– Praktikumsaufgabe 3 –

Thema: *Shellskripte*

Zielstellung: Erlernen des Umgangs mit Variablen, dem `test`-Kommando sowie Schleifen und Bedingungen innerhalb von Shellskripten

1. Vervollständigen Sie die verbleibenden Aufgaben aus Praktikum 2!

2. Erweitern Sie das Skript aus Aufgabe 7.b) des Praktikums 2 um eine Prüfung der Eingabeparameter. Nutzen Sie dafür die `if`-Anweisung der Shell. Geben Sie im Fehlerfall eine deskriptive Meldung aus und beenden Sie das Skript mit einem Wert ungleich 0 (`exit`).

   a) Prüfen Sie die korrekte Anzahl Parameter!

   b) Prüfen Sie, ob die Suchdatei tatsächlich existiert!

   Hinweise:

   - `$#` enthält die Anzahl der an das Skript übergebenen Parameter
   - Struktur der `if`-Anweisung der Bash:

     ```
     if <bedingung>
     then
       <anweisungsfolge>
     fi
     ```

   - numerische Vergleichsoperatoren (vgl. `man bash`):

Bash-Operator	Bedeutung	math. Operator
-lt	**l**ower **t**han	$<$
-le	**l**ower or **e**qual than	$\leq$
-eq	**eq**ual	$=$
-ne	**n**ot **e**qual	$\neq$
-gt	**g**reater **t**han	$>$
-ge	**g**reater or **e**qual	$\geq$

   - Die Prüfung von Bedingungen formulieren Sie mittels des `test`-Kommandos. Dieses hat zwei alternative Schreibweisen. Entweder Sie schreiben `test bedingung` (wie im Beispiel im Vorlesungsskript), oder Sie schreiben `[ bedingung ]`. Die beiden folgenden Formulierungen sind also äquivalent:

     ```
     if [ $1 -lt 2 ]
     if test $1 -lt 2
     ```

- Achtung! Alle Elemente innerhalb sowie die Klammersymbole (`[` und `]`) selbst müssen bei `test` durch Leerzeichen voneinander getrennt sein.

3. Erweitern Sie das Skript, indem die Suchbegriffe nicht mehr an der Kommandozeile angegeben, sondern mittels `read` interaktiv eingelesen werden. Das Skript soll in einer Schleife

   - einen Suchbegriff einlesen,
   - die Häufigkeit des Suchbegriffes in der zu durchsuchenden Datei ermitteln,
   - Häufigkeit und Suchbegriff auf dem Bildschirm ausgeben.

   Bei Eingabe eines leeren Suchbegriffes soll das Skript beendet werden.

   Nutzen Sie danach eine Umleitung der Standardeingabe **an der Kommandozeile**, um beim Aufruf des Skriptes nicht mehr interaktiv einzelne Suchbegriffe einzugeben, sondern eine (vordefinierte) Liste von Suchbegriffen aus einer Datei einzulesen und automatisiert abzuarbeiten. Als Ergebnis sollte eine (einfache) Tabelle am Bildschirm ausgegeben werden, die in absteigender Häufigkeit sortiert ist.

   Hinweise:

   - `read tier` liest von der Standardeingabe eine Zeichenkette und legt diese in der Variablen `tier` ab
   - die Variable muss nicht vorher definiert werden
   - den Wert einer Variablen referenziert man mittels `$`, also z. B. `echo $tier`
   - als Schleifenkonstrukt kann z. B. `while` eingesetzt werden (`man bash`). Das Konstrukt hat folgende Form:

     ```
     while <bedingung>
     do
       <anweisungsfolge>
     done
     ```

   - Eine Schleife können Sie mittels `break` verlassen (analog zu C).
   - Nutzen Sie `exit` und einen numerischen Rückgabewert, um das Skript zu beenden. Gewöhnlich wird 0 zurückgeliefert, wenn ein Programm/Skript erfolgreich abgearbeitet wurde und ein Wert größer 0, wenn ein Fehler aufgetreten ist.
   - Um eine Zeichenkettenvariable `$wort` mit einer Zeichenkette `"foo"` zu vergleichen, kann man folgenden Ausdruck einsetzen: `[ "$wort" = "foo" ]`

4.* „Write a shell script that takes a directory name as an argument and writes to standard output the maximum of the lengths of all filenames in that directory. If the script's

argument is not a directory name, write an error message to standard output and exit with nonzero status.“ (Mark G. Sobell, A Practical Guide to Linux Commands, Editors, and Shell Programming, Prentice Hall, 2005)

5.** Implementieren Sie in C eine (rudimentäre) Shell!