Wie funktioniert der Selbstlernkurs?

Konzeption des Kurses

Der Kurs besteht aus mehrteiligen Lernmodulen zu den Grundlagen der Programmiersprache Python, diese Lernmodule enthalten:

- Kurze themenspezifische Videos mit ausführlichen Erklärungen und Live-Codebeispielen
- Ein ausführliches "interaktives Skript" zum Nachlesen und Ausprobieren
- Übungsaufgaben zum synchronen Bearbeiten mit Hilfe der Videos oder im Selbststudium

In der ersten Lerneinheit (Prolog) geben wir ausführliche Hinweise und eine "Gebrauchsanleitung", wie Sie je nach Lerntyp möglichst effektiv mit den bereitgestellten Materialien arbeiten können.

Der gesamte Kurs ist entworfen für eine selbstständige Bearbeitung unter den folgenden Gesichtspunkten:

- Die mittlere Lernbelastung beträgt 4,5-6 Stunden pro Lernmodul oder pro Woche für 7 Wochen, d.h. ein halbes Semester oder gerade die vorlesungsfreie Zeit zwischen zwei Semestern.
- Selbstverständlich ist es auch möglich, den Stoff zu strecken auf eine Vorlesungszeit.
- Eine stärkere Kondensierung auf mehr Wochenstunden ist nicht empfehlenswert, zumindest nicht, wenn Sie den Kurs ohne vorherige Programmiererfahrung absolvieren.
- Investieren Sie die Zeit lieber in selbst gestellte "Übungsaufgaben", um das Maximum aus diesem Kurs zu extrahieren.
- Wichtig: Diskussion mit anderen Kursteilnehmer*innen
- Empfehlung: Selbststudium in einem Zweier- oder Dreierteam

Benötigte Vorkenntnisse

Der Kurs bietet die Möglichkeit, dass ihn auch komplette Programmier-Neulinge bearbeiten können, deshalb sind die erwarteten Vorkenntnisse sehr gