input'. Danach starten Sie das Programm wie gewohnt mit F4.

Aufgabe 4: Galgenraten (mittel bis schwer) 8 Punkte

Bei 'Galgenraten' handelt es sich um ein Spiel, welches wie folgt abläuft: Ein Spieler (der Fragesteller) denkt sich ein Wort aus und verrät dem anderen Spieler (dem Rater) die Länge des Wortes. Der Rater muss nun dieses Wort raten, indem er dem Fragesteller einzelne Buchstaben nennt. Nennt er einen Buchstaben, der im Wort vorkommt, verrät ihm der Fragesteller, an welchen Stellen im Wort der Buchstabe vorkommt. Nennt der Rater einen Buchstaben, der nicht im Wort vorkommt, so zählt dies als Fehlversuch. Der Fragesteller gewinnt das Spiel, sobald der Rater 10 Fehlversuche hatte. Der Rater gewinnt das Spiel, sobald er alle Buchstaben erraten hat, falls er dies mit weniger als 10 Fehlversuchen schafft.
Implementieren Sie dieses Spiel, wobei der Rechner die Rolle des Fragestellers übernimmt (das Wort geben Sie am Anfang des Spiels aber selber ein). Das Wort soll nur aus den Großbuchstaben `A` bis `Z` bestehen. Am Anfang soll die Länge des Worts mit einer entsprechende Zahl von Strichen (`-`) angezeigt werden. Nach jedem erratenen Buchstaben soll die Ausgabe erneuert werden, wobei die bereits erratenen Buchstaben dort angezeigt werden, wo sie vorkommen. Beispiel: Wenn das Wort `PROGRAMMIERKURS` ist, erfolgt am Anfang die Ausgabe

```
----------
```

Wird das `M` geraten, soll

```
------MM-------
```

ausgegeben werden; wird danach das `R` geraten, dann

```
-R--R-MM--R--R-
```

Hinweis: Seinen Namen hat das Spiel daher, dass die Anzahl der Fehlversuche üblicherweise in einer bestimmten graphischen Form dargestellt wird. Sie können sich darauf beschränken, die Anzahl der Fehlversuche einfach als Zahl auszugeben.

Aufgabe 5: Galgenraten II (Zusatzaufgabe, mittel) 5 Punkte

Programmieren Sie eine Erweiterung des Spiels, bei der Sie das zu erratende Wort nicht mehr selbst eingeben müssen. Stattdessen soll der Rechner Sie nach dem Namen einer Datei fragen, in der Wörter stehen (jedes in einer eigenen Zeile); von diesen soll er eines zufällig aussuchen. Eine solche Datei, die Sie zum Probieren verwenden können, finden Sie auf der Webseite des Programmierkurses bei den Übungsblättern.
Hinweis 1: Wenn Sie das Paket `Text_IO` benutzen, können Sie mit `datei:File_Type;` eine 'Dateivariable' anlegen und dann mit `Open(datei,In_File,dateiname);` eine Datei öffnen (wobei `dateiname` eine Zeichenkette mit dem Namen der Datei sein sollte). Mit `u:=Get_Line(datei);` können Sie jetzt eine Zeile der Datei nach `u` einlesen, falls `u` ein `Unbounded_String` ist (siehe Aufgabe 3.3). Der boolesche Ausdruck `End_of_File(datei)` gibt an, ob das Ende der Datei erreicht ist. Mit `Close(datei);` können Sie das Einlesen der Datei beenden und diese 'schließen'.

Hinweis 2: Für (Pseudo-)Zufallszahlen gibt es das Paket `Ada.Numerics.Float_Random`. Wenn Sie dieses benutzen und eine Variable `seed:Generator` deklarieren, liefert der Ausdruck `random(seed)` eine zufällige reelle Zahl im Bereich von 0 bis 1.

Hinweise

- Programme sind in Ada95 zu schreiben und müssen fehlerfrei übersetzbar sein!
- Schreiben Sie Ihre Programme so, dass ihre Wirkungsweise möglichst leicht nachvollzogen werden kann (insbesondere vom Korrektor). Dazu gehören:
  - die Verwendung sinnvoller Variablennamen (oder zumindest die Erklärung, was der Zweck jeder Variablen ist, falls dies aus dem Namen nicht hervorgeht);

  - Kommentare dort, wo der Zweck von Anweisungen nicht unmittelbar offensichtlich ist, insbesondere, wenn mathematische Überlegungen eine Rolle spielen;
  - ein übersichtlicher Programmierstil, beispielsweise sollten die Inhalte von Schleifen oder die Zweige von if-Anweisungen eingerückt werden, damit die Struktur des Programms offenbar wird.

- Pro Aufgabenblatt werden maximal 20 Punkte auf den Übungsschein angerechnet.

- Falls Sie Fragen irgendwelcher Art haben, sei es zu den Aufgaben, zum Ablauf, oder falls Sie mit dem Stoff Probleme haben, wenden Sie sich bitte an Ihren Tutor oder an die Übungsleitung: `Nicole.Weicker@informatik.uni-stuttgart.de`, Raum 1.101, oder Tel. 7816-412