# Computational Thinking, WS2023/24

## Praktische Übung 1: Wege der Python Programmierung

Prof. Dr.-Ing. Martin Hobelsberger
Dr. Benedikt Zönnchen
Prof. Dr.-Ing. Benedikt Dietrich

Sie können auf verschiedene Arten Programme in Python entwickeln. Im Rahmen unseres Kurses empfehlen wir zunächst die bereitgestellten Jupyter Notebooks in Kombination mit dem VS Code Server zu bearbeiten. Dies sollte direkt im Browser und ohne Installation funktionieren und ist somit der einfachste Einstieg in die Welt des Programmierens.

Sobald die Aufgaben komplexer werden, oder gerne falls Sie schon Vorerfahrung mitbringen von Anfang an, empfehlen wir die lokale Installation von Python und VS Code. Ab der Hälfte des Semesters werden wir vermutlich auch von Notebooks auf Skripte umsteigen, um so effizienter Programme entwickeln zu können. Ab diesem Zeitpunkt kommen Sie an einer lokalen Installation von Python und VS Code nicht mehr herum.

## Ziele dieses Praktikums

Ziel dieses Praktikums ist es, dass Sie wissen, wie sie die nächsten Praktikumsaufgaben öffnen, bearbeiten und abgeben können. Hierfür lernen Sie zunächst, was *Jupyter Notebooks* sind und wie Sie diese mit Hilfe des Datahubs und JupyterLab bearbeiten können. Im nächsten Schritt setzen Sie VS Code auf dem Datahub ein, um erste Erfahrungen mit dem Debugger zu sammeln.
Optional ist weiter unten im Dokument noch beschrieben, wie Sie Python lokal auf Ihrem Rechner installieren und nutzen können.

### Aufgabe 1.1: Neues Notebook im Datahub erstellen

Besuchen Sie als erstes unseren Datahub unter https://datahub.cs.hm.edu/. Dort können Sie *Notebooks* erstellen, verwalten, hochladen, herunterladen und ausführen.

Loggen Sie sich mit Ihrer HM-Nutzerkennung ein und wählen Sie Standard-Container aus:

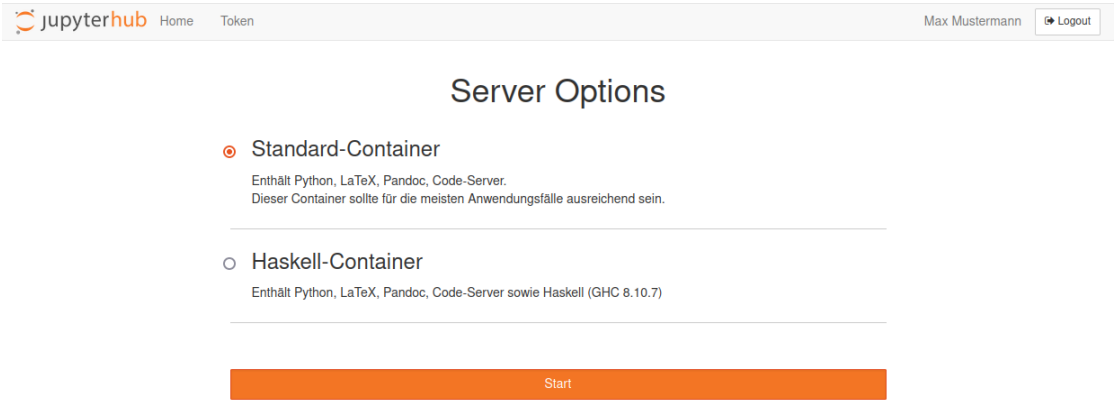


Abbildung 1: Wählen Sie Standard-Container und klicken Sie auf Start.

