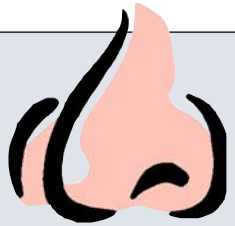
Übungsblatt 07

zu bearbeiten bis 24.06.2010

Viele kleine Aufgaben, die nach Klausur riechen...



Wir sind in der letzten Zeit immer wieder gebeten worden, anstelle von „einer großen“ lieber „mehrere kleine“ Aufgaben zu stellen. Diesem Wunsch möchten wir mit diesem Übungsblatt nachkommen und geben Euch daher mehere kleine Aufgaben, die jede für sich vom Umfang und Schwierigkeitsgrad her eine Klausuraufgabe sein könnte (bzw. sogar auch mal gewesen ist). Aus diesem Grund schmückt *eine* große Nase dieses Blatt – sie gilt für alle Aufgaben!

Das Ziel der ersten Aufgabe ist, dass Ihr Euch Gedanken über das Vorgehen beim Programmieren – also über den benötigten Algorithmus – macht. Aus Java-Sicht benötigen die Programme nur grundlegende Dinge, die wir bereits unzählige Male verwendet haben und die mittlerweile „in Fleisch und Blut“ übergegangen sein sollten. Hier kommt es auf die Anwendung Eurer erworbenen Kenntnisse an.

Die zweite Aufgabe dient zum Üben des Umgangs mit Strings, und die dritte Aufgabe soll Euch motivieren, ein bestehendes Programm abzuändern, um auf diese Weise eine Alternative zur WHILE-Schleife kennenzulernen. Wie üblich schließt das Blatt mit zwei Fragen zum Verständnis.

Aufgabe 1 – The Bad Touch 2010: Director‘s Cut (15 Punkte und 14 Bilder)

Dass die Bilder der Party am vergangenen Donnerstag mittlerweile online stehen, hat sich sicherlich herum gesprochen[1]. Aber: es gibt noch eine „Director‘s Cut“-Version der Bilder, nämlich eine kleine Zusatzsammlung von Fotos, auf denen der eine oder die andere von Euch zu sehen sind und die ein wenig nachbearbeitet wurden. Und diese Zusatzsammlung von Fotos wollen wir uns in dieser Aufgabe verdienen.

Die besagten Bilder stehen online und sind mit Zugangsdaten (Benutzername und Passwort) geschützt. Es geht nun also darum, die drei benötigten Informationen (Adresse/URL, Name, Passwort) herauszufinden, um die Fotos ansehen zu können. Dazu stellen wir nun drei Teilaufgaben, die jeweils 5 Punkte bringen und schwer klausurverdächtig sind!

1 http://www.mathematik.uni-ulm.de/badTouch2010/

An die Zugangsdaten kommt man natürlich mit ein wenig Fleißarbeit auch ohne Java und Computer. Der Sinn der Sache ist es aber selbstverständlich nicht, lediglich die Zugangsdaten zu erfahren, sondern jeweils ein kleines Java-Programm zu schreiben, dass aus der gegebenen Eingabe die jeweilige Information nach dem gegebenen Algorithmus extrahiert und ausgibt. Nur dafür gibt es Punkte!

Teilaufgabe a): Die URL zur Sammlung

Auf der Homepage findet Ihr die Datei „`url.code`“, die eine große Menge wahllos angeordneter Zeichen enthält. Aber: so ganz wahllos ist ihr Aufbau nicht, denn setzt man jedes 200. Zeichen (beginnend mit dem ersten) zusammen, dann erhält man die URL. Damit ist also schonmal klar, dass das erste Zeichen das „h“ ist, das zweite befindet sich dann 200 Positionen weiter in der Datei (also auf 201).

Tip: Ob Ihr auf dem richtigen Weg seid, könnt Ihr leicht überprüfen, denn jede URL beginnt ja mit „`http://`“.

