Abteilung für INFORMATIK

HTL ST. PÖLTEN

Titel: Labor 8

Klasse: 3BHIF

Name: Haiden

Gruppe: 01

Aufgabe: 19.02.2020 Abgabe: 04.03.2020

Inhaltsverzeichnis

Shell-Skripte – Labor 8

1 Shell-Skripte – Labor 8

1.1 Aufgabe 1

Schreiben Sie ein Shellskript, das

- alle übergebenen Argumente
- die Anzahl der Argumente
- den Namen des Shellskripts
- den Inhalt des Shellskripts

auf stdout ausgibt.

1.1.1 Skript

```
#!/bin/bash
#
#Uebung 08 Beispiel 1
#Haiden Niklas 3BHIF-06
#2020-02-19

if test $# -eq 0
then
      echo "Es wurden keine Argumente angegeben."
      exit 1
fi

anz_Args="Anzahl der Argumente: "$#
uebergebene_Args="Uebergebene Argumente "$*
skript_Name="Name des Skripts: "$0

echo $anz_Args
echo $uebergebene_Args
echo $skript_Name
cat $0

exit 0
```

1.1.2 Testläufe

1.1.2.1 Aufruf mit Parametern

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe1_labor8.sh hello
welt
Anzahl der Argumente: 2
Uebergebene Argumente hello welt
Name des Skripts: ./aufgabe1_labor8.sh
#!/bin/bash
#
#Uebung 08 Beispiel 1
#Haiden Niklas 3BHIF-06
#2020-02-19

if test $# -eq 0
then
      echo "Es wurden keine Argumente angegeben."
      exit 1
fi

anz_Args="Anzahl der Argumente: "$#
uebergebene_Args="Uebergebene Argumente "$*
skript_Name="Name des Skripts: "$0

echo $anz_Args
echo $uebergebene_Args
echo $skript_Name
cat $0

exit 0
```

1.1.2.2 Aufruf ohne Parameter

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe1_labor8.sh
Es wurden keine Argumente angegeben.
```

1.2 Aufgabe 2

Das Shellskript soll genau ein Argument entgegennehmen und dieses dann zusammen mit einer Ausgabe in der Form

```
Es wurde der Parameter "hallo" eingegeben
```

ausgeben. Zusätzlich soll jedes Mal, wenn das Script ausgeführt wird, ein Logeintrag mit dem Usernamen, dem aktuellem Timestamp im Format yyyy-MM-dd hh:mm:ss und dem eingegebenen Parameter an die Datei ue_02.log angehängt werden.Wurde nicht genau ein Parameter angegeben, so beinhaltet der Logeintrag den Usernamen, den Timestamp, eine Fehlermeldung und die Anzahl der übergebenen Argumente.

1.2.1 Skript

```
#!/bin/bash
#
#Uebung 08 Beispiel 2
#Haiden Niklas 3BHIF-06
#2020-02-19

ausgabe_shell="Es wurde der Parameter \""$1"\" ausgegeben"
username=$USER
timestamp=$(date +"%Y-%m-%d %H-%M-%S")
fehlermeldung="nix"
ausgabe_file=$USER", "$timestamp", "$1

if test $# -ne 1
then
     if test $# -eq 0
     then
          fehlermeldung="Es wurde kein Parameter ausgegeben."
     fi

     if test $# -gt 1
     then
          fehlermeldung="Es wurde mehr als ein Parameter
ausgegeben."
     Fi
```

```
      echo $fehlermeldung
      ausgabe_file=$USER", "$timestamp", "$fehlermeldung", "$#
      echo $ausgabe_file>>ue_02.log
      exit 1
fi


echo $ausgabe_file >> ue_02.log
echo $ausgabe_shell
```

1.2.2 Testläufe

1.2.2.1 Aufruf ohne Parameter

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe2_labor8.sh
Es wurde kein Parameter ausgegeben.
schueler@Debian10nvs:~/skripte_uebung8$ cat ue_02.log
schueler, 2020-02-19 09-46-21, Es wurde kein Parameter ausgegeben.,
0
```

1.2.2.2 Aufruf mit Parameter:

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe2_labor8.sh hello
Es wurde der Parameter "hello" ausgegeben
schueler@Debian10nvs:~/skripte_uebung8$ cat ue_02.log
schueler, 2020-02-19 09-46-21, Es wurde kein Parameter ausgegeben.,
0
schueler, 2020-02-19 09-50-04, hello
```