Erzeugen Sie nun ein neues JupyterNotebook:

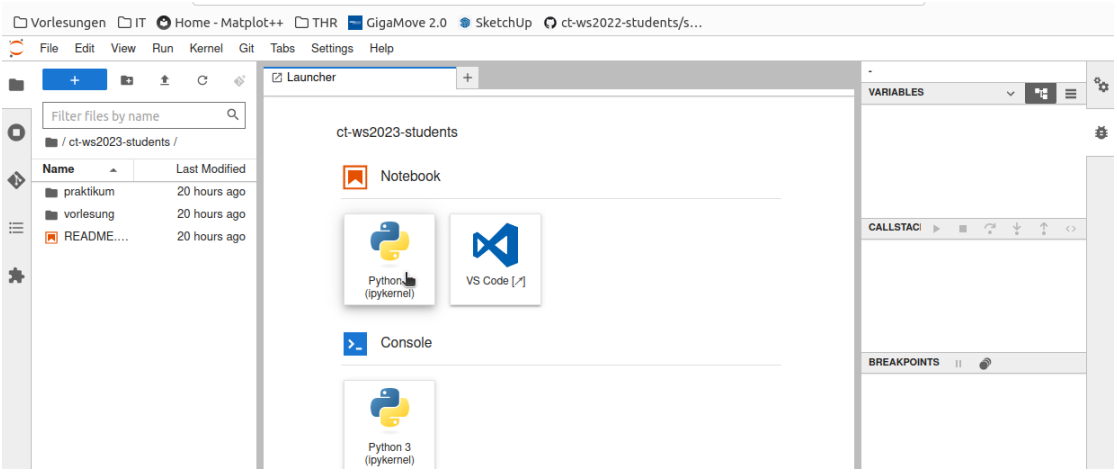


Abbildung 2: Klicken Sie im Launcher unter Notebook auf Python 3.

Glückwunsch! Sie haben Ihr erstes Jupyter Notebook erstellt.

## Aufgabe 1.2: Mit dem Notebook arbeiten

Lesen Sie nun die folgende Dokumentation zu Jupyter Notebooks: Link.

Probieren Sie dann unbedingt in Ihrem neuen Notebook folgende Dinge aus, um sich mit der Bedienung von Notebooks vertraut zu machen:

- Legen Sie Code-Zellen, fügen Sie eins der Beispiele aus der Beschreibung hinzu und führen den Code aus.
- Fügen Sie eine neue Code-Zelle hinzu und geben mit folgendem Befehl den Wert einer Variablen ‘x‘ aus:

```
x = 42
print(f"Der Wert von x ist: {x}")
```

- Setzen Sie mittels Kernel -> Restart Kernel and Clear All Outputs... alles zurück.
- Führen Sie alle Zellen des Dokuments aus.
- Vertauschen Sie Reihenfolgen von Zellen.
- Verursachen Sie Python-Fehler und lesen Sie nach Ausführung die Fehlermeldung.
- Erzeugen Sie Markdown-Zellen zu Dokumentationszwecken.

Zögern Sie nicht, bei Fragen auf Ihren Dozenten zuzugehen.
*Hinweis: Sollten Sie die Notebooks lokal bearbeiten wollen, können Sie die Notebooks vom Datahub einfach herunterladen.*

## Aufgabe 1.3: Aufgabe in den Datahub laden

Wir stellen Ihnen die Praktikumsnotebooks über Links bereit. Die Links sind so konfiguriert, dass die Aufgaben direkt in den Datahub importiert werden und die zu bearbeitende Datei geöffnet wird. Sie finden die Links zu den Praktikumsaufgaben in den Vorlesungsfolien.

a) Laden Sie die nun das Notebook `01_arbeitsumgebung.ipynb` in den Datahub, indem Sie den entsprechenden Link öffnen.

b) Der Link ist so konfiguriert, dass das Notebook mit JupyterLab geöffnet wird.

c) Lösen Sie die Aufgaben des *Notebooks* auf dem *Datahub*.

d) Führen Sie alle Zellen von oben nach unten aus.

e) Dabei sollte eine neue *Zip*-Datei entstehen (Ihre Lösung)

f) Geben Sie Ihre Lösung (die Zip-Datei) in Moodle ab.

