TECH STARTER

1.2.2. Grundlagen der Befehlszeile in Linux (100 Punkte)

Achtung: Hier gibt es eine Extra-Aufgabe, in der ihr zusätzliche Punkte sammeln könnt 🙂

1. Grundlagen der Linux Shell (25 Punkte)

a. Was ist eine Shell in Linux? (5 Punkte)
b. Wie ist ein Befehl in Linux aufgebaut? Beschreibe es kurz. (5 Punkte)
c. Teile die folgenden Zeilen in die Bestandteile Befehl, Option(en)/Parameter und Argument(e) auf. Fülle die Tabelle mit den weiteren Befehlen aus. Bestimme ferner den Befehlstyp

```
i.   cp -r /home/user/docs /backup/docs (3 Punkte)
ii.  find /var/log -name "*.log" (3 Punkte)
iii. grep -i "error" /var/log/syslog (3 Punkte)
iv.  tar -czvf archive.tar.gz /home/user/data (3 Punkte)
v.   chmod 755 /home/user/script.sh (3 Punkte)
```

	Befehl	Option	Argument	Befehlstyp
0.	cd		/home/user	Built-in
i.	cp	-r	/home/user/docs /backup/docs	Extern
ii.	find	-name “*.log”	/var/log	Extern
iii.	grep	-i “error”	/var/log/syslog	Extern
iv.	tar	-czvf	archive.tar.gz /home/user/data	Extern
v.	chmod		755 /home/user/script.sh	Extern

2. Quoting (25 Punkte)

Beantworte die Teilaufgaben in jeweils 1-2 kurzen Sätzen.

a. Erkläre, was passiert, wenn du folgenden Befehl ausführst: (5 Punkte)

```
echo "Hello, $USER!"
   -  Hello + Benutzernamen
```

- Doppelte Anführungszeichen lassen zu, dass $ mitinterpretiert wird als Sonderzeichen, d.h. wir können die Variable auflösen. (Variable = Username)

b. Was passiert, wenn du den folgenden Befehl eingibst? Erkläre die Rolle der Anführungszeichen: (5 Punkte)
`mkdir "My Folder"`
   - Wir erstellen einen Ordner mit dem Namen “My Folder”
   - Wir erstellen NICHT zwei Ordner, da das Leerzeichen seine Sonderbedeutung als Trenner zwischen zwei Argumenten verliert (Doppelte Anführungszeichen)

c. Wie unterscheidet sich der folgende Befehl von dem vorherigen? Erkläre den Unterschied: (5 Punkte)
`mkdir 'My Folder'`
   - Wir erstellen einen Ordner mit dem Namen “My Folder”
   - Alle Sonderzeichen WÜRDEN ihre Bedeutung verlieren. Wir haben in diesem Fall aber nur das Leerzeichen als Sonderzeichen (das wird auch vom doppelten Anführungszeichen aufgelöst)

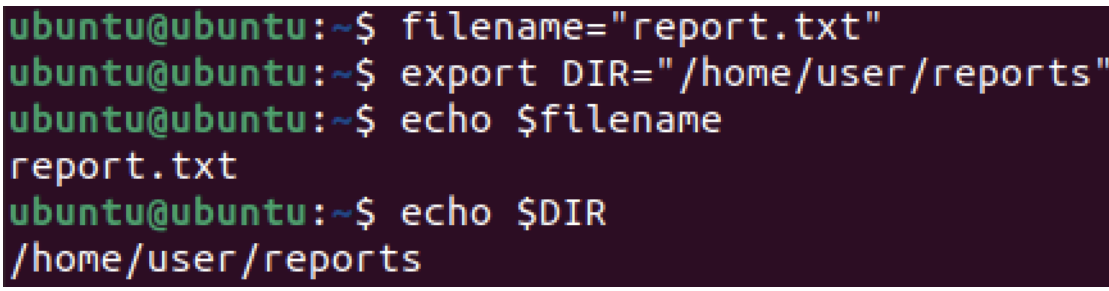
d. Erkläre, was passiert, wenn du den folgenden Befehl ausführst: (5 Punkte)
`echo 'Hello, $USER!'`
   - Die Ausgabe wäre: Hello, $USER
   - Alle Sonderzeichen verlieren ihre Bedeutung, also auch das $-Zeichen, so dass die Variable nicht aufgelöst werden kann.

e. Was ist der Unterschied zwischen den Befehlen `touch "file name"` und `touch 'file name'`? Erkläre den Unterschied und welche Auswirkungen dies hat. (5 Punkte)
   1. Es würde dieselbe Ausgabe rauskommen
   2. Erst wenn wir Sonderzeichen, wie $, \ oder ‘ dann würde es einen signifikanten Unterschied in der Ausgabe geben.

