PROGRAMMING IN PYTHON I

Installation, Operating System, and Terminal

Michael Widrich
Institute for Machine Learning

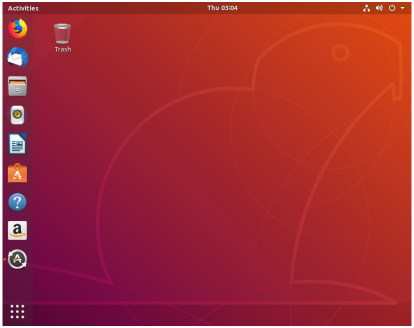
JOHANNES KEPLER
UNIVERSITY LINZ

Institute for
Machine Learning

EXCURSION: OPERATING SYSTEM (OS)

The Operating System

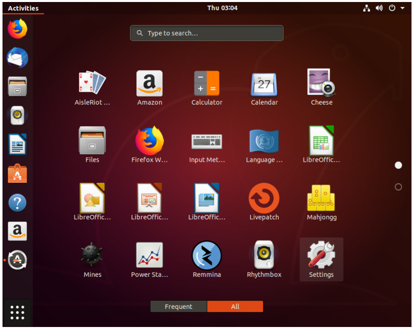
- Your Operating System (OS) is a program running on your machine
  - Linux (e.g. Ubuntu), MacOS, Windows, . . .
  - Examples will be for `Ubuntu 18.04`



Ubuntu desktop in one of the (many) Ubuntu flavors

Programs and processes (1)

- You can view (most of) the programs you install as plug-ins for your OS



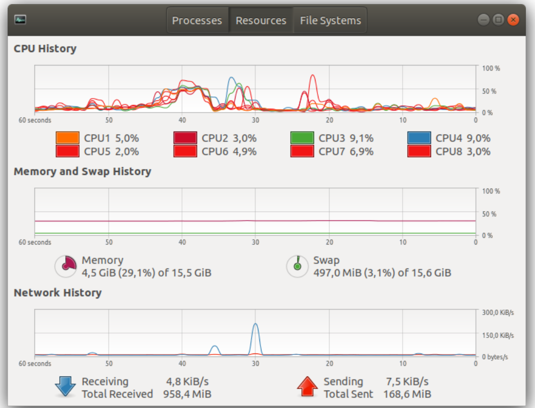
Part of programs installed on standard Ubuntu 18.04

Programs and processes (2)

- Your OS manages (most of) the other programs that you install
  - It schedules when/how long a program and its processes can use the CPU
  - It abstracts from your specific hardware using drivers (drivers are programs that provide a standard interface to hardware components)
- Paths to installed programs are stored in environment variables
  - The environment variable *PYTHONPATH* is usually used for setting paths to Python packages. If you run into package-errors, check this variable.

Programs and processes (3)

- The System Monitor or Task Manager is one of the tools to view some of the OS management



CPU History

🟧 CPU1 5,0%	🟥 CPU2 3,0%	🟩 CPU3 9,1%	🟦 CPU4 9,0%
🟥 CPU5 2,0%	🟥 CPU6 4,9%	🟥 CPU7 6,9%	🟥 CPU8 3,0%

Memory and Swap History

Memory
4,5 GiB (29,1%) of 15,5 GiB

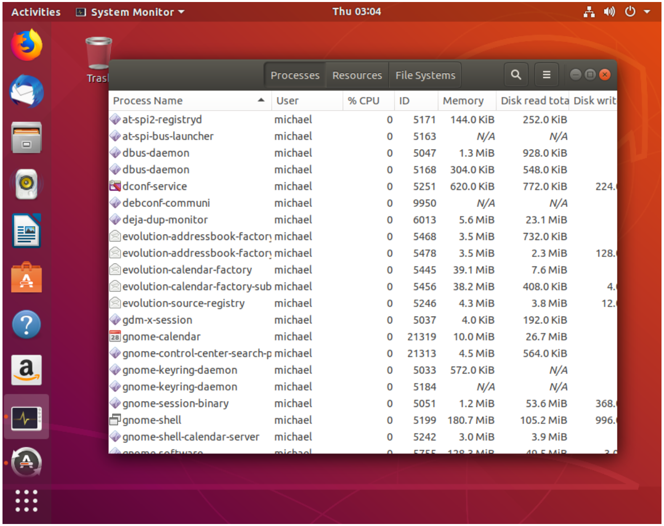
Swap
497,0 MiB (3,1%) of 15,6 GiB

Network History

Receiving	4,8 KiB/s	Sending	7,5 KiB/s
Total Received	958,4 MiB	Total Sent	168,6 MiB

System Monitor shows the current hardware utilization

Programs and processes (4)



