ZWEITE ÜBUNG

ZUR EINFÜHRUNG IN DIE PROGRAMMIERUNG FÜR COMPUTERLINGUISTEN

LEONIE WEIßWEILER

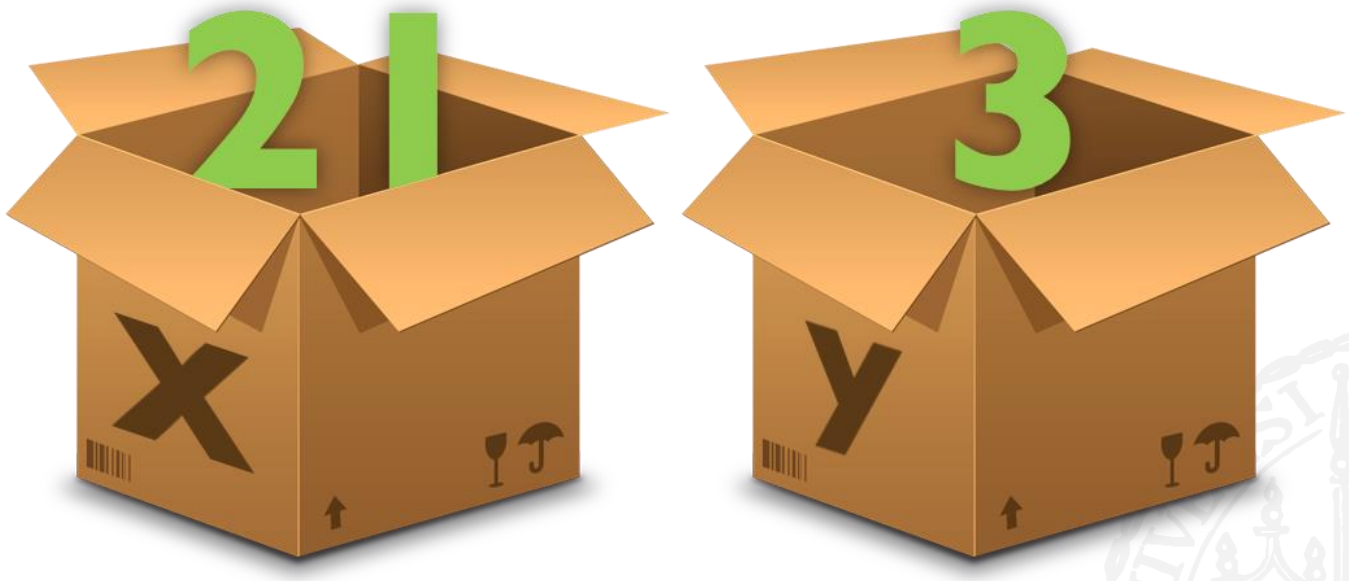
4.11.2016

ABSTIMMEN!

Semesterticket München

https://abstimmung.semesterticket-muenchen.de/

WIEDERHOLUNG - VARIABLEN



WIEDERHOLUNG - VARIABLEN

```
Clemens = ‘Simone’
Simone = ‘Felix’
Annabelle = Clemens + Simone
print(Felix)
```

ERROR: Die Variable Felix existiert nicht

```
print(Annabelle)
```

SimoneFelix

WIEDERHOLUNG - DATENTYPEN

Datentyp	Inhalt	Operatoren
`integer`	Ganze Zahl	`+ - * / %` `> < <= >= ==`
`float`	Kommazahl	
`string`	Text	`+ * == < > <= >=`
`boolean`	Wahrheitswert (**`True`** oder **`False`**)	`&& || ! ==`

WIEDERHOLUNG - DATENTYPEN

```
x = 4
y = 5.0
z = ‘6’
a = ‘7’
print(x+y)
```

9.0

```
print(y+z)
```

FEHLER: Float und String können nicht addiert werden

```
print(z+a)
```

67

UNIX - PFADE

- Ein Pfad gibt den Ort einer Datei oder eines Ordners an.
- Ein Pfad kann **absolut** oder **relativ** sein.
- `cd ../privat/sicherung`
- `kate hello.py`
- `kate ./hello.py`
- `python3 /home/weissweiler/programme/weltherrschaft.py`

