Python in verschiedenen Programmierumgebungen ausführen

1. **Integrierte Entwicklungsumgebungen**
   - Englisch: Integrated Development Environment, kurz *IDE*)
   - Code wird geschrieben UND ausgeführt in derselben Umgebung
   - Beispiele: JupyterLab, Visual Studio Code, PyCharm etc.
2. **Command Line (CL)**
   - bei macOS Terminal bzw. bei Windows Eingabeaufforderung öffnen
   - allgemein: die CL ist eine Benutzeroberfläche zur Steuerung des Computers über Befehle, sog. *commands*:
     - einfache Befehle (ausführen jeweils mittels Enter):
       - `pwd / cd` (macOS / Windows): aktuelles Arbeitsverzeichnis ausgeben
       - `ls / dir`: Dateien und Verzeichnisse in aktuellem Arbeitsverzeichnis ausgeben
       - `cd path`: aktuelles Arbeitsverzeichnis hinzu *path* wechseln (*path* kann relativ und absolut angegeben werden, vgl. Notebook *Input und Output Teil 1*)
       - `mkdir folder`: Verzeichnis *folder* im aktuellen Arbeitsverzeichnis anlegen
       - `touch file.txt / type NUL > file.txt`: txt-Datei namens *file* im aktuellen Arbeitsverzeichnis anlegen
     - den Computer so zu steuern, ist etwas unintuitiv, dafür aber umso effizienter, v.a. bei wiederholten Operationen (vgl. Notebook *Einführung* zu den Gründen, warum wir Programmieren lernen)
     - Terminal Crashkurs, The Command Line sind zwei tolle Tutorials zur Steuerung des Computers über die CL
     - auch Module (vgl. Notebook *Funktionen und Methoden Teil 2*), die nicht zur Standardbibliothek von Python gehören, lassen sich über die CL installieren: `pip3 install module` (ohne *3* bei Windows)
   - nun lässt sich auch Python in der CL ausführen:
     - `python3` (ohne *3* bei Windows), um Python in der CL zu starten
     - anschließend können normale Python-Befehle eingegeben werden, aber nur ein Befehl pro Zeile, etwa so:

```
Last login: Wed Nov 30 11:33:29 on ttys000
(base) yannick@Air-von-Yannick ~ % python3
Python 3.9.12 (main, Apr  5 2022, 01:53:17)
[Clang 12.0.0 ] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more in
formation.
>>> x=10
>>> y=20
>>> x+y
30
```

     - Python in der CL eignet sich, um sich mit Python (und der CL) vertraut zu machen
     - `quit()`: beendet Session; wichtig: es wird nichts gespeichert!
3. **Skript + Command Line**
   - gesamten Code in einem Text Editor, bspw. Sublime Text schreiben und Code als Datei mit *.py*-Endung speichern
   - anschließend CL öffnen, zu Speicherort des Skripts navigieren (s. o.), und `python3 file.py` (ohne *3* bei Windows) ausführen, eventueller Output wird in der CL ausgegeben (Output kann durch Hinzufügen von `> output.txt` in Datei im aktuellen Arbeitsverzeichnis umgeleitet werden)

Fazit: Coding in der CL (2) eignet sich nur für kleinere Unternehmungen; ob IDE (1) oder Skript + CL (3) ist Geschmackssache, wobei IDEs den Vorteil bieten, dass alles an einem Ort abläuft; JupyterLab punktet zudem durch die Einbindung von multimodalen Inhalten (Texte, Grafiken, Videos) in den Code