Process Name	User	% CPU	ID	Memory	Disk read tota	Disk writ
at-spi2-registryd	michael	0	5171	144.0 KiB	252.0 KiB	
at-spi-bus-launcher	michael	0	5163	N/A	N/A	
dbus-daemon	michael	0	5047	1.3 MiB	928.0 KiB	
dbus-daemon	michael	0	5168	304.0 KiB	548.0 KiB	
dconf-service	michael	0	5251	620.0 KiB	772.0 KiB	224.
debconf-communi	michael	0	9950	N/A	N/A	
deja-dup-monitor	michael	0	6013	5.6 MiB	23.1 MiB	
evolution-addressbook-factory	michael	0	5468	3.5 MiB	732.0 KiB	
evolution-addressbook-factory	michael	0	5478	3.5 MiB	2.3 MiB	128.
evolution-calendar-factory	michael	0	5445	39.1 MiB	7.6 MiB	
evolution-calendar-factory-sub	michael	0	5456	38.2 MiB	408.0 KiB	4.
evolution-source-registry	michael	0	5246	4.3 MiB	3.8 MiB	12.
gdm-x-session	michael	0	5037	4.0 KiB	192.0 KiB	
gnome-calendar	michael	0	21319	10.0 MiB	26.7 MiB	
gnome-control-center-search-p	michael	0	21313	4.5 MiB	564.0 KiB	
gnome-keyring-daemon	michael	0	5033	572.0 KiB	N/A	
gnome-keyring-daemon	michael	0	5184	N/A	N/A	
gnome-session-binary	michael	0	5051	1.2 MiB	53.6 MiB	368.
gnome-shell	michael	0	5199	180.7 MiB	105.2 MiB	996.
gnome-shell-calendar-server	michael	0	5242	3.0 MiB	3.9 MiB	
gnome-software	michael	0	5755	128.3 MiB	49.5 MiB	3.

System Monitor shows the currently managed processes

The System Terminal (1)

- Some OS and programs provide an abstract Graphical User Interface (GUI) with cursor, desktop, etc.
  - Sometimes comfortable, simpler, visually nicer
  - Additional work (needs to be implemented), not always handy, needs resources for rendering
  - Remote servers and scientific ML programs usually do not provide GUIs

Institute for Machine Learning

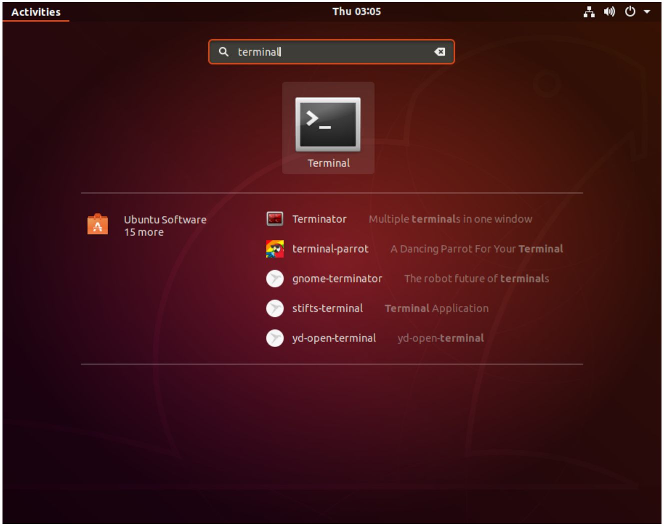
The System Terminal (1)

- Some OS and programs provide an abstract Graphical User Interface (GUI) with cursor, desktop, etc.
  - Sometimes comfortable, simpler, visually nicer
  - Additional work (needs to be implemented), not always handy, needs resources for rendering
  - Remote servers and scientific ML programs usually do not provide GUIs
- You can also use your OS directly via the terminal (also console or command-line)
  - Less additional work for the developer
  - Easier to interface with other programs

The System Terminal (1)

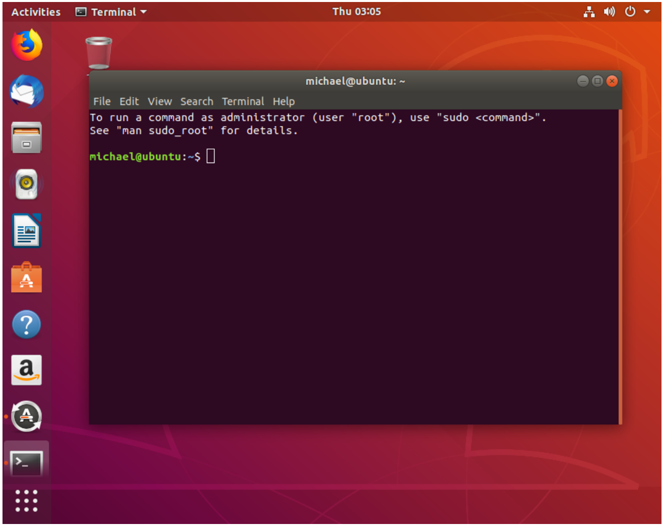
- Some OS and programs provide an abstract Graphical User Interface (GUI) with cursor, desktop, etc.
  - Sometimes comfortable, simpler, visually nicer
  - Additional work (needs to be implemented), not always handy, needs resources for rendering
  - Remote servers and scientific ML programs usually do not provide GUIs
- You can also use your OS directly via the terminal (also console or command-line)
  - Less additional work for the developer
  - Easier to interface with other programs
  - → You'll need the terminal but you'll be fine, don't worry :)