3. Variablen (25 Punkte)

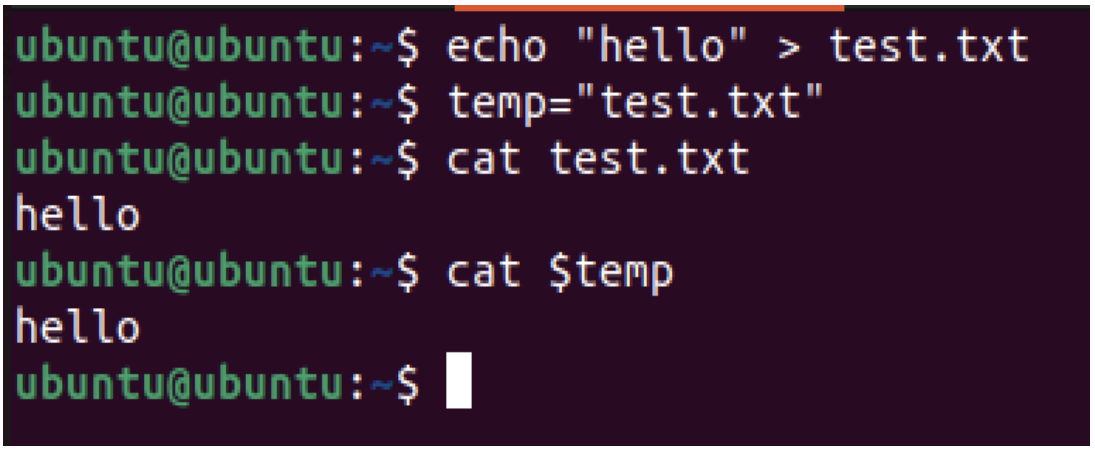
Mache pro Teilaufgabe bitte einen Screenshot bzw. 1-2 kurze Sätze.

a. Erzeuge eine lokale Variable `filename` mit dem Wert `"report.txt"`. Setze anschließend eine Umgebungsvariable `DIR` mit dem Wert `"/home/user/reports"` und zeige beide Variablen an. (6 Punkte)
   - filename=”report.txt”
   - export DIR=”/home/user/reports”
   - echo $filename
   - echo $DIR
   -

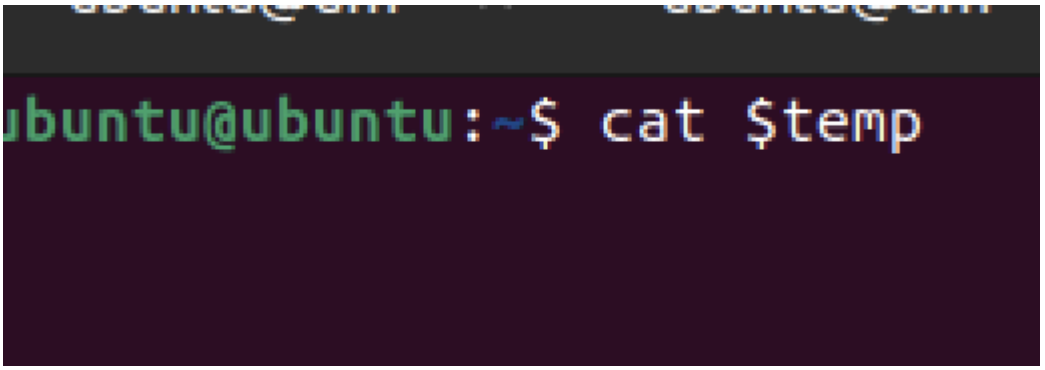


b. Setze eine lokale Variable `count` auf den Wert `10`, starte eine neue Sub-Shell (durch den Befehl `bash`) und überprüfe, ob die Variable `count` in der neuen Sub-Shell verfügbar ist. Erkläre das Ergebnis. (6 Punkte)
   - count=10
   - dann bash. Wenn wir dann echo $count eingeben, dann erhalten wir keine Ausgabe, da wir von der Subshell keinen Zugriff auf die lokale Variable haben.

