Basic Tools for NLP

Rishu Kumar, Iuliia Zaitova, Chris Hyek

Saarland University

11.05.2023

Topics for today:

- Bash script organization
- Control flow statements (if-else, loops)
- Regex
- Plugins for Unix shells

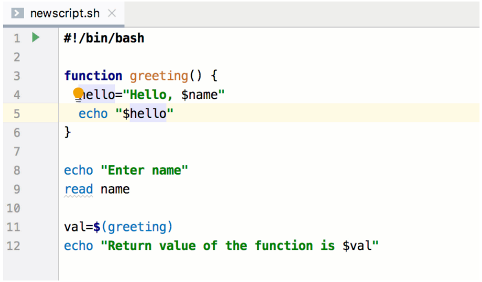
Bash scripts

- Bash scripts are plain text files with a ".sh" extension, and they are executed by the Bash shell.

Command-line interface

```
zaira@Zaira:~$
```

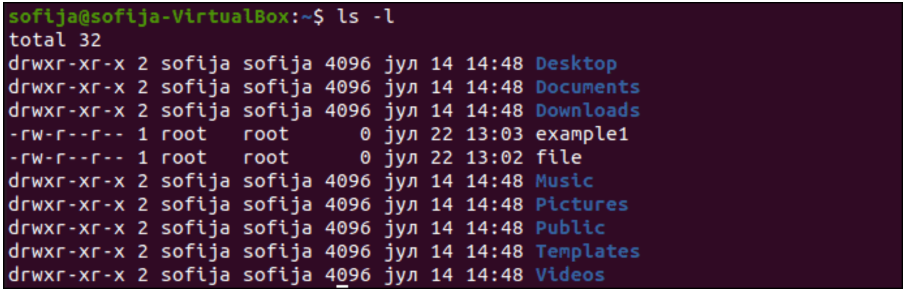
Bash scripts



newscript.sh

```bash
#!/bin/bash

function greeting() {
  hello="Hello, $name"
  echo "$hello"
}

echo "Enter name"
read name

val=$(greeting)
echo "Return value of the function is $val"
```

- Bash scripts are plain text files with a ".sh" extension, and they are executed by the Bash shell.

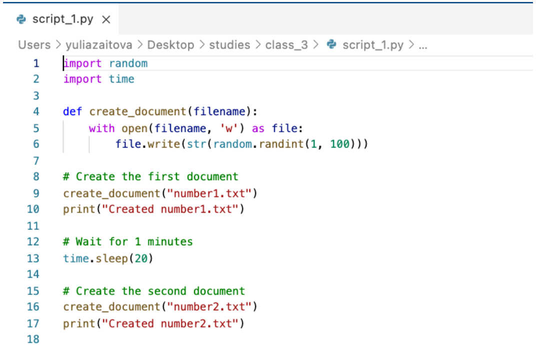


```
sofija@sofija-VirtualBox:~$ ls -l
total 32
drwxr-xr-x 2 sofija sofija 4096 јун 14 14:48 Desktop
drwxr-xr-x 2 sofija sofija 4096 јун 14 14:48 Documents
drwxr-xr-x 2 sofija sofija 4096 јун 14 14:48 Downloads
-rw-r--r-- 1 root   root      0 јун 22 13:03 example1
-rw-r--r-- 1 root   root      0 јун 22 13:02 file
drwxr-xr-x 2 sofija sofija 4096 јун 14 14:48 Music
drwxr-xr-x 2 sofija sofija 4096 јун 14 14:48 Pictures
drwxr-xr-x 2 sofija sofija 4096 јун 14 14:48 Public
drwxr-xr-x 2 sofija sofija 4096 јун 14 14:48 Templates
drwxr-xr-x 2 sofija sofija 4096 јун 14 14:48 Videos
```

Warm up!