1.2.2.3 Aufruf mit mehr als zwei Parametern

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe2_labor8.sh hello
world
Es wurde mehr als ein Parameter ausgegeben.
schueler@Debian10nvs:~/skripte_uebung8$ cat ue_02.log
```

```
2020-02-19 09-31-06

schueler, 2020-02-19 09-46-21, Es wurde kein Parameter ausgegeben., 
0

schueler, 2020-02-19 09-50-04, hello

schueler, 2020-02-19 09-52-15, Es wurde mehr als ein Parameter 
ausgegeben., 2
```

1.3 Aufgabe 3

Schreiben sie ein Script, dasauf 3 unterschiedliche Argumente reagiert:

- Wenn es mit dem Argument login aufgerufen wird, dann soll es jene Zeile aus der Datei /etc/passwd ausgeben, die dem gerade aktivem User zugeordnet ist.
- Wenn es mit dem Argument ps-root aufgerufen wird, soll dasKommando ps -efaufgerufen werden, und aus dessen Ausgabe sollen alle Zeilen, die das Wort root enthalten, herausgefiltert und dann seitenweise angezeigt werden.
- Wenn es mit dem Argument -h aufgerufen wird, dann soll ein kurzer Hilfetext zur Benutzung erscheinen.
- Wenn es ohne Argumente aufgerufen wird, dann soll ebenfalls der Hilfetext ausgegeben werden.

Erklären sie in einem Kommentar auch die Bedeutung des Kommandos ps –ef.

1.3.1 Skript:

```
#!/bin/bash

#

#Uebung 08 Beispiel 3

#Haiden Niklas 3BHIF-06

#2020-02-19


# Funktion, um die Hilfe nicht zweimal schreiben zu müssen

hilfe_ausgabe () {

     echo "Dies ist die Hilfe fuer Aufgabe 3

------------------------------------------------------------------------
----------------------------------------

Mit dem Parameter -login wird die Passwd-Zeile des aktuellen Users 
ausgegeben.

------------------------------------------------------------------------
----------------------------------------

Mit dem Parameter -h wird dieser Hilfetext ausgegeben.

------------------------------------------------------------------------
----------------------------------------
```

Shell-Skripte – Labor 8

```
Mit dem Parameter -ps-root werden alle Zeilen vom kommando ps -ef
herausgefiltert, die das Wort root enthalten."
}

if test $# -gt 1
then
     echo "Mehr als 1 Argument wurde angegeben."
     exit 1
fi

if test $# -eq 0
then
     hilfe_ausgabe
     exit 0
fi

if test $1 = "-h"
then
     hilfe_ausgabe
     exit 0
fi

if test $1 = "-login"
then
     cat /etc/passwd | grep $USER
     exit 0
fi

if test $1 = "-ps-root"
then
     ps -ef | grep "^root"
     exit 0
fi
```

```
exit 0
```

1.3.2 Testläufe

1.3.2.1 Aufruf ohne Parameter

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe3_labor8.sh

Dies ist die Hilfe fuer Aufgabe 3

-------------------------------------------------------------------
----------------------------------------

Mit dem Parameter -login wird die Passwd-Zeile des aktuellen Users
ausgegeben.

-------------------------------------------------------------------
----------------------------------------

Mit dem Parameter -h wird dieser Hilfetext ausgegeben.

-------------------------------------------------------------------
----------------------------------------

Mit dem Parameter -ps-root werden alle Zeilen vom kommando ps -ef
herausgefiltert, die das Wort root enthalten.
```

1.3.2.2 Aufruf mit Parameter –h

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe3_labor8.sh -h

Dies ist die Hilfe fuer Aufgabe 3

-------------------------------------------------------------------
----------------------------------------

Mit dem Parameter -login wird die Passwd-Zeile des aktuellen Users
ausgegeben.

-------------------------------------------------------------------
----------------------------------------

Mit dem Parameter -h wird dieser Hilfetext ausgegeben.