The System Terminal (2)



Starting a terminal in Ubuntu

The System Terminal (2)



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

michael@ubuntu:~$
```

A terminal in Ubuntu

The System Terminal (3)

- The terminal should be in your list of programs (windows: `cmd.exe`, linux: `terminal`)
- Commands are written as text into the terminal and executed by hitting *Enter*
- The *Up* and *Down* keys let you view previously executed commands
- The *Tab* key will auto-complete your command/filepath (hit twice to get a list of suggestions)

Many programs are only available via terminal!

INSTALLING PYTHON

Task: Download and install Python

- Python 64bit, version 3.6 or higher! (on many OS, e.g. Ubuntu 18.04, this is already installed)
- Python official website: `http://www.python.org`
  - Windows: `https://www.python.org/ftp/python/3.7.4/python-3.7.4-amd64.exe`
    Important: Make sure to select that you want to add the Python path to the PATH environment variable!
  - MacOS: `https://www.python.org/ftp/python/3.7.4/python-3.7.4-macosx10.9.pkg`
  - Linux: `https://www.python.org/ftp/python/3.7.4/Python-3.7.4.tgz`
    - For many linux distributions you can use the package manager to install Python
    - Ubuntu (only if you want to have 3.7 instead of 3.6!):
      `sudo apt-get install python3.7`

Python packages (1)

- You can add new functions to your Python installation by installing additional Python packages
- Packages can be installed via pip (package installer for Python)
  - Pip guide:
    `https://docs.python.org/3/installing/index.html`
  - In the terminal you can install a package with the command
    `pip3 install packagename`
    or, depending on your installation,
    `pip install packagename`
  - or, if you lack permissions,
    `pip3 install -U packagename`

Python packages (2)

- pip for specific Python versions

  You can use this line to install packages for e.g. version 3.7:

  ```
  python3.7 -m pip install packagename
  ```

- Under Ubuntu you might have to run the following for versions other than 3.6

  ```
  sudo apt install -y python3-pip
  python3.7 -m pip install pip
  ```

- Some packages require certain operating systems, software, or drivers

→ Python is mostly out-of-the-box platform independent – some packages are not!

Alternative: Anaconda

- Alternatively, you may use Anaconda:
  - Manages your Python installations
  - Allows for different Python versions and setups on one machine
  - If you know what you are doing, you may use Anaconda, otherwise stick with the standard Python installation
  - Download: `https://www.anaconda.com/distribution/`

Python Documentation

- Official documentation:
  `http://www.python.org/doc`
- Official tutorial:
  `https://docs.python.org/3.7/tutorial/index.html`
- A Byte of Python (online tutorial book):
  `http://www.swaroopch.com/notes/python/`
- For experienced programmers:
  `http://www.diveintopython3.net`

OPERATING SYSTEMS IN MACHINE LEARNING

Operating Systems in Machine Learning (1)

- Any OS will do, as long as you can get it to run
- Getting Python and PyCharm to run on different OS is straight-forward

Operating Systems in Machine Learning (1)

- Any OS will do, as long as you can get it to run
- Getting Python and PyCharm to run on different OS is straight-forward
  - $\rightarrow$ if it weren't for some important details. . .

Operating Systems in Machine Learning (2)

- GPU and other hardware optimization
  - ☐ GPU drivers (NVIDIA CUDA + CUDNN) and their interface with packages like PyTorch and Tensorflow is crucial
  - → Setup of these drivers can be tricky for some OS and virtual machines
  - ☐ Differences in multitasking between Windows and Linux
  - → Python does a good job in abstraction but interface of such functions might differ

Institute for Machine Learning

Operating Systems in Machine Learning (2)

- GPU and other hardware optimization
  - GPU drivers (NVIDIA CUDA + CUDNN) and their interface with packages like PyTorch and Tensorflow is crucial
  - → Setup of these drivers can be tricky for some OS and virtual machines
  - Differences in multitasking between Windows and Linux
  - → Python does a good job in abstraction but interface of such functions might differ
- Usage of (GPU) servers
  - Large-scale Machine Learning is done on dedicated severs, which typically run Linux
  - → You need to know how to use a Linux terminal if you want to use such servers

Operating Systems in Machine Learning (3)

- Portability issues (relevant for assignments!)
  - ☐ Python code is as portable as you design it to be
  - ☐ Assignment solutions will be graded on a Linux system
  - → Paths, filenames, etc. are an easy source of portability issues!

Operating Systems in Machine Learning (4)

- We recommend and provide support for Ubuntu 18.04+
  - Free to use
  - Straight-forward installation (`https://moodle.jku.at/jku2015/mod/page/view.php?id=2860614`)
  - NVIDIA driver support
  - Will get you used to Linux
  - You can install it along-side a Windows installation even without partitioning
  - Always backup your data!

Operating Systems in Machine Learning (4)

- We recommend and provide support for Ubuntu 18.04+
  - Free to use
  - Straight-forward installation (`https://moodle.jku.at/jku2015/mod/page/view.php?id=2860614`)
  - NVIDIA driver support
  - Will get you used to Linux
  - You can install it along-side a Windows installation even without partitioning
  - Always backup your data!

In this course you can use whatever OS you want, as long as your assignments are correct!

TASKS AND FIRST STEPS

Task 0: Using the System Terminal (1)

1. Open a system terminal (windows: `cmd.exe`)

   Now you can type commands for your OS. Your current location is your home directory.

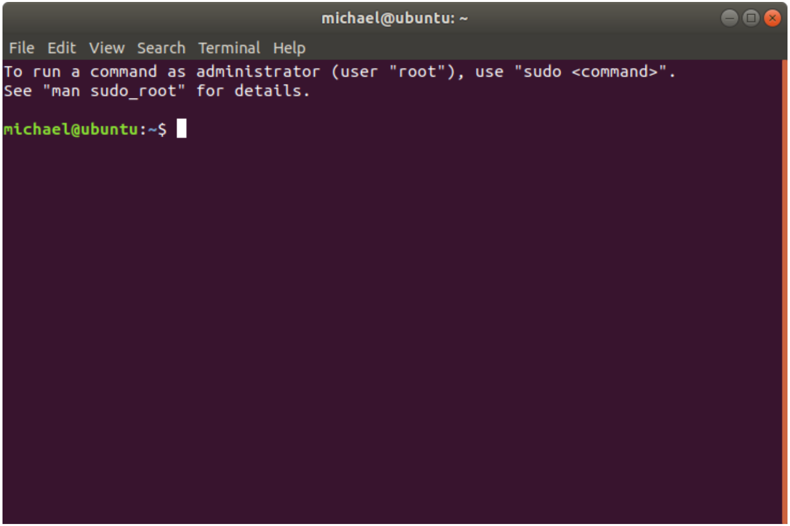
2. Type `ls` and press *Enter*

   You should see a list of files in the current directory

3. Type `cd mypathname` and press *Enter* to change the current directory

   Your current directory should have changed to `mypathname`, if that directory exists

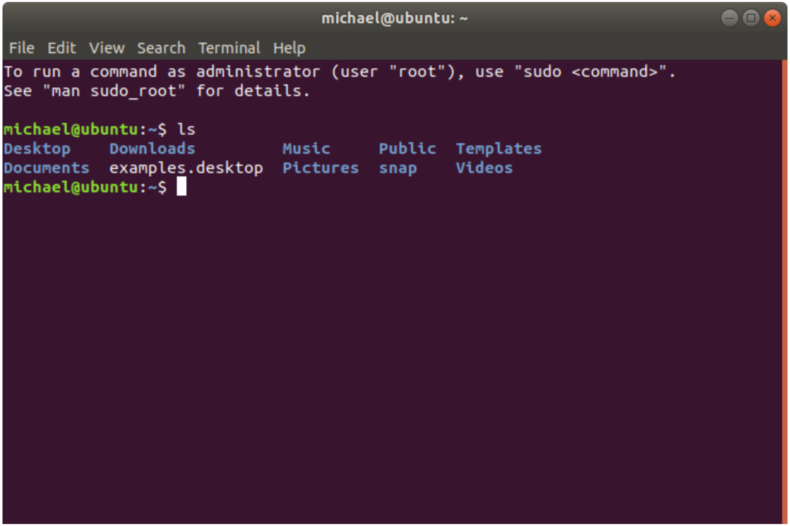
Task 0: Using the System Terminal (2)



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

michael@ubuntu:~$ 
```

Starting a terminal in Ubuntu

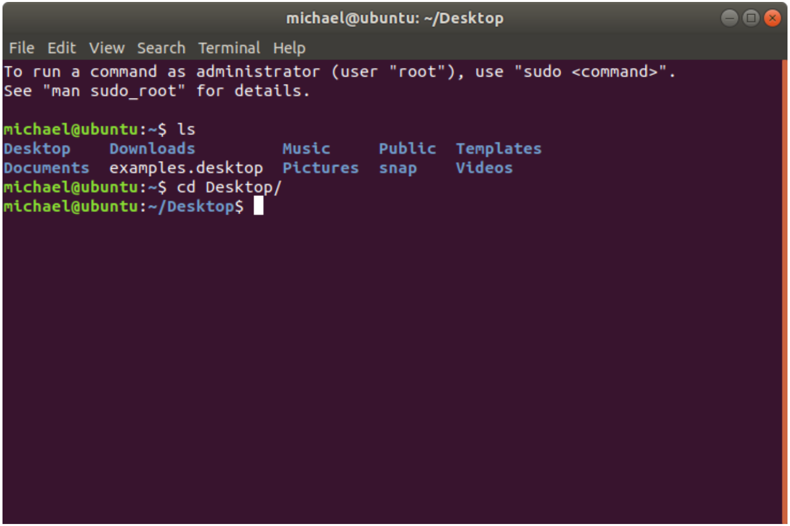
Task 0: Using the System Terminal (3)