Teilaufgabe b): Benutzername

Auch den Benutzernamen müssen wir aus einer Menge von Informationen herausfiltern. Dazu gibt es auf der Homepage die Datei „`WM-Teams.txt`“. Sie enthält zeilenweise alle teilnehmenden Länder der Fußball-WM sowie eine ganze Zahl, die durch ein Whitespace-Zeichen vom Team-Namen abgetrennt ist. Die Zahl gibt an, der wievielte Buchstabe des jeweiligen Team-Namens benötigt wird. (Beispiel: „Italy 2“ würde das ‚t‘ bringen.) Schreibt man alle auf diese Weise erhaltenen Buchstaben hintereinander, erhält man den benötigten Benutzernamen.

Aber Achtung: Es gibt jede Menge Einträge, die ungültige Zahlen enthalten, weil diese größer sind, als der Team-Name Buchstaben enthält. (Beispiel: Der Team-Name „USA“ hat drei Buchstaben, aber es ist die 4 notiert). Euer Programm muß diese ungültigen Zeilen herausfiltern. Mit anderen Worten: der gesuchte Benutzername ergibt sich ausschließlich aus den gültigen Zeilen!

Teilaufgabe c): Das Passwort

Auch das Passwort liegt verschlüsselt vor. Ihr findet es in der Datei „`password.code`“ auf unserer Homepage.

Und so funktioniert die Entschlüsselung: das Passwort setzt sich genau aus den Zeichen zusammen, die jeweils *einzeln* zwischen zwei „+“-Zeichen stehen. Alle diese Zeichen ergeben – *von hinten nach vorne gelesen* – das benötigte Passwort.

(Beispiel: Gibt man den String

`ab+c+de+f++gh+i+j`

ein, so lautet das daraus gewonnene Passwort „`ifc`“, denn nur die Zeichen ‚`c`‘, ‚`f`‘ und ‚`i`‘ sind (einzeln) von Plus-Zeichen umschlossen, und von hinten nach vorne gelesen ergibt sich deshalb das Passwort „`ifc`“.)

Wenn Ihr alle drei „Zutaten“ habt, könnt Ihr Euch nun ein paar Minuten Pause gönnen und die Bilder anschauen :-)

Aufgabe 2 – Palindrome (12 Punkte)

Betrachtet die folgenden Worte:

- Neozoen
- Lagerregal
- Otto
- Reittier
- Reliefpfeiler
- Rentner
- Rotor

Preisfrage: Was haben sie gemeinsam? Antwort: Es handelt sich bei ihnen um sogenannte „Palindrome“, also um Wörter, die vorwärts und rückwärts gelesen gleich sind.

Schreibt ein Java-Programm, das einen String von der Standardeingabe einliest und ausgibt, ob der eingegebene String ein Palindrom ist oder nicht. Verwendet in Eurem Programm anstelle der WHILE-Schleife diesmal die FOR-Schleife!

Teilaufgabe a) – 8 Punkte

Schreibt das Programm so, dass es Palindrome genau dann erkennt, wenn der eingegebene String *vollständig* aus Klein- oder Großbuchstaben besteht. (Dies ist der einfachste Fall.)

(Beispiel: Als Palindrom erkannt werden muß „OTTO“ und „otto“, nicht jedoch „Otto“.)

Teilaufgabe b) – 4 Punkte

Ergänzt Euer Programm dahingegehend, dass nun auch ein gemsischte Groß-/Kleinschreibung erkannt wird. (Nun soll also auch „Otto“ als Palindrom erkannt werden.) Wenn Ihr die Dokumentation der Klasse String bzw. die Übungsbeispiele anschaut, werdet Ihr schnell feststellen, dass diese Änderung nur einen minimalen Aufwand bedeutet!

Aufgabe 3 – Der Automat mit Switch (4 Punkte)

In der Vorlesung und in den Übungen ist Euch diese Woche die Switch-Anweisung als Alternative zu komplizierte(re)n IF-Verzweigungen vorgestellt worden. Schreibt die Beispiellösung zur Automatenaufgabe von Blatt 07 (Aufgabe 2) so um, dass sie anstelle von IF-Verzweigungen nur noch Switch-Anweisungen verwendet.

Aufgabe 4 – Fragen (4 Punkte)

1. Erklärt Eurem Tutor die Funktionsweise der Switch-Anweisung. Mit welchen Datentypen funktioniert sie (nur)?
2. Was versteht man unter dem „Durchfall“-Prinzip bei Switch-Anweisungen und welches Statement verhindert dieses „Durchfallen“?