UNIX - PFADE

```
Leonie@Laptop:Seminar $ python3
../programme/lmu.py
```

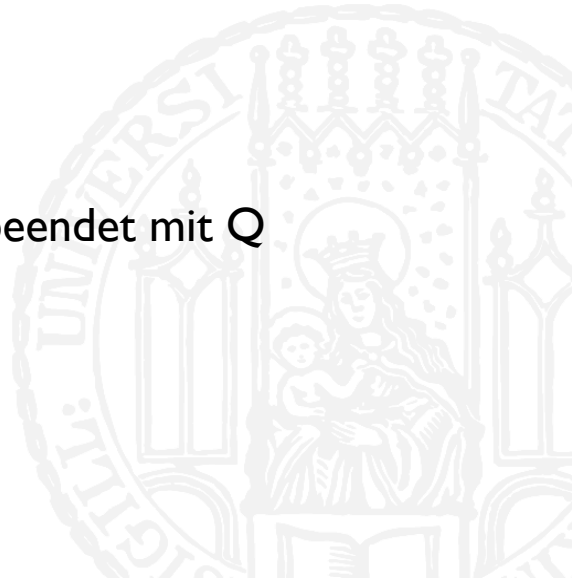
```
Leonie@Laptop:Seminar $ pwd
/home/leonie/Seminar
Leonie@Laptop:Seminar $ cd ..
Leonie@Laptop:~ $ pwd
/home/leonie/
Leonie@Laptop:~ $ cd programme
Leonie@Laptop:programme $ pwd
/home/leonie/programme
Leonie@Laptop:Seminar $ ls
lmu.py
Leonie@Laptop:Seminar $ python3 lmu.py
```

UNIX - PFADE

- Relative Pfade starten beim aktuellen Verzeichnis
- Absolute Pfade starten im Root-Verzeichnis (“ganz oben”) und beginnen deswegen mit einem **/**
- Wir befinden uns im Ordner `/home/leonie/seminar`
- Relativer Pfad: `../privat/lmu.py`
- Absoluter Pfad: `/home/leonie/seminar/../privat/lmu.py = /home/leonie/privat/lmu.py`

UNIX - MAN PAGES

- man steht für "Manual", also "Anleitung"
- Man kann zu jedem Befehl eine Anleitung aufrufen, in der Befehl und Optionen erklärt werden
- `man ls`
- `man pwd`
- `man python3`
- In einer man page bewegt man sich mit den Pfeiltasten und beendet mit Q



UNIX - WILDCARDS

- Beim Angeben von Dateinamen kann man Platzhalter, sog. Wildcards verwenden
- * steht für beliebig viele beliebige Zeichen
- ? steht für genau ein beliebiges Zeichen

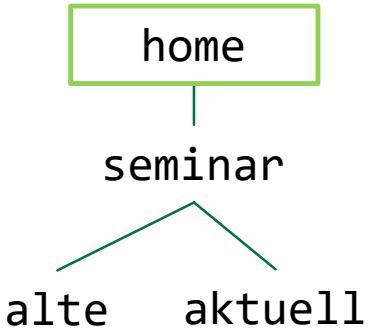
UNIX - WILDCARDS

```
Leonie@Laptop:Seminar $ ls
Hallo.py  Hello.py  Halloho.txt

Leonie@Laptop:Seminar $ ls *
Hallo.py  Hello.py  Halloho.txt

Leonie@Laptop:Seminar $ ls *.py
Hallo.py  Hello.py

Leonie@Laptop:Seminar $ ls ?????.py
Hallo.py  Hello.py
```

```
Leonie@Laptop:Seminar $ ls H?llo.py
Hallo.py  Hello.py

Leonie@Laptop:Seminar $ ls Hall*
Hallo.py  Halloho.txt

Leonie@Laptop:Seminar $ ls H?llo*
Hallo.py  Hello.py Halloho.txt

Leonie@Laptop:Seminar $ ls a*
```