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

michael@ubuntu:~$ ls
Desktop    Downloads         Music     Public  Templates
Documents  examples.desktop  Pictures  snap    Videos
michael@ubuntu:~$ 
```

Executing ls in a terminal in Ubuntu

Task 0: Using the System Terminal (4)


michael@ubuntu: ~/Desktop
File Edit View Search Terminal Help

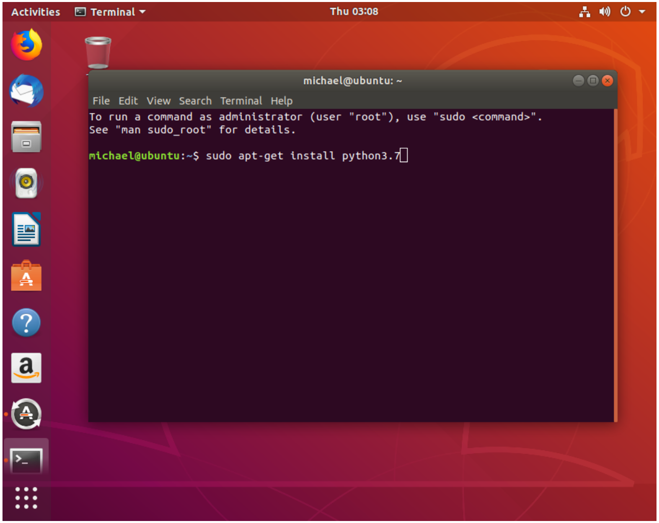
```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

michael@ubuntu:~$ ls
Desktop     Downloads          Music     Public   Templates
Documents   examples.desktop   Pictures  snap     Videos
michael@ubuntu:~$ cd Desktop/
michael@ubuntu:~/Desktop$ 
```

Executing cd in a terminal in Ubuntu

Task 1: Installing Python (1)

1. Install Python 64bit, version 3.6 or higher, on your machine

Task 1: Installing Python (2)



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

michael@ubuntu:~$ sudo apt-get install python3.7
```

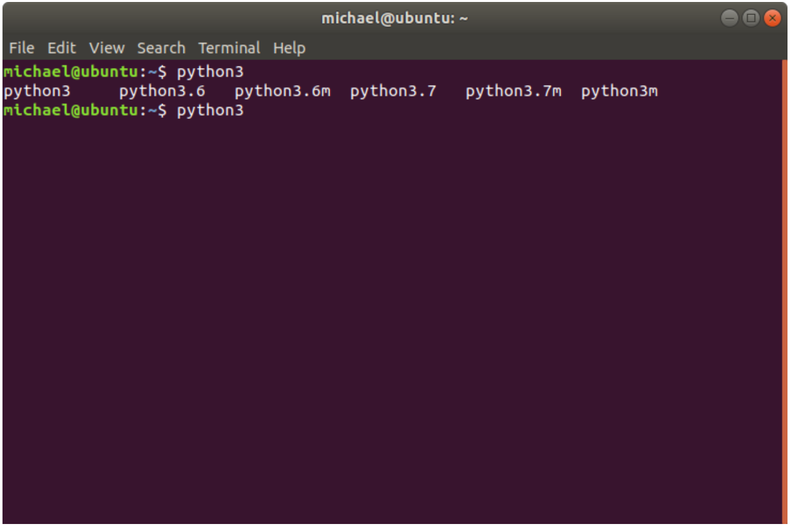
Installing Python under Ubuntu (Python 3.6 should already be installed so you can skip this). Ask if you need help.

Task 2: Using the Python Interpreter (1)

1. Open a system terminal
2. Type `python3` or `python` on windows (or `python3.7` for specific version `3.7`)

   Or type `pyth` and hit *Tab* for auto-complete (*Tab* twice for suggestions)
3. Press *Enter*
4. Now the terminal should have opened a Python interpreter, here you can use Python code
5. Verify that the Python version shown is 3.6 or higher
6. Type `4+5` and hit *Enter*
7. You should see the text `9` in your Python interpreter
8. Close the window or type `exit()` to exit the interpreter