- Download `class_3.zip`
- Unzip it `unzip class_3.zip`
- Open and run scripts `script_1.py` and `script_2.py`
- (`python script_1.py`)

Script 1

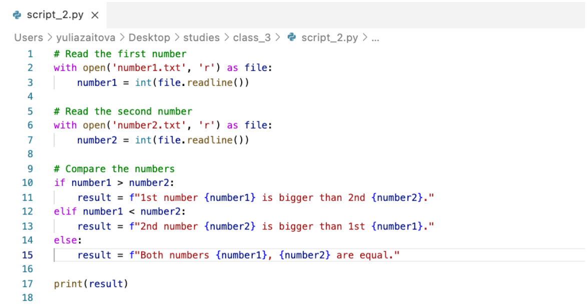


script_1.py ×

Users > yuliazaitova > Desktop > studies > class_3 > script_1.py > ...

```python
import random
import time

def create_document(filename):
    with open(filename, 'w') as file:
        file.write(str(random.randint(1, 100)))

# Create the first document
create_document("number1.txt")
print("Created number1.txt")

# Wait for 1 minutes
time.sleep(20)

# Create the second document
create_document("number2.txt")
print("Created number2.txt")

```

Script 2

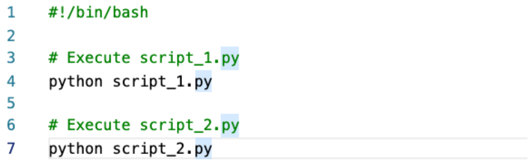


script_2.py ×

Users > yuliazaitova > Desktop > studies > class_3 > script_2.py > ...

```python
# Read the first number
with open('number1.txt', 'r') as file:
    number1 = int(file.readline())

# Read the second number
with open('number2.txt', 'r') as file:
    number2 = int(file.readline())

# Compare the numbers
if number1 > number2:
    result = f"1st number {number1} is bigger than 2nd {number2}."
elif number1 < number2:
    result = f"2nd number {number2} is bigger than 1st {number1}."
else:
    result = f"Both numbers {number1}, {number2} are equal."

print(result)

```

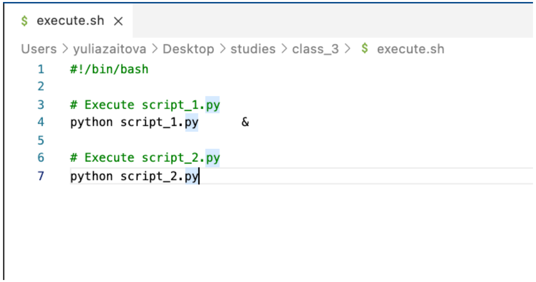
- Open your CLI
- `chmod 750 execute.sh`
  `./execute.sh`



```bash
#!/bin/bash

# Execute script_1.py
python script_1.py

# Execute script_2.py
python script_2.py
```

And... we don't need to wait for the execution of the first program :)

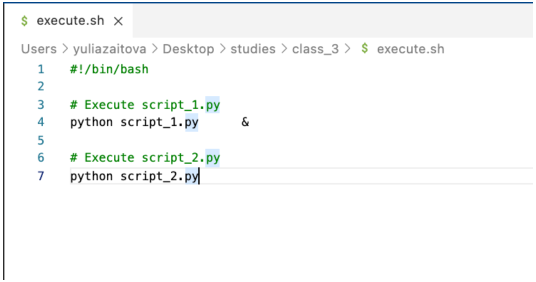
- Add &
- Execute again `./execute.sh`


$ execute.sh ×
Users > yuliazaitova > Desktop > studies > class_3 > $ execute.sh

```
#!/bin/bash

# Execute script_1.py
python script_1.py      &

# Execute script_2.py
python script_2.py
```

What happens?

- Add &
- Execute again `./execute.sh`


$ execute.sh ×

Users > yuliazaitova > Desktop > studies > class_3 > $ execute.sh

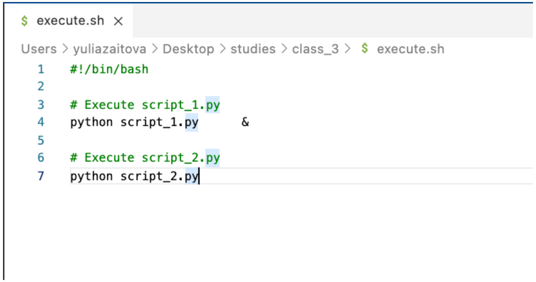
```
#!/bin/bash

# Execute script_1.py
python script_1.py      &

# Execute script_2.py
python script_2.py
```

What happens?