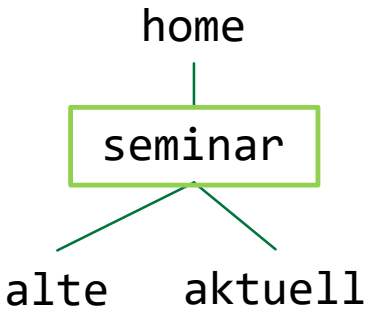
2.1 A

- Erstellen Sie in Ihrem Homeverzeichnis ein Verzeichnis "seminar" mit den Unterverzeichnissen "alte" und "aktuell“
- `cd`
- `mkdir seminar`
- `cd seminar`
- `mkdir alte`
- `mkdir aktuell`



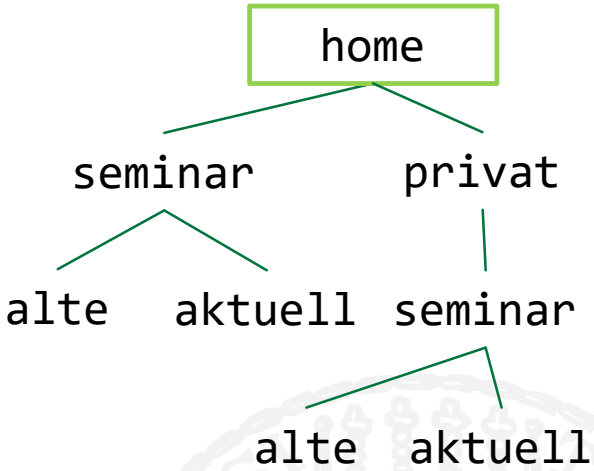
2.1 B

- Erstellen Sie weiterhin im Homeverzeichnis ein Verzeichnis privat. Kopieren Sie das Verzeichnis seminar mit seinen Unterverzeichnissen in privat.
- `cd`



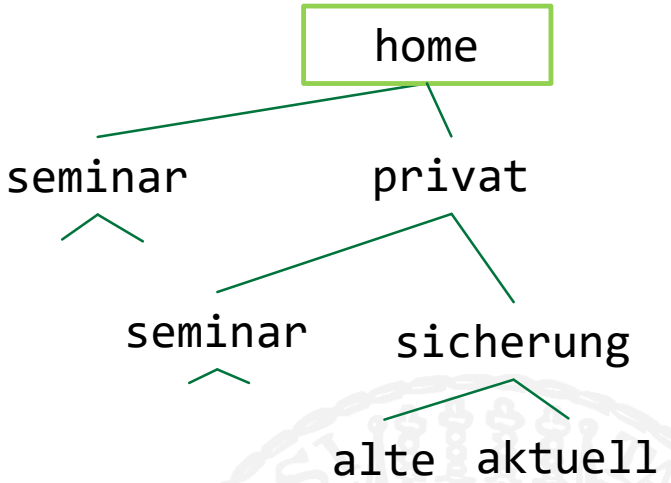
2.1 B

- Erstellen Sie weiterhin im Homeverzeichnis ein Verzeichnis privat. Kopieren Sie das Verzeichnis seminar mit seinen Unterverzeichnissen in privat.
- `cd`
- `mkdir privat`
- `cp –r seminar privat`