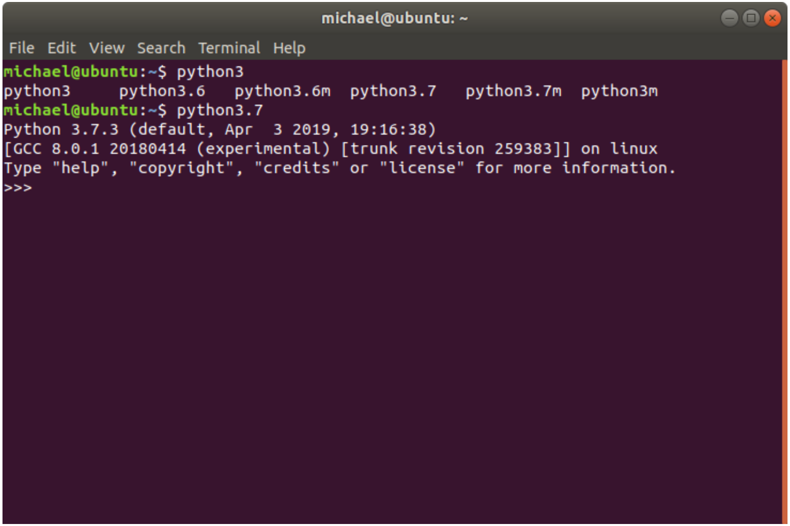
Task 2: Using the Python Interpreter (2)



```
michael@ubuntu:~$ python3
python3      python3.6    python3.6m   python3.7    python3.7m   python3m
michael@ubuntu:~$ python3
```

Tab twice for possibilities after typing *python3*

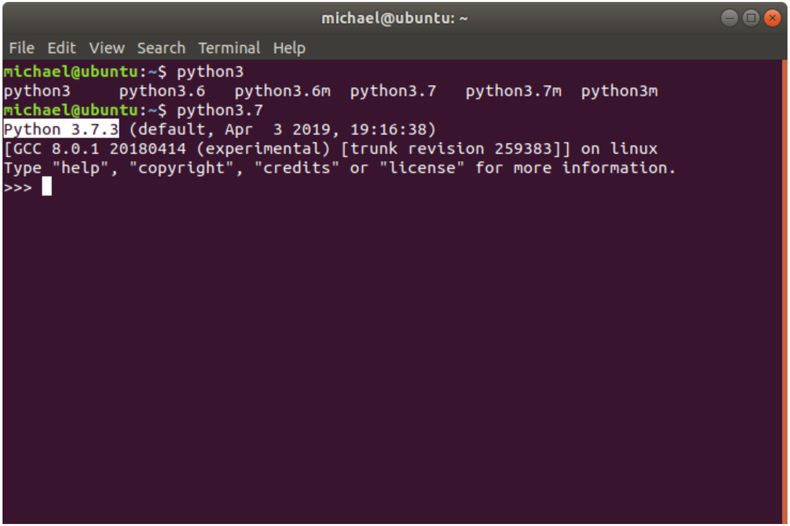
Task 2: Using the Python Interpreter (3)



```
michael@ubuntu:~$ python3
python3      python3.6    python3.6m   python3.7    python3.7m   python3m
michael@ubuntu:~$ python3.7
Python 3.7.3 (default, Apr  3 2019, 19:16:38)
[GCC 8.0.1 20180414 (experimental) [trunk revision 259383]] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Starting Python interpreter with version 3.7

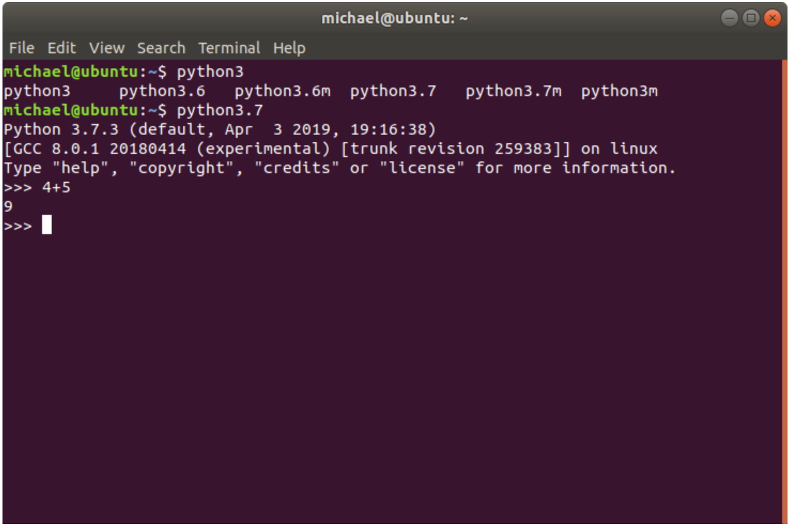
Task 2: Using the Python Interpreter (4)



```
michael@ubuntu:~$ python3
python3      python3.6    python3.6m   python3.7    python3.7m   python3m
michael@ubuntu:~$ python3.7
Python 3.7.3 (default, Apr  3 2019, 19:16:38)
[GCC 8.0.1 20180414 (experimental) [trunk revision 259383]] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Verifying Python version visually

