

1.2.5. Von Befehlen zum Skript I

Hausaufgabe - Lösung

1. Recap (15 Punkte)

- a. Welche Texteditoren kann ich nutzen, um innerhalb der Bash eine Datei zu erstellen, zu editieren und zu speichern? (5 Punkte)

Vi(m), Nano

- b. Wie kann ich das folgende Problem lösen? (5 Punkte)

```
$ ./new_script  
/bin/bash: ./new_script: Permission denied
```

Ich muss die Berechtigung zum Ausführen der Datei ändern.
Lösung: `chmod +x ./new_script` (Hier vergebe ich mithilfe des `chmod`-Befehls die Ausführrechte für die Datei)

- c. Was ist ein Interpreter und inwiefern brauchen wir ihn in unserem Bash Skript? (5 Punkte)

Interpreter übersetzt den Programmiercode in Maschinensprache, damit es dann vom Computer umgesetzt werden kann.
Wir brauchen am Anfang der Datei eine Angabe des Befehlsinterpreters, d.h. womit das Skript ausgeführt werden soll (Pfadangabe am Anfang der Datei → Shebang)

2. Bash Skript I: Persönliche Begrüßung (30 Punkte)

Schreibe ein Bash-Skript, das den Benutzer begrüßt. Das Skript soll den Benutzernamen in einer Begrüßungsnachricht ausgeben. (Schreibe nötige Befehle in die leeren Boxen unter die jeweilige Teilaufgabe. Am Ende der Aufgabe gib bitte außerdem ein Screenshot von dem Bash-Skript ab und/oder hänge das Skript in der Abgabe mit an 😊)

Schritte:

1. Öffne Vim und erstelle eine neue Datei namens `greeting.sh`. (5 Punkte)

vim greeting.sh

2. Füge die Shebang-Zeile `#!/bin/bash` hinzu. (5 Punkte)

3. Erstelle eine Variable namens `name` und weise ihr den Wert zu, der in der Umgebungsvariablen `$USER` gespeichert ist. (5 Punkte)

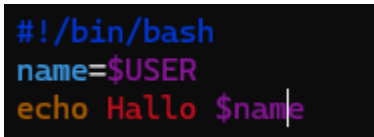
4. Verwende `echo`, um eine Begrüßungsnachricht anzuzeigen, die den Namen des Benutzers enthält. (5 Punkte)

5. Speichere das Skript und mache es ausführbar. (5 Punkte)

6. **Bonus:** Verwende Variablen, um die Begrüßungsnachricht weiter zu personalisieren, z. B. mit dem aktuellen Datum. (5 Punkte)

7. **Bonus:** Schreibe das Skript so um, dass in der Variablen `name` nicht der `$USER` gespeichert wird, sondern das übergebene Argument. Später soll unser Skript dann mit dem Befehl `./greeting.sh Max` aufgerufen werden. (5 Punkte)

8. **Screenshot-Abgabe** (5 Punkte):



3. Bash Skript II: Taschenrechner (30 Punkte)

Schreibe ein Bash-Skript, das zwei Zahlen nimmt und dann die Summe der beiden Zahlen berechnet und ausgibt. (Schreibe nötige Befehle in die leeren Boxen unter die jeweilige Teilaufgabe. Am Ende der Aufgabe gib bitte außerdem ein Screenshot von dem Bash-Skript ab und/oder hänge das Skript in der Abgabe mit an 😊)

Schritte:

1. Erstelle eine neue Datei namens `add_numbers.sh` in Vim. (5 Punkte)

```
touch add_numbers.sh | vim add_numbers.sh
```

2. Füge die Shebang-Zeile hinzu. (5 Punkte)

```
#!/bin/bash
```

3. Verwende Variablen, um die beiden Zahlen zu speichern. Weise ihr die Werte 1 und 2 zu. (5 Punkte)

```
num1=1  
num2=2
```

4. Berechne die Summe der beiden Zahlen und speichere das Ergebnis in einer neuen Variable. (5 Punkte)

```
summe=$((num1+num2))
```