-------------------------------------------------------------------
----------------------------------------

Mit dem Parameter -ps-root werden alle Zeilen vom kommando ps -ef
herausgefiltert, die das Wort root enthalten.
```

1.3.2.3 Testlauf mit –login

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe3_labor8.sh -login

schueler:x:1000:1000:Schueler der Abteilung
Informatik,,,:/home/schueler:/bin/bash
```

Shell-Skripte – Labor 8

1.3.2.4 Testlauf mit –ps-root

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe3_labor8.sh -ps-
root
root         1     0  0 07:57 ?        00:00:03 /sbin/init
root         2     0  0 07:57 ?        00:00:00 [kthreadd]
```

1.4 Aufgabe 4

Schreiben sie ein Script, das je nach aktueller Systemzeit "Good morning" (ab 00:00), "Good afternoon" (ab 12:00) oder "Good evening" (ab 18:00) ausgibt.

1.4.1 Skript

```
#!/bin/bash
#
#Uebung 08 Beispiel 4
#Haiden Niklas 3BHIF-06
#2020-02-19

time=$(date +"%H")

if test $time -ge 08
then
     echo "Good morning"
fi

if test $time -ge 12
then
     echo "Good afternoon"
fi

if test $time -ge 18
then
     echo "Good evening"
fi
```

Shell-Skripte – Labor 8

1.4.2 Testläufe

1.4.2.1 Aufruf: ./aufgabe4_labor8.sh

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe4_labor8.sh
Good morning
```

1.5 Aufgabe 5

Schreiben Sie ein Shellskript, das in der als 1. Parameter übergebenen Datei nach dem Wort sucht, das als 2.Parameter übergeben wurde. Das Skript soll alle Zeilen und deren Anzahl ausgeben, in denen das gesuchte Wort vorkommt.

Returnwerte:

- 0 fehlerfrei
- 1 falsche Anzahl an Parametern
- Datei $1 existiert nicht
- Datei $1 existiert, darf aber nicht gelesen werden

1.5.1 Skript:

```
#!/bin/bash
#
#Uebung 08 Beispiel 5
#Haiden Niklas 3BHIF-06
#2020-02-19


FILE=$1
SEARCH_TERM=$2
if [ $# -ne 2 ]
then
     echo "Es wurden nicht 2 Parameter angegeben."
     exit 1
fi


if [ ! -f $FILE ]
then
     echo "Die Datei existiert nicht."
     exit 2
```

```
fi


if [ ! -r $FILE ]
then
     echo "Die Datei ist existiert nicht und / oder ist nicht
lesbar."
     exit 3
fi


grep $SEARCH_TERM $FILE
anzahl=$(grep $SEARCH_TERM $FILE | wc -l)


echo "Anzahl: " $anzahl
```

1.5.2 Testläufe

1.5.2.1 Normaler Fall – Datei existiert

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe5_labor8.sh
testfile.txt Katze
Hallo hier sollte Katze stehen
Hier steht wieder Katze
Anzahl:  2
schueler@Debian10nvs:~/skripte_uebung8$ echo $?
0
```

1.5.2.2 Datei existiert nicht

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe5_labor8.sh
somefile.txt Katze
Die Datei existiert nicht.
schueler@Debian10nvs:~/skripte_uebung8$ echo $?
2
```

1.5.2.3 Datei ist nicht lesbar

```
schueler@Debian10nvs:~/skripte_uebung8$ chmod 000 testfile.txt
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe5_labor8.sh
somefile.txt Katze
Die Datei existiert nicht.
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe5_labor8.sh
testfile.txt Katze
```

Shell-Skripte – Labor 8

```
Die Datei ist existiert nicht und / oder ist nicht lesbar.
schueler@Debian10nvs:~/skripte_uebung8$ echo $?
3
```

1.5.2.4 Anzahl der Parameter ist nicht richtig

```
schueler@Debian10nvs:~/skripte_uebung8$ ./aufgabe5_labor8.sh
Es wurden nicht 2 Parameter angegeben.
schueler@Debian10nvs:~/skripte_uebung8$ echo $?
1
```

1.6 Aufgabe 6

Schreiben Sie ein einfaches Shellskript, das eine beliebige Anzahl an Argumenten übernimmt und das jedes Argument mit einem vorangestellten Hallo in einer eigenen Zeile ausgibt. Das Skript soll z.B. wie folgt arbeiten:

```
./ue_03_1_3a00.sh Hugo Franz
Hallo Hugo
Hallo Franz
```