& – code executed simultaneously

Shebang

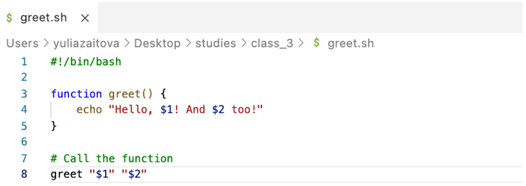
- It's a line at the beginning, starting with #!/ followed by the interpreter's path.
  Why?
- Without shebang the shell relies on the system's default behavior for command execution (can be bad)
- Indicator to other developers and users how the script is intended to be executed


$ execute.sh ×
Users > yuliazaitova > Desktop > studies > class_3 > $ execute.sh

```
#!/bin/bash

# Execute script_1.py
python script_1.py      &

# Execute script_2.py
python script_2.py
```

Passing Arguments to a Script

- Bash scripts can accept arguments from the command line.
- They can be accessed within the script using special variables.
- Open `./greet.sh` to see :)



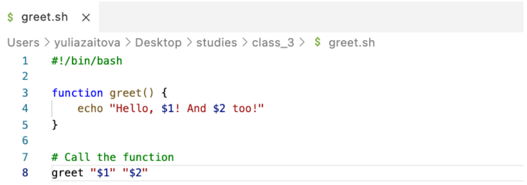
greet.sh ×

Users > yuliazaitova > Desktop > studies > class_3 > greet.sh

```bash
#!/bin/bash

function greet() {
    echo "Hello, $1! And $2 too!"
}

# Call the function
greet "$1" "$2"
```

Passing Arguments to a Script

- Let's run `./greet.sh`
- `chmod 750 greet.sh`
  `./greet.sh "name1" "name2"`
- Try without quotes
- Try one name
- Try three names



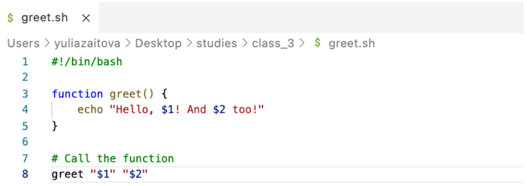
greet.sh

Users > yuliazaitova > Desktop > studies > class_3 > greet.sh

```bash
#!/bin/bash

function greet() {
    echo "Hello, $1! And $2 too!"
}

# Call the function
greet "$1" "$2"
```

Passing Arguments to a Script

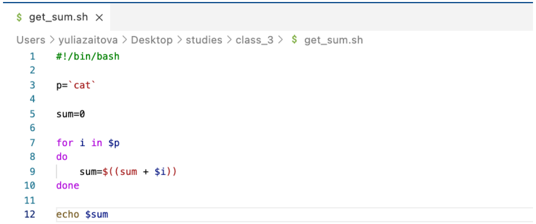
- Run `username=$(whoami)`
- Try one name
- Try three names



$ greet.sh ×
Users > yuliazaitova > Desktop > studies > class_3 > $ greet.sh

```
#!/bin/bash

function greet() {
    echo "Hello, $1! And $2 too!"
}

# Call the function
greet "$1" "$2"
```

- Open `./get_sum.sh`
- Try to understand it
- `chmod 750 get_sum.sh`
  `echo 1 3 4 5| ./get_sum.sh`

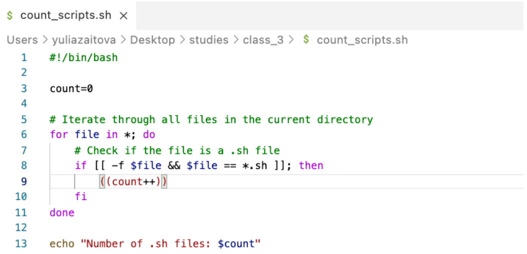


get_sum.sh ×

Users > yuliazaitova > Desktop > studies > class_3 > get_sum.sh

```bash
#!/bin/bash

p=`cat`

sum=0

for i in $p
do
    sum=$((sum + $i))
done

echo $sum
```

- Open `./count_scripts.sh`
- Try to understand it
- `chmod 750 get_sum.sh`
  `echo 1 3 4 5| ./get_sum.sh`



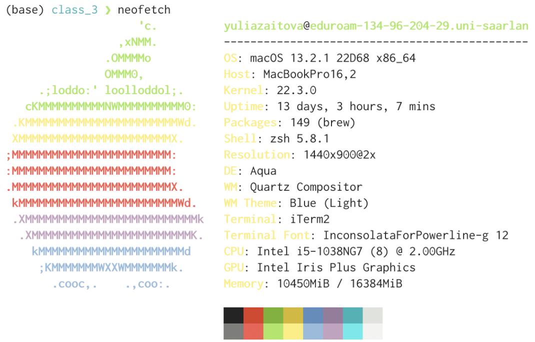
count_scripts.sh

Users > yuliazaitova > Desktop > studies > class_3 > count_scripts.sh

```bash
#!/bin/bash

count=0

# Iterate through all files in the current directory
for file in *; do
    # Check if the file is a .sh file
    if [[ -f $file && $file == *.sh ]]; then
        ((count++))
    fi
done

echo "Number of .sh files: $count"
```

