Wirtschaftsinformatik – Ausgewählte Lösungen zu den Übungen 1 - 4

Übung 1

- Berechtigungskonzept (vollständige Tabelle)

Befehl	binär	Berechtigung
chmod 440 <dateiname>	*100 100 000*	*r--r----*
chmod 571	*101 111 001*	r-xrwx--x
chmod 027 <dateiname>	*000 010 111*	*----w-rwx*
chmod 220	*010 010 000*	-w--w----

Übung 2

- *lsass* steht für **L**ocal **S**ecurtiy **A**uthentication **S**ubsystem. Der Prozess ist also ein Sicherheitsdienst, der die Zugriffsrechte und Benutzeranmeldung überwacht.
- *csrss.exe* steht für **C**lient/**S**erver **R**un-Time **S**ub**s**ystem und ist zuständig für Konsolenfenster, das Anlegen und Beenden von Threads und implementiert manche Teile der 16-Bit virtuellen MS-DOS-Umgebung.
- *svchost.exe* ist ein Systemprozess mit dessen Hilfe dll-Dateien ausgeführt werden. Normalerweise werden darin bestimmte Dienste geladen. Der Prozess kann beliebig oft vorkommen, da jede Instanz für unterschiedliche Prozesse verantwortlich ist. Details hierüber können mit dem *Process Explorer* ausgelesen werden.
- *smss.exe* steht für **S**ession **M**anager **S**ub**s**ystem (Sitzungsmanager) und ist für das Starten von Diensten und Subsystemen zuständig

Übung 3

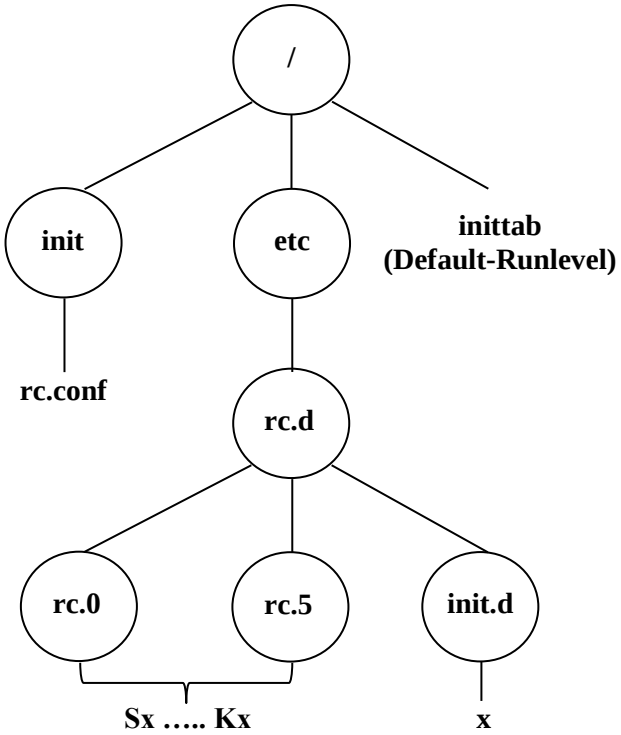

Abbildung 1 - Linux-Runlevel – Grafische Verkettung der zugehörigen Skripte

- Übersicht *Runlevel* unter Linux
  - Runlevel 0 = halt
  - Runlevel 1 = Single User Mode
  - Runlevel 2 = Multiuser, without NFS
  - Runlevel 3 = Full multiuser mode
  - Runlevel 4 = unused
  - Runlevel 5 = X11 (= grafische Oberfläche)
  - Runlevel 6 = reboot

Übung 4

Lösung für erweiterte Aufgabenstellung

```
#!/bin/bash

echo "Start das Programms"

#Hilfefunktion
if [ $1 = "-help" ] ; then
    echo "---------------HILFE----------------"
    echo "Das Programm dient dazu, Dateien mit bestimmter Endung in ein
frei wählbares Verzeichnis zu verschieben und, falls gewünscht, eine
Protokollierungsdatei des Vorgangs erstellt"
    echo "------------------------------------"
    echo "Auflistung der möglichen Parameter:"
    echo "1.Parameter: Dateiendung"
    echo "2.Parameter: Verzeichnisname"
    echo "3.Parameter: Protokollierung"
    exit 0
fi

dateiendung="$1"
verzeichnis="$2"
protokollierung="$3"

#Test der Übergabeparameter
if [ -z $dateiendung ] ; then
    echo "Sie haben keine Dateiendung spezifiert"
    exit 0
fi

if [ -z $verzeichnis ] ; then
    echo "Sie haben kein Verzeichnis übergeben"
    exit 0
fi

if [ -z $protokollierung ] ; then
    echo "keine Protokollierung übergeben --> default-Wert"
    protokollierung="N"
fi

#Überprüfung ob Verzeichnis bereits vorhanden
if [ -d $verzeichnis ]; then
    echo "Verzeichnis $verzeichnis vorhanden"
else
    echo "Verzeichnis $verzeichnis noch nicht vorhanden --> wird erstellt"
    mkdir $verzeichnis
fi
```

```
#Dateien werden verschoben
mv *.$dateiendung $verzeichnis

#Protokollierung
if [ $protokollierung = "J" ] |||[ $protokollierung = "Y" ] ; then

    ls $verzeichnis/*.$dateiendung | wc -w > prot.txt
    echo "Dateien in das Verzeichnis $verzeichnis verschoben" >> prot.txt
    ls *.$1 >> prot.txt
fi

echo "Programm erfolgreich beendet!"
```

Verwendung des `read` Befehls

- Anstatt die Parameter dem Skript zu übergeben, können diese auch während der Ausführung interaktiv eingelesen werden. Hierzu steht der `read` Befehl zur Verfügung. Folgende Syntax liest z. B. das Zielverzeichnis von der Konsole ein und speichert ihn in der Variable „verzeichnis“:

  ```
  read -p "Bitte geben Sie das Zielverzeichnis an: " zielverzeichnis
  ```

  Die Variable „zielverzeichnis“ kann anschließend mittels `$zielverzeichnis` im Skript verwendet werden, um den Ordner anzulegen und die Dateien zu verschieben. Die Dateiendung kann ebenfalls interaktiv eingelesen und verwendet werden:

  ```
  read -p "Bitte geben Sie die Dateiendung an: " dateiendung
  ```

- Mittels einer Schleife lässt sich die Eingabe nun solange wiederholen, bis eine Eingabe getätigt wurde. Mit folgendem Code lässt sich z. B. die Dateiendung einlesen. Das obige Skript ist entsprechend anzupassen::

  Aus obigem Skript zu entfernen

  ```
  dateiendung="$1"
  verzeichnis="$2"
  protokollierung="$3"

  if [ -z $dateiendung ] ; then
      echo "Sie haben keine Dateiendung spezifiert"
      exit 0
  fi
  ```

  Neuer Code

  ```
  dateiendung=""
  while [ -z $dateiendung ]
  do
  read -p "Bitte geben Sie die Dateiendung an: " dateiendung
  done
  ```