## Aufgabe 1.4: Der Datahub und Visual Studio Code

Ein Nachteil des JupyterLabs ist die umständliche Bedienung des Debuggers. Wir empfehlen Ihnen daher, die Notebooks mit Hilfe von *Visual Studio Code (VS Code)* zu bearbeiten. VS Code ist eine schlanke, aber sehr gut erweiterbare und vielseitige Entwicklungsumgebung. Sie können VS Code entweder lokal installieren oder Sie nutzen einen sog. *Visual Studio Code (VS Code) Server* der auf unserem *Datahub* läuft. Das bedeutet, Sie können auch mit einer abgespeckten Variante von VS Code auf unserem Server/*Datahub* arbeiten.

Um VS Code manuell auf dem Server zu starten, öffnen Sie den Datahub (https://datahub.cs.hm.edu/) und wählen *File -> New Launcher* aus. Starten Sie VS Code, indem Sie auf den entsprechenden Eintrag im Launcher klicken.

Alle weiteren Links zu den Praktikumsaufgaben öffnen automatisch die Aufgaben in VS Code. Öffnen Sie nun den Link zum Praktikum `01_debugging.ipynb`.

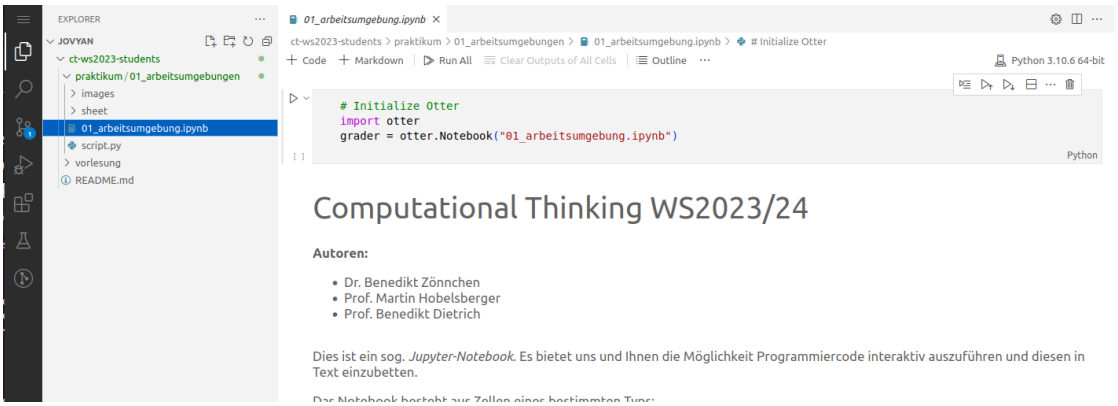


Abbildung 3: So sieht es aus, wenn Sie das Notebook in VS Code korrekt geöffnet haben.

**WICHTIG:** Bevor Sie in VS Code zum Arbeiten beginnen, müssen Sie unbedingt noch die korrekte Python-Umgebung auswählen. Klicken Sie hierfür auf das Symbol `Python 3.10.6 64-bit` und wählen Sie `base (Python 3.10.6)` aus, wie die folgenden zwei Screenshots zeigen:

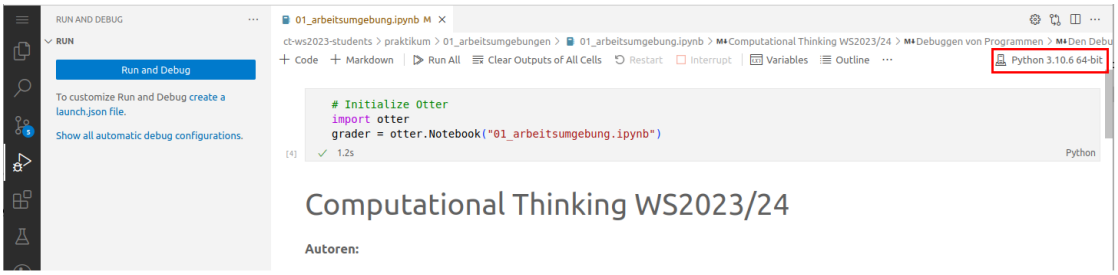


Abbildung 4: Klicken Sie auf *Python 3.10.6 64-bit...*

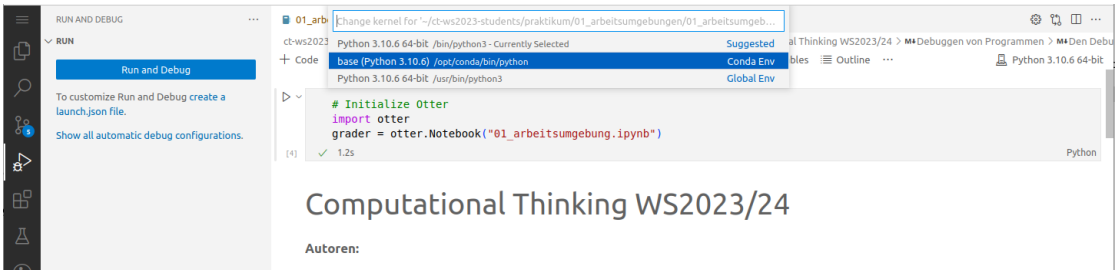


Abbildung 5: und wählen Sie *base (Python3.10.6)* aus.

## Aufgabe 1.5: Aufgabe auf dem VS Code Server bearbeiten

Jetzt sollte VS Code einsatzbereit sein. Bearbeiten Sie die Aufgabe `01_debugging.ipynb` und lernen Sie so den Debugger kennen und schätzen.

**Hinweis:**
Wollen Sie von Anfang an auch ohne Internet lokal auf Ihrem Rechner arbeiten können, können Sie die folgenden Schritte bereits durchführen. Sind Sie erst Mal mit der kennengelernten Entwicklungsumgebung zufrieden, können Sie die Aufgaben überspringen, müssen diese aber etwa zur Mitte des Semesters nachholen.

## Aufgabe 1.6: Python installieren

Zunächst müssen wir sicherstellen, dass Sie *Python* auf Ihrem System installiert haben. **Hinweise** zur Installation finden Sie beispielsweise hier.

a) Überprüfen Sie, ob auf Ihrem System *Python* installiert ist.

b) Lassen Sie sich die Version von *Python* auf Ihrer Kommandozeile ausgeben, falls *Python* installiert ist.

c) Installieren Sie `python 3.11` durch die Installation von *Python*.

**Achtung:** Achten Sie beim Installieren darauf, dass Sie Python zur `PATH`-Umgebungsvariablen Ihres Betriebssystems hinzufügen, sodass Ihr System *Python* und *PIP* findet. Unter Windows ist hierfür gleich beim ersten Dialog des Installers ein Haken zu setzen.

d) Lassen Sie sich die Version von *Python* auf Ihrer Kommandozeile ausgeben und überprüfen Sie diese.

```
python --version
```

e) Prüfen Sie auf die selbe Art ob *PIP* richtig installiert wurde.

```
pip --version
```

## Aufgabe 1.7: Starten des Python-Interpreters

Starten Sie den sog. *Python*-Interpreter und berechnen Sie wie viele Sekunden innerhalb von 5 Tagen verstreichen. Spielen Sie mit dem Interpreter herum. Probieren Sie verschiedene Eingaben aus. Sie können den Interpreter mit den Tastenkürzeln `Strg + D` bzw. `ctrl + D` beenden. **Hinweise** finden Sie z. B. hier.

## Aufgabe 1.8: Python-Skript ausführen

1) Im *Moodlekurs* finden Sie eine Datei `script.py`. Speichern Sie sich die Datei in einem Ordner, welchen Sie auf Ihrem System finden.

2) Starten Sie Ihr Kommandozeilenprogramm (Konsole, Shell, Terminal) und bewegen Sie sich in den Ordner indem sich die Datei `script.py` nun befindet.

3) Führen Sie die Datei aus.