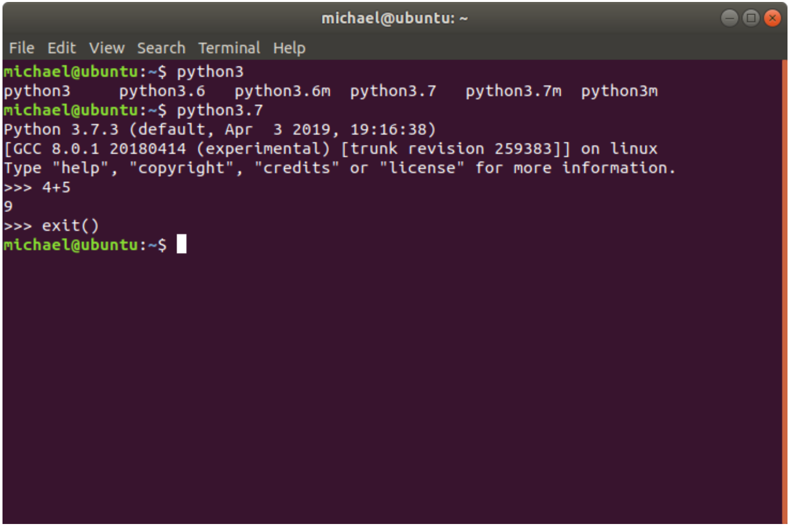
Task 2: Using the Python Interpreter (5)



```
michael@ubuntu:~$ python3
python3     python3.6    python3.6m   python3.7    python3.7m   python3m
michael@ubuntu:~$ python3.7
Python 3.7.3 (default, Apr  3 2019, 19:16:38)
[GCC 8.0.1 20180414 (experimental) [trunk revision 259383]] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 4+5
9
>>> 
```

Calculating $4 + 5$

Task 2: Using the Python Interpreter (6)



```
michael@ubuntu:~$ python3
python3      python3.6    python3.6m   python3.7    python3.7m   python3m
michael@ubuntu:~$ python3.7
Python 3.7.3 (default, Apr  3 2019, 19:16:38)
[GCC 8.0.1 20180414 (experimental) [trunk revision 259383]] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 4+5
9
>>> exit()
michael@ubuntu:~$ 
```

Exiting Python interpreter

Taks 3: Running a Python Script (1)

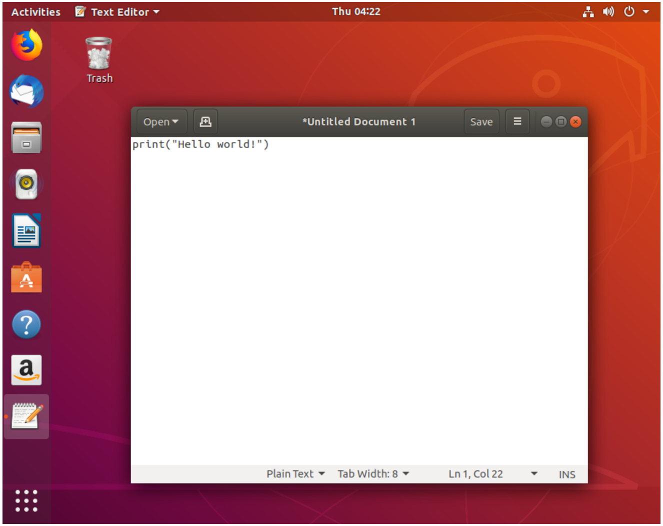
- Create an empty file named *test.py* with the contents

```
print("Hello world!")
```

  - ☐ Use `notepad`, `texteditor`, `gedit`, … to create it
  - ☐ Don't use `MSWord`, `Libreoffice`, … (will store format information in the file!)

- Run the file with Python
  1. Open a system terminal
  2. Change to the directory your file is located in:
     `cd path_to_directory`
  3. Run the file by typing `python3 test.py` and pressing *Enter*
  4. You should see the text *Hello world!* in your system terminal
  5. Ask for help if you ran into troubles

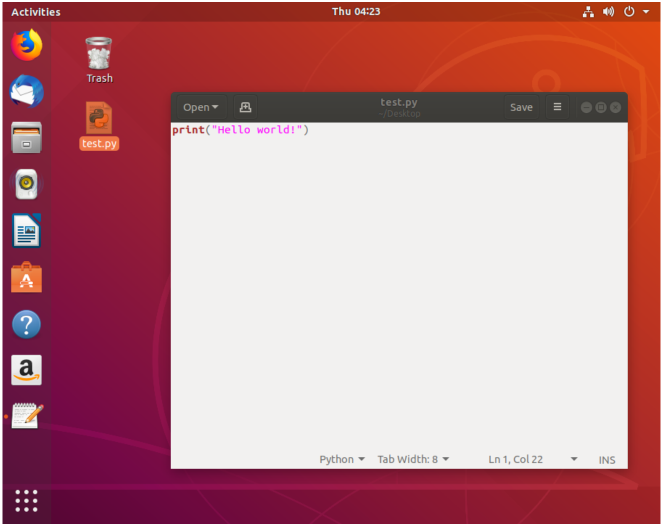
Taks 3: Running a Python Script (2)



```
print("Hello world!")
```

Opening a text editor and entering text print("Hello world!")

