Übungen zum Programmierkurs I

Abgabe bis zum Freitag, 08.11.2002, 15:00 Uhr

Bitte beachten Sie die allgemeinen Hinweise am Ende des Blattes!

Aufgabe 2.1 Caesarcode (leicht) 4 Punkte

Sie haben in den Übungen zur Einführungsvorlesung den Caesar-Code kennengelernt. Dieser dient zur Verschlüsselung von Zeichenketten. Er lässt sich wie folgt beschreiben:

> Als Parameter sei eine natürliche Zahl k vorgegeben ($0 < k < 26$). Eine Zeichenkette wird kodiert, indem jeder Buchstabe durch seinen k-ten Nachfolger im Alphabet ersetzt wird. Als Nachfolger von `Z` sei `A` anzusehen.

Schreiben Sie ein Programm, welches den Parameter k und eine Zeichenkette als Eingabe nimmt und letztere entsprechend der oberen Vorschrift kodiert ausgibt. Zeichen, die keine Buchstaben sind, sollen nicht verändert werden. 'Verschlüsseln' Sie den folgenden Text mit $k = 16$: `Kvok skmdk ocd.`

Hinweis 1: Zur Codierung müssen Sie mit den Ordnungszahlen der Zeichen im ASCII-Code rechnen. Diese sind auf Folie 81 der Einführungsvorlesung zu finden. Zu einem gegebenen Zeichen `c` finden Sie mit `character'pos(c)` seinen ASCII-Code, und mit `character'val(x)` finden Sie zu einer natürlichen Zahl `x` dasjenige Zeichen, welches den ASCII-Code `x` hat.

Hinweis 2: Die Länge der eingegebenen Zeichenkette ist nicht beschränkt; der Benutzer soll einen beliebig langen Text eingeben können. Die Eingabe ist durch eine spezielle Tastenkombination zu beenden (unter Linux ist dies `Ctrl-D`, unter Windows `Ctrl-Z`). Sie sollten daher die Eingabe Zeichen für Zeichen einlesen. Ob die Eingabe beendet wurde, können Sie bei Verwendung des Pakets `Text_IO` durch den booleschen Ausdruck

```
end_of_file(standard_input)
```

abfragen; dieser liefert den Wert `True`, wenn die Eingabe zu Ende ist.

Aufgabe 2.2 Klammerfolge (leicht) 4 Punkte

In mathematischen Ausdrücken werden zur Unterteilung gewöhnlich Klammern verwendet. Als korrekt geklammert gilt eine Zeichenkette genau dann, wenn

- zu jeder öffnenden Klammer `'('` eine schließende Klammer `')'` vorhanden ist und umgekehrt;
- im jedem solchen Klammerpaar die öffnende Klammer irgendwo links von der schließenden Klammer steht.

Schreiben Sie ein Programm, welches überprüft, ob eine gegebene Zeichenkette korrekt geklammert ist. Alle Zeichen außer `'('` und `')'` spielen für diese Korrektheit keine Rolle. Zum Einlesen von Zeichenketten siehe Hinweis 2 in Aufgabe 2.1.

Aufgabe 2.3 Zahlendarstellung (mittel) 4 Punkte

In der Vorlesung haben Sie gelernt, wie man Zahlen zu einer beliebigen positiven oder negativen Basis darstellen kann. Schreiben Sie ein Programm, welches zwei beliebige natürliche Zahlen z und b (wobei $b > 1$) einliest und anschließend die Darstellung von z zur Basis b ausgibt. Sie sollen die Darstellung *selbst* errechnen und *nicht* wie in Aufgabe 1.2(b) die vorgefertigte Ausgabefunktion von Ada dazu benutzen; letztere beherrscht im übrigen nur Basen zwischen 2 und 16. Berechnen Sie mit Ihrem Programm mindestens 23 zur Basis 7, 840 zur Basis 13, 7173001 zur Basis 34.

Hinweise: Geben Sie die Zahl 'richtig herum' aus, d.h. mit der höchstwertigen Ziffer zuerst. Wenn $b > 10$ ist, erhalten Sie 'Ziffern', die größer als 9 sein können. Geben Sie diese als in Klammern gesetzte Dezimalzahlen aus, damit die Darstellung eindeutig ist, also etwa 42 zur Basis 15 als `2(12)`.

Aufgabe 2.4 Hotel (mittel) 5+1+2 Punkte

In einem Hotel befinden sich 100 Zimmer, alle entlang eines langen Flurs. Am späten Abend gehen alle Gäste schlafen und schließen die Türen hinter sich. Leider geht es in dem Hotel nicht mit rechten Dingen zu. Um Mitternacht rennt ein Teufelchen über den Flur und reißt alle Türen wieder auf. Kurz danach kommt ein zweites Teufelchen und schlägt jede zweite Tür wieder zu. Ein weiteres schließt die dritte Tür, öffnet die sechste, schließt die neunte usw. Das jeweils n-te Teufelchen verhält sich also so, dass es sich an jeder n-ten Tür zu schaffen macht, d.h. sie öffnet, wenn sie geschlossen ist, und umgekehrt. Insgesamt passieren 100 dieser Störenfriede den Flur.

(a) Schreiben Sie ein Programm, welches das Verhalten der Teufelchen simuliert. Verwenden Sie einen geeigneten Aufzählungstyp, um den Zustand der Türen darzustellen.

(b) Wie viele Türen stehen am nächsten Morgen offen? Welche sind dies? (kurze Antwort)

(c) Begründen Sie mathematisch, warum ausgerechnet diese Türen offen stehen.

Aufgabe 2.5 Münzproblem II (Zusatzaufgabe, schwer) 10 Punkte

In der Zusatzaufgabe 1.5 von Übungsblatt 1 sollten Sie die Darstellung eines beliebigen Geldbetrags mit Münzen vom Wert 1, 7 und 11 Cent finden. Dieses Mal ist eine Erweiterung gefragt: Der Benutzer soll beliebige Münzwerte (jedoch höchstens 5) sowie einen Geldbetrag vorgeben können, und Ihr Programm soll eine Darstellung des Geldbetrags mit der minimalen Anzahl Münzen der angegebenen Werte suchen.

Allgemeine Hinweise

- Gegenüber der ursprünglichen Planung haben wir eine kleine Änderung des Ablaufs beschlossen, die jedoch zu Ihren Gunsten ist. Die Aufgabenblätter werden freitags auf der Webseite stehen, Montag in der Übung in Papierform verteilt, und sind abweichend von der Ankündigung bis freitags, 15:00 Uhr zu lösen (zuvor angepeilt:

donnerstags). Die Webseite zum Programmierkurs ist wie folgt:
`http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ws02-03/ada95/`

- Es wird darauf hingewiesen, dass der Besuch der Übungen grundsätzlich Pflicht ist, da Sie in der Lage sein müssen, Ihre Programme auf Aufforderung in der Übung zu erklären. Sind Sie einmal aus gutem Grunde verhindert, sprechen Sie dies bitte (sofern möglich) zuvor mit Ihrem Tutor ab.
- Pro Aufgabenblatt werden maximal 20 Punkte auf den Übungsschein angerechnet. Diese Punktzahl ist auch ohne Bearbeitung der Zusatzaufgaben erreichbar, letztere dienen jedoch Ihrer Übung.
- Bei Fragen wenden Sie sich bitte an Ihren Tutor oder an die Übungsleitung:
`Stefan.Schwoon@informatik.uni-stuttgart.de` oder Tel. 7816-427