5. Gib das Ergebnis auf der Konsole aus. (5 Punkte)

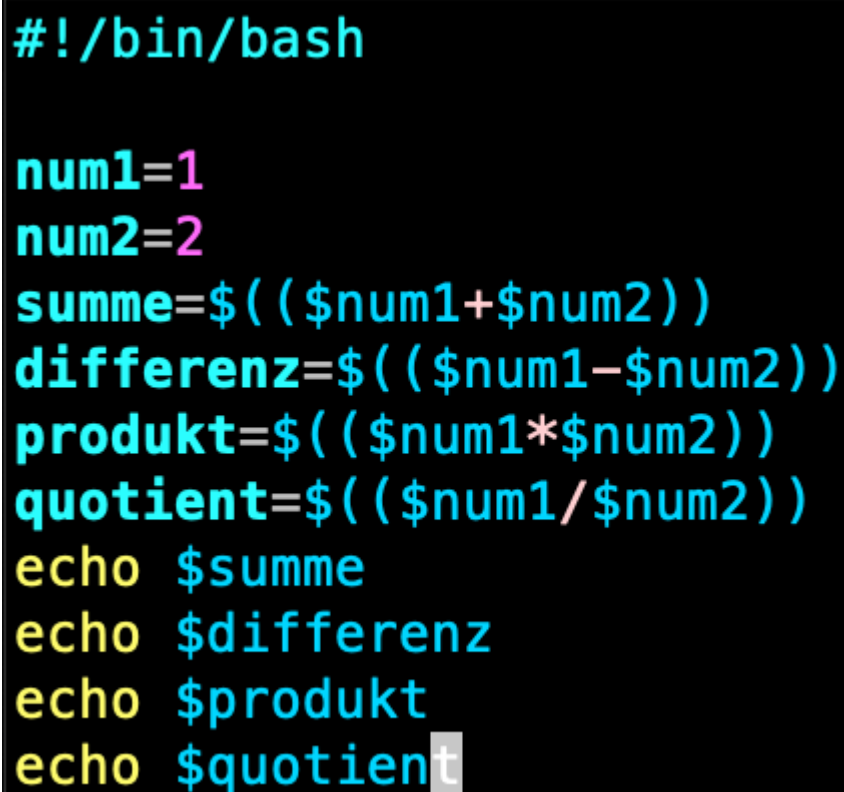
```
echo $summe
```

6. **Bonus:** Erweitere das Skript, um auch die Differenz, das Produkt und den Quotienten der beiden Zahlen auszugeben. (5 Punkte)

```
#!/bin/bash  
  
num1=1  
num2=2  
summe=$((num1+num2))
```

```
differenz=$(( $num1-$num2 ))
produkt=$(( $num1*$num2 ))
quotient=$(( $num1/$num2 ))
echo $summe
echo $differenz
echo $produkt
echo $quotient
```

7. **Screenshot-Abgabe** (5 Punkte):



```
#!/bin/bash

num1=1
num2=2
summe=$(( $num1+$num2 ))
differenz=$(( $num1-$num2 ))
produkt=$(( $num1*$num2 ))
quotient=$(( $num1/$num2 ))
echo $summe
echo $differenz
echo $produkt
echo $quotient
```

4. **Grep und Curl** (25 Punkte):

Curle dir die Adresse <https://marcolindner.io/api/access/index.txt> und pipe dir die Ausgabe in eine Datei mit dem Namen logs.txt.

- a. Suche nach allen Zeilen, die den Statuscode 200 enthalten: (5 Punkte)

```
grep ' 200 ' logs.txt
```



- b. Finde alle Anfragen, die von einem iPhone gesendet wurden: (5 Punkte)

```
grep iPhone logs.txt
```

- c. Suche nach allen Zeilen, die das Wort „POST“ enthalten, um alle POST-Anfragen zu finden: (5 Punkte)

```
grep POST logs.txt
```

- d. Suche nach allen Anfragen, die den Pfad /admin enthalten: (5 Punkte)

```
grep "/admin" logs.txt
```

- e. Finde alle Zugriffe, die zwischen 14:14 und 14:24 Uhr erfolgt sind: (5 Punkte)

```
grep -E "14:1[4-9]:??|14:2[1-3]:??" logs.txt
```

- f. **Bonus:** Extrahiere alle Einträge, die auf Ressourcen mit der Endung .css zugreifen: (5 Punkte)

```
grep -E "\.css" logs.txt
```