– Praktikumsaufgabe 5 –

Thema: *Resultate und Funktionen in der Shell*

Zielstellung: Erlernen der Generierung und Auswertung von Resultatwerten mittels Shellskripten, Nutzung von Funktionen in Shellskripten

1. a) Schreiben Sie ein Shellskript, das

   - ein Kommando als Kommandozeilenparameter übernimmt,
   - das Kommando ausführt und
   - den Rückgabewert des Kommandos anzeigt.

   Können Sie Kommandos finden, die Rückgabewerte ungleich Null liefern?

   Hinweis: Den (numerischen) Rückgabewert des letzten ausgeführten Kommandos erhält man mit `$?`.

   b) Erweitern Sie das Skript, so dass dem Kommandozeilenparameter beliebig viele Optionen folgen dürfen! Schauen Sie sich dazu die Variable `$*` an.

2. Schreiben Sie ein Shellskript, das einen Zufallswert auf das Terminal schreibt und diesen Zufallswert an die rufende Umgebung mittels `exit` zurückliefert! Überzeugen Sie sich dann mittels des Skriptes aus Aufgabe 1, dass keine Werte, die größer als 255 sind, an die rufende Umgebung zurückgegeben werden können. Hinweis: Jede Referenz der Bash-Variablen `RANDOM` liefert eine Zufallszahl (im Intervall [0;32767]).

3. Entwerfen Sie eine Funktion, die als Parameter einen Dateinamen übernimmt. Die Funktion soll per `scp` die Datei auf einen (statisch festgelegten) anderen Rechner kopieren. Die Funktion soll 0 an die rufende Instanz übergeben, wenn der Kopiervorgang erfolgreich war, 1 ansonsten.

   Schreiben Sie nun ein Shellskript, das eine Pfadangabe übernimmt und dann alle Dateien, die sich in diesem Pfad befinden, mittels der entworfenen Funktion kopiert. Der Nutzer soll über Erfolg/Misserfolg der Kopiervorgänge entsprechend informiert werden.

   Hinweise:

   - `scp` kopiert (wie `cp`) Dateien, jedoch über Rechnergrenzen hinweg. Die Aufrufsyntax ist `scp <quelle> <ziel>`. Entweder `<quelle>` oder `<ziel>` muss eine lokale Pfadangabe sein, die jeweils andere adressiert einen entfernten Rechner nach dem Schema `nkz@rechnername:pfadangabe`, also zum Beispiel `s23500@isys121:~/tmp`.
     Um alle Dateien aus dem lokalen Unterverzeichnis `bla/bin` auf den Rechner gollum mit gleichem Nutzerkennzeichen in das Verzeichnis `/archive/backup`

zu kopieren, ruft man

```
scp ./bla/bin/* gollum:/archive/backup
```

- Funktionen und Prozeduren werden in der Bash folgendermaßen definiert:

```
funktionsname ()
{
  kommando1
  kommando2
  ...
}
```

  Die Definition muss vor dem Aufruf erfolgen, d. h., Vorwärtsdeklarationen gibt es nicht.

- Parameter an Funktionen werden mit den Positionsparametern `$1`, `$2`, ... `$9` übergeben.

- Resultatwerte von Funktionen können Sie beispielsweise via `stdout` übergeben. Die gerufene Funktion schreibt das Ergebnis einfach nach `stdout`

```
foo ()
{
    echo $result
}
```

  Die rufende Instanz verhindert die Ausgabe nach `stdout` durch Einsatz des Eval-Operators:

```
a=`foo <somearg>`
a=$(foo somemorearg)
```

  Beide Methoden sind gleichwertig.

- Alternative dazu ist es auch möglich, Resultate mittels `return` zu übergeben, jedoch nur Werte zwischen 0 und 255 (ähnlich `exit`).

- Mehr Informationen zur Resultatkommunikation finden Sie in diesem Artikel: *http://www.linuxjournal.com/content/return-values-bash-functions*.

- Für die Lösung benötigen Sie noch das `for`-Statement der Bash, das die folgende Form besitzt:

```
for var in suchspec
do
   ...
done
```

  Die Suchspezifikation (z. B. `*.*`) wird expandiert und pro Schleifeniteration jeweils ein Wert der Variablen `var` zugewiesen.

4. Analysieren Sie, was der folgende Ausdruck tut (das führende Dollarzeichen steht für den Prompt):

```
$ :(){ :|:&};:
```

Achtung: Führen Sie den Ausdruck nicht aus! Die Analyse muss rein theoretisch erfolgen!