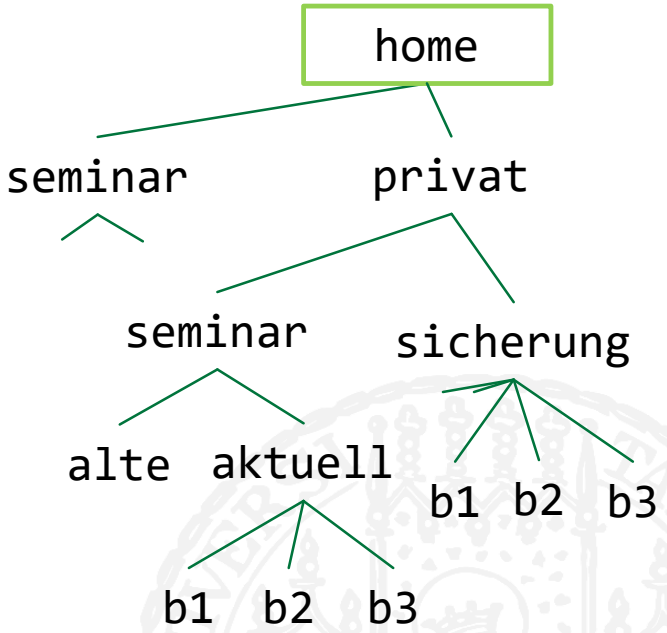
2.1 C

- Kopieren Sie das Verzeichnis seminar mit seinen Unterverzeichnissen in das Verzeichnis sicherung im Verzeichnis privat, ohne das Verzeichnis sicherung vorher zu erstellen. Vergleichen Sie die resultate von b) und c)
- `cd`
- `cp -r seminar privat/sicherung`



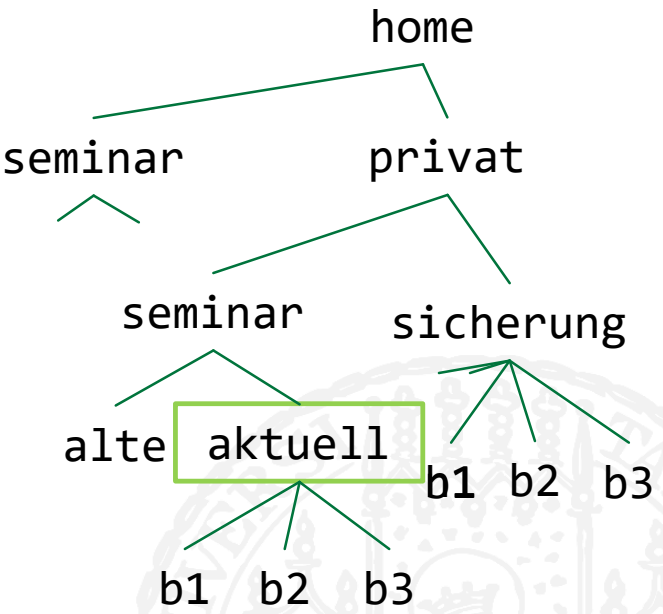
2.2

- Erstellen Sie in privat/seminar/aktuell die Dateien b1,b2,b3. (Hinweis Unix Befehl touch oder ein Texteditor). Kopieren Sie b1,b2 und b3 in privat/sicherung
- `cd ~/privat/seminar/aktuell`
- `touch b1 b2 b3`
- `cp * ../../sicherung`



2.3

- Benennen Sie in privat/sicherung die Datei b1 um in die Datei n1
- `cd ../../sicherung`
- `mv b1 n1`



2.4

- Lassen Sie sich alle zweistelligen Kommandos im Verzeichnis /bin auflisten.

```
Leonie@Laptop:Seminar $ cd /bin
Leonie@Laptop:/bin $ ls ??
cp  dd  df  ed  ip  ln  ls  mt  mv  nc  ps  rm  sh  ss  su
```

2.5

- Wo finden sie das Programm python3 auf der Platte ihres Rechners?

```
Leonie@Laptop:Seminar $ which python3

/usr/bin/python3

Leonie@Laptop:Seminar $ whereis python3

python3: /usr/bin/python3 /usr/bin/python3.4 /usr/bin/python3.4m /etc/python3
/etc/python3.4 /usr/lib/python3 /usr/lib/python3.4 /usr/bin/X11/python3
/usr/bin/X11/python3.4 /usr/bin/X11/python3.4m /usr/local/lib/python3.4
/usr/share/python3 /usr/share/man/man1/python3.1.gz
```