c. Setze eine lokale Variable `temp_value`, weise ihr einen Wert zu, und rufe ein externes Programm (z.B. `ls`) auf, das den Wert dieser Variable verwenden soll. Erkläre, warum die Variable nicht im externen Programm verfügbar ist, und wie man dies beheben kann. 6 Punkte)
   - temp_value="/home/"
   - ls $temp_value
   -



   -



d. Erzeuge eine lokale Variable `DATE` und setze sie auf das aktuelle Datum (Tipp: Verwende den Befehl `date`). Erzeuge anschließend eine Umgebungsvariable `DATE_GLOBAL` mit demselben Wert und überprüfe die Verfügbarkeit beider Variablen in einer neuen Sub-Shell. (7 Punkte)
   - DATE=$(date) [date ist hier ein Befehl, der das aktuelle Datum aufruft]
   - export DATE_GLOBAL = $DATE
   - bash
   - echo $DATE (hier passiert nichts)
   - echo $DATE_GLOBAL (hier wird etwas ausgegeben, weil es eine Umgebungsvariable ist)

4. Man Pages und info-Seiten: (25 Punkte)

Mache pro Teilaufgabe bitte einen Screenshot bzw. 1-2 kurze Sätze.

a. Finde die Manpage für das Shell Builtin `echo`. Warum unterscheidet sich die Ausgabe von `man echo` von `echo --help`? (5 Punkte)
   echo –help → gibt uns einfach nur –help aus, (/bin/echo --help alternativ help echo eingeben)
   man echo → gibt uns ausführliche Übersicht über die Optionen und Paramter

b. Öffne die Manpage des Befehls `grep`. Finde heraus, was die Option `-i` bewirkt, und erkläre, wie du damit nach einem Begriff in einer Datei suchen kannst, unabhängig von Groß- und Kleinschreibung. (5 Punkte)
   - Groß- und Kleinschreibung wird ignoriert

c. Öffne die Info-Seite für `coreutils`. Navigiere zur Beschreibung des Befehls `cp` und finde heraus, welche Optionen es gibt, um eine Sicherungskopie einer Datei zu erstellen. (5 Punkte)
   - Öffnen mit info coreutils
   - Suche mit / nach cp
   - Scrolle nach Backup, dann sehen wir, dass -b bzw. –backup Sicherungskopien ermöglicht

d. Vergleiche die Informationen, die du in der Manpage von `tar` und in der Info-Seite von `tar` findest. Welche Informationen bietet die Info-Seite zusätzlich? (5 Punkte)
   - man tar:
   - info tar: Bietet mehr Übersicht über den Befehl bzw. auch Anwendungsbeispiele

5. **Zusatzaufgabe** find: (10 Punkte)

Mache pro Teilaufgabe bitte einen Screenshot bzw. 1-2 kurze Sätze.

a. Verwende den Befehl `find`, um in deinem Home-Verzeichnis nach allen Dateien zu suchen, deren Namen mit `log` enden. Zeige nur die Dateinamen an, ohne den Pfad. (2 Punkte)

b. Verwende `find`, um alle `.txt`-Dateien in einem Verzeichnis und seinen Unterverzeichnissen zu finden und dann mit dem Befehl `wc -l` die Anzahl der Zeilen in jeder dieser Dateien zu zählen. (2 Punkte)

c. Finde alle Dateien in `/etc`, die in den letzten 7 Tagen geändert wurden, und gib ihre Namen und Änderungszeiten aus. (3 Punkte)

d. Finde in deinem Home-Verzeichnis alle leeren Dateien und lösche sie. (3 Punkte)