Some nice tools

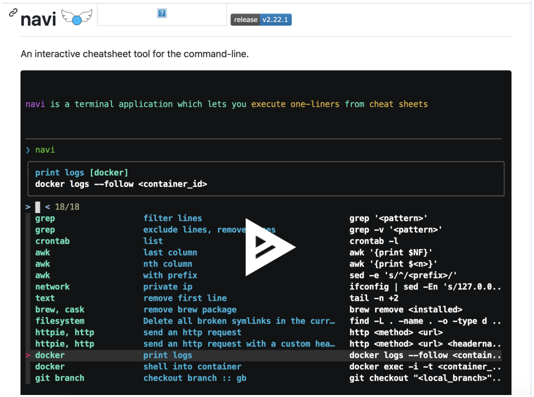
- Neofetch
- `sudo apt install neofetch`
  `brew install neofetch`



```
(base) class_3 ❯ neofetch
                    'c.          yuliazaitova@eduroam-134-96-204-29.uni-saarlan
                 ,xNMM.          ----------------------------------------------
               .OMMMMo           OS: macOS 13.2.1 22D68 x86_64
               OMMM0,            Host: MacBookPro16,2
     .;loddo:' loolloddol;.      Kernel: 22.3.0
   cKMMMMMMMMMMNWMMMMMMMMMM0:    Uptime: 13 days, 3 hours, 7 mins
 .KMMMMMMMMMMMMMMMMMMMMMMMWd.    Packages: 149 (brew)
 XMMMMMMMMMMMMMMMMMMMMMMMX.      Shell: zsh 5.8.1
;MMMMMMMMMMMMMMMMMMMMMMMM:       Resolution: 1440x900@2x
:MMMMMMMMMMMMMMMMMMMMMMMM:       DE: Aqua
.MMMMMMMMMMMMMMMMMMMMMMMMX.      WM: Quartz Compositor
 kMMMMMMMMMMMMMMMMMMMMMMMMWd.    WM Theme: Blue (Light)
 .XMMMMMMMMMMMMMMMMMMMMMMMMMMk   Terminal: iTerm2
  .XMMMMMMMMMMMMMMMMMMMMMMMMK.   Terminal Font: InconsolataForPowerline-g 12
    kMMMMMMMMMMMMMMMMMMMMMMd     CPU: Intel i5-1038NG7 (8) @ 2.00GHz
     ;KMMMMMMMWXXWMMMMMMMk.      GPU: Intel Iris Plus Graphics
       .cooc,.    .,coo:.        Memory: 10450MiB / 16384MiB
```

Some nice tools

- Navi – cheatsheets for CLI commands



navi release v2.22.1

An interactive cheatsheet tool for the command-line.

```
navi is a terminal application which lets you execute one-liners from cheat sheets

❯ navi

 print logs [docker]
 docker logs --follow <container_id>

> ▌ < 18/18
  grep                       filter lines                           grep '<pattern>'
  grep                       exclude lines, remove   es             grep -v '<pattern>'
  crontab                    list                                   crontab -l
  awk                        last column                            awk '{print $NF}'
  awk                        nth column                             awk '{print $<n>}'
  awk                        with prefix                            sed -e 's/^/<prefix>/'
  network                    private ip                             ifconfig | sed -En 's/127.0.0..
  text                       remove first line                      tail -n +2
  brew, cask                 remove brew package                    brew remove <installed>
  filesystem                 Delete all broken symlinks in the curr…  find -L . -name . -o -type d ..
  httpie, http               send an http request                   http <method> <url>
  httpie, http               send an http request with a custom hea…  http <method> <url> <headerna..
> docker                     print logs                             docker logs --follow <contain..
  docker                     shell into container                   docker exec -i -t <container_..
  git branch                 checkout branch :: gb                  git checkout "<local_branch>"..
```

Passing Arguments to a Script To quickly go to a specific directory

- open `~/.bashrc` (or `~/.zshrc`)
- `pwd`
- `alias class_3="<your path>"`
- `source ~/.bashrc`
- `cd $class_3`