2.6

- Wie lautet der Befehl, der alle Dateien, auch die verborgenen Dateien in ihrem Homedirectory auflistet?

```
Leonie@Laptop:~ $ ls -a
.    .bash_history  .bashrc       .cpan     seminar   .ssh  .viminfo
..   .bash_logout   .bashrc.save  .lesshst  .profile  .vim  .viminfo.tmp
```

2.7

- Wieviel Speicher belegt Ihr Homeverzeichnis auf der Platte?

```
Leonie@Laptop:Seminar $ cd
Leonie@Laptop:~ $ du -sh
71M     .
```

2.8

- Wieviel Gigabyte umfaßt die Platte mit Ihrem Homeverzeichnis?

```
Leonie@Laptop:~ $ df -h
Filesystem      Size  Used Avail Use% Mounted on
rootfs          119G   81G   39G  68% /
tmpfs           119G   81G   39G  68% /run
```

2.9

- Wie lautet der Befehl, der die Namen und Inhalte aller Verzeichnisse und aller Unterverzeichnisse ihres Homedirectories auflistet?

```
Leonie@Laptop:Seminar $ ls -R

.:
aktuell  alte

./aktuell:
datei.txt

./alte:
meine_datei.txt
```

2.9

- Wie lautet der Befehl, der die Namen und Inhalte aller Verzeichnisse und aller Unterverzeichnisse ihres Homedirectories auflistet?

```
Leonie@Laptop:Seminar $ tree

.
├── aktuell
│   └── datei.txt
└── alte
    └── meine_datei.txt

2 directories, 2 files
```

2.10

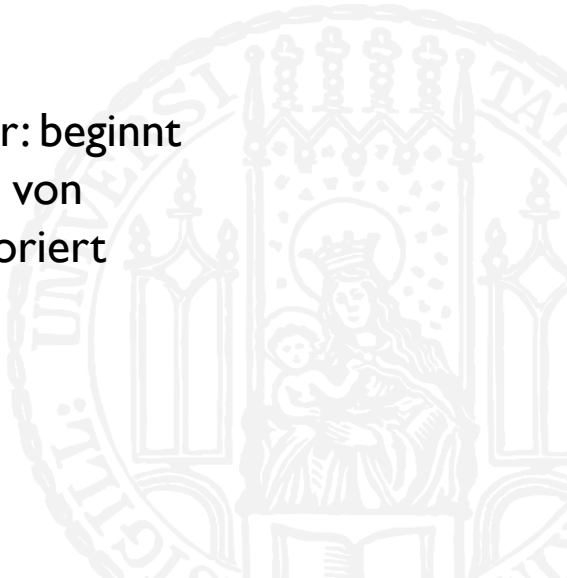
```
#!/usr/bin/python3
# Aufgabe 2-10
# Autorin: Leonie Weißweiler
print ('Hallo')
zahl1 = input('Geben Sie eine Zahl ein\n')
zahl2 = input('Geben Sie noch eine Zahl ein\n')
if zahl1 == zahl2:
    print ('Ja:', zahl1, ',', zahl2)
else:
    print ('Nein:', zahl1,',',zahl2)
```

2.10 - PROGRAMMANFANG

```
#!/usr/bin/python3
# Aufgabe 2-10
# Autorin: Leonie Weißweiler
```

Shebang: gibt an, wo der Python-Interpreter auf der Festplatte liegt

Kommentar: beginnt mit #, wird von Python ignoriert



2.10 - PRINT

```
print ('Hallo')
```

- `print` gibt den Text in den Klammern und Anführungszeichen an, UND
- ein Newline-Zeichen \n, damit bei
```
print ('Hallo')
print ('Welt')
```
- nicht `HalloWelt`, sondern
```
Hallo
Welt
```
rauskommt
- falls man das nicht möchte, muss man stattdessen print(‚Hallo‘, end=‘) schreiben
- end ist by default auf newline gesetzt, damit ändern wir end auf den leeren String