**Hinweise** finden Sie z. B. hier.

### Aufgabe 1.9: Python-Skript schreiben I

1) Öffnen Sie einen Texteditor Ihrer Wahl (z. B. TextEdit, Notepad, Notepad++)
2) Schreiben Sie folgenden Text hinein:

```
print("Hallo Welt!")
```

3) Speichern Sie die Datei unter dem Namen `hello.py` ab. (**Achten Sie darauf, dass Sie reinen Text und keine Formatierung abspeichern**)
4) Führen Sie die Datei, d. h. Ihr *Python*-Skript über die Kommandozeile aus.
5) Verändern Sie den Inhalt der Datei, Experimentieren Sie herum und führen Sie Ihr Skript immer wieder aus.

### Aufgabe 1.10: Python-Skript schreiben II

Öffnen Sie die zuvor heruntergeladene Datei `script.py` mit ihrem Texteditor. Nutzen Sie den *Python*-Code der Datei um ein Programm zu schreiben, was Ihnen eine Zahl $n$ über die Kommandozeile einließt und die Anzahl der verstrichenen Sekunden innerhalb von $n$ Tagen ausgibt.

### Aufgabe 1.11: Jupyter-Notebooks (lokal) installieren

Verwenden Sie den Paketmanager `PIP` um die *Jupyter-Notebook-Umgebung* zu installieren. Der Kommandozeilenbefehl lautet:

```
pip install jupyterlab
```

Sie müssen die Installation möglicherweise mit `y` bestätigen. **Hinweise** finden Sie z. B. hier.

### Aufgabe 1.12: Jupyter-Notebooks lokal starten

Starten Sie nun mit dem Kommandozeilenbefehl

```
jupyter lab
```

das sog. *Jupyter-Lab* in Ihrem *Browser*.
Erstellen Sie ein neues *Notebook*.

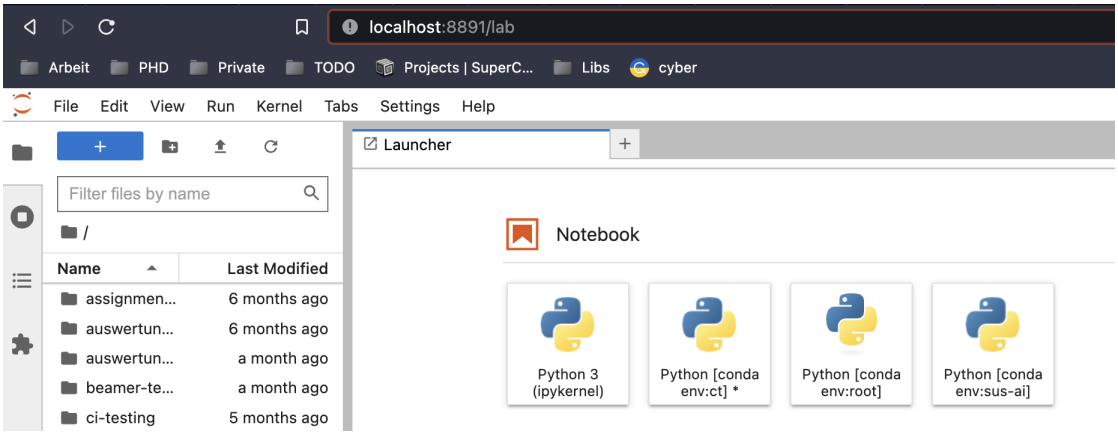


Abbildung 6: So sollte das etwa in Ihrem Browser aussehen.

## Aufgabe 1.13: Notebooks in Visual Studio Code

Sie können auch *VSC* lokal auf Ihrem Rechner benutzen. Dazu müssen Sie zunächst VSC selbst installieren (siehe https://code.visualstudio.com/). Haben Sie VSC erfolgreich installiert, benötigen Sie noch die folgenden Erweiterungen:

- Python (Identifier: `ms-python.python`)
- Jupyter (Identifier: `ms-toolsai.jupyter`)

Versuchen Sie ein *Notebook* in *VSC* auszuführen.

**Hinweis**: Wenn Sie die von uns bereitgestellten Notebooks auf Ihrem System ausführen wollen, müssen Sie ein *Python*-Paket namens `otter-grader` installieren. Installieren Sie es durch den Kommandozeilenbefehl

```
pip install otter-grader
```