1.6.1 Skript

```
#!/bin/bash
#
#Uebung 08 Beispiel 6
#Haiden Niklas 3BHIF-06
#2020-03-03

if [ $# -lt 1 ]
then
        echo "Es muss mindestens ein Parameter vorhanden sein."
        exit 1
fi

str=$*

IFS=' '
read -ra ADDR <<< "$str"
for i in "${ADDR[@]}"; do
```

Shell-Skripte – Labor 8

```
    echo "Hallo, "$i
done
```

Dieses Skript benutzt den eingebauten `read` Command der Bash. Über die Variable `IFS` wird ein Delimiter festgelegt, welcher bei dem Splitten benutzt werden soll. Mit der Option -a sagt man dem eingebauten Command, dass er die Wörter in ein Array in diesem Fall `ADDR` schreiben soll. In der For-Schleife werden die einzelnen Elemente aus dem Array herausgeholt und in die Variable i gespeichert. Diese wird dann mittels `echo Befehl` ausgegeben.

1.6.2 Testläufe

1.6.2.1 Guter Testlauf

```
niklas@nhaiden:~/nvs/labor8$ ./aufgabe6_labor8.sh Susi Franzl Bernd
Hallo, Susi
Hallo, Franzl
Hallo, Bernd
```

1.6.2.2 Testlauf mit zu wenig Parametern

```
niklas@nhaiden:~/nvs/labor8$ ./aufgabe6_labor8.sh
Es muss mindestens ein Parameter vorhanden sein.
```

1.7 Aufgabe 7

Schreiben Sie ein Shellskript, das genau zwei ganze Zahlen a und b als Parameter übernimmt und das mit Hilfe einer while-Schleife alle Zahlen von a bis b (inklusive), jeweils genau durch ein Blank getrennt ausgibt. Sollten nicht genau 2 Parameter übergeben worden sein, beendet sich das Skript mit 1, wenn einer der beiden Parameter keine ganze Zahl ist mit 2 und wenn a > b ist mit 3.

Beispiel: `./ue_03_2_3a00.sh 2 8`

```
2345678
```

1.7.1 Skript:

```
#!/bin/bash
#
#Uebung 08 Beispiel 7
#Haiden Niklas 3BHIF-06
#2020-03-03


if [ $# -ne 2 ]
then
        echo "Zu wenige / zu viele Parameter."
        exit 1
fi
```

Shell-Skripte – Labor 8

```
if ! [[ "$1" =~ ^[+-]?[0-9]+$ ]] || ! [[ "$2" =~ ^[+-]?[0-9]+$ ]];
then
        echo "Es sind nur Integer erlaubt!"
        exit 2
fi

if [ $1 -gt $2 ]
then
        echo "Der erste Parameter darf nicht groesser als der zweite
sein!"
        exit 3
fi

counter=$1

while [ $counter -le $2 ]
do
        echo $counter
        counter=$(($counter+1))
done
```

1.7.2 Testläufe

1.7.2.1 Normaler Testlauf mit richtigen Parametern

```
niklas@nhaiden:~/nvs/labor8$ ./aufgabe7_labor8.sh 2 8
2
3
4
5
6
7
8
```

Shell-Skripte – Labor 8

1.7.2.2 Testlauf mit falscher Anzahl an Parametern

```
niklas@nhaiden:~/nvs/labor8$ ./aufgabe7_labor8.sh 2

Zu wenige / zu viele Parameter.

niklas@nhaiden:~/nvs/labor8$ ./aufgabe7_labor8.sh

Zu wenige / zu viele Parameter.
```

1.7.2.3 Testlauf mit nicht gueltigen Parametern

```
niklas@nhaiden:~/nvs/labor8$ ./aufgabe7_labor8.sh xx 2

Es sind nur Integer erlaubt!

niklas@nhaiden:~/nvs/labor8$ ./aufgabe7_labor8.sh 2 xx

Es sind nur Integer erlaubt!
```

1.7.2.4 Testlauf wo a größer b

```
niklas@nhaiden:~/nvs/labor8$ ./aufgabe7_labor8.sh 3 1

Der erste Parameter darf nicht groesser als der zweite sein!
```