2.10 - INPUT

```
zahl1 = input('Geben Sie eine Zahl ein\n')
zahl2 = input('Geben Sie noch eine Zahl ein\n')
```

- **`input`** gibt als String das zurück, was der Benutzer eingegeben hat, bis er ein Newline eingibt
- Der String, den input übergeben bekommt, wird dem Benutzer als Eingabeaufforderung angezeigt
- anders als **`print`** druckt **`input`** danach kein newline aus, man muss es also selbst hinzufügen, damit der der Nutzer in einer neuen Zeile seine Eingabe macht

2.10 – IF ELSE

```
if zahl1 == zahl2 :
    print ('Ja:', zahl1, ',', zahl2)
else:
    print ('Nein:', zahl1,',',zahl2)
```

- zahl1 == zahl2 wird evaluiert
- Falls True heraus kommt, wird der if-Block ausgeführt
- Falls False herauskommt, wird der else-Block ausgeführt

print werden mehrere Argumente übergeben, die dadurch alle nacheinander ausgegeben warden, getrennt durch je ein Leerzeichen

2.10 - NOCHMAL

```
#!/usr/bin/python3
# Aufgabe 2-10
# Autorin: Leonie Weißweiler
print ('Hallo')
zahl1 = input('Geben Sie eine Zahl ein\n')
zahl2 = input('Geben Sie noch eine Zahl ein\n')
if zahl1 == zahl2:
    print ('Ja:', zahl1, ',', zahl2)
else:
    print ('Nein:', zahl1,',',zahl2)
```

2.10 – WER HAT DIE SCHUMMELEI ENTDECKT?

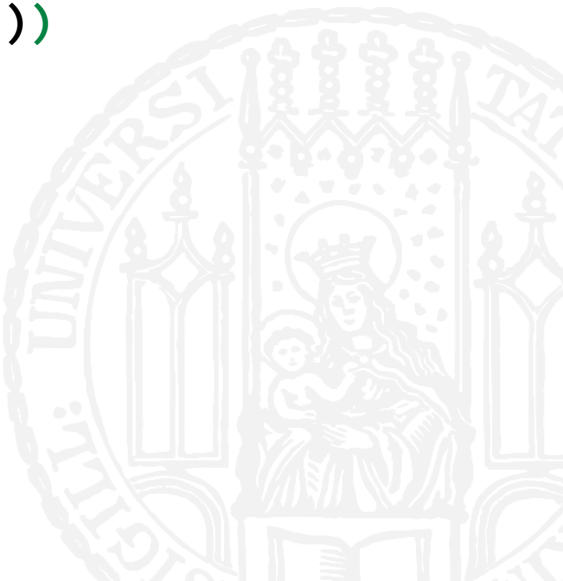
```
#!/usr/bin/python3
# Aufgabe 2-10
# Autorin: Leonie Weißweiler
print ('Hallo')
zahl1 = input('Geben Sie eine Zahl ein\n')
zahl2 = input('Geben Sie noch eine Zahl ein\n')
if zahl1 == zahl2:
    print ('Ja:', zahl1, ',', zahl2)
else:
    print ('Nein:', zahl1,',',zahl2)
```

Tipp: Es hat mit Datentypen zu tun

2.10 – BESSER

```
#!/usr/bin/python3
# Aufgabe 2-10
# Autorin: Leonie Weißweiler
print ('Hallo')
zahl1 = int(input('Geben Sie eine Zahl ein\n'))
zahl2 = int(input('Geben Sie noch eine Zahl ein\n'))
if zahl1 == zahl2:
    print ('Ja:', zahl1, ',', zahl2)
else:
    print ('Nein:', zahl1,',',zahl2)
```

Tipp: Es hatte mit Datentypen zu tun



QUIZ