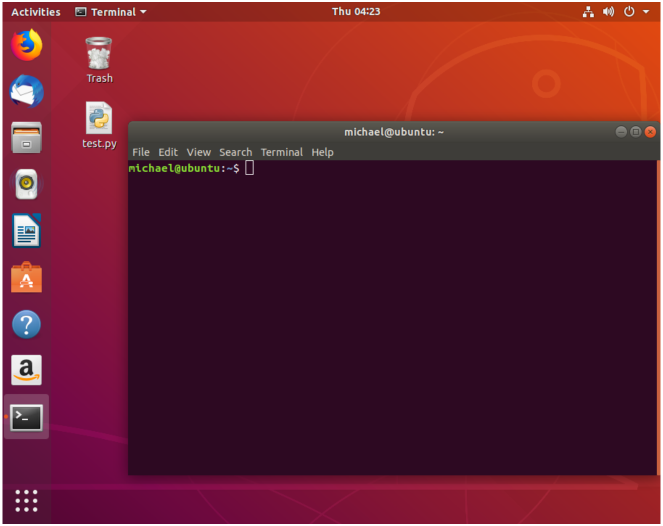
Taks 3: Running a Python Script (3)



```
print("Hello world!")
```

Saving file with name test.py to desktop

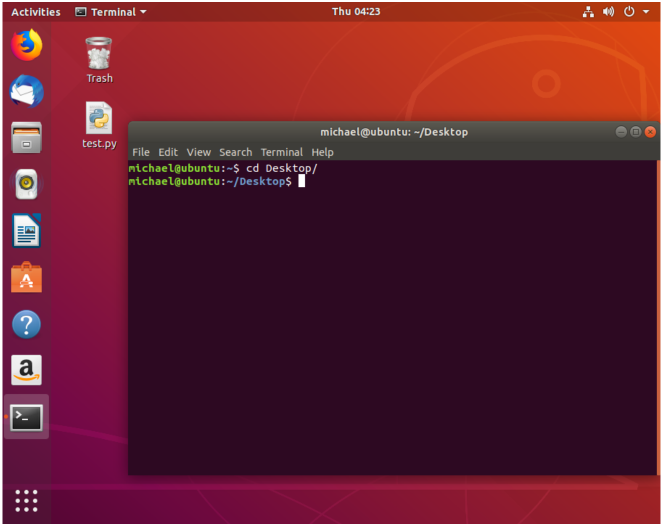
Taks 3: Running a Python Script (4)



```
michael@ubuntu:~$ 
```

Opening terminal

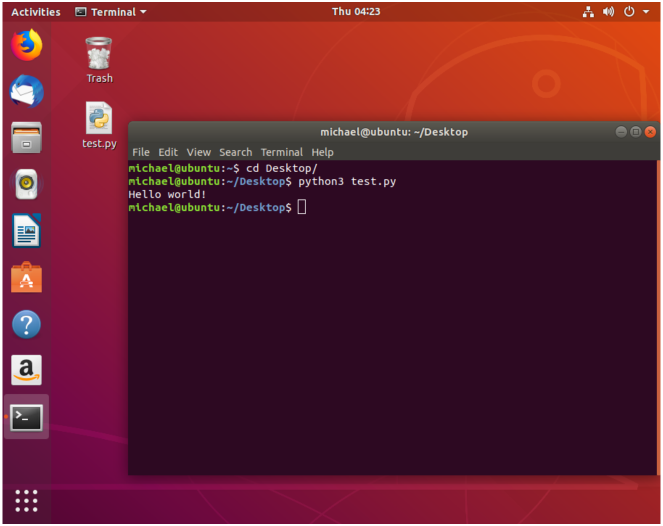
Taks 3: Running a Python Script (5)


Activities
Terminal
Thu 04:23
Trash
test.py
michael@ubuntu: ~/Desktop
File Edit View Search Terminal Help

```
michael@ubuntu:~$ cd Desktop/
michael@ubuntu:~/Desktop$ 
```

cd to directory where file is located at (in this case to desktop)

Taks 3: Running a Python Script (6)


Activities
Terminal
Thu 04:23
Trash
test.py
michael@ubuntu: ~/Desktop
File Edit View Search Terminal Help

```
michael@ubuntu:~$ cd Desktop/
michael@ubuntu:~/Desktop$ python3 test.py
Hello world!
michael@ubuntu:~/Desktop$ 
```

Running file test.py

You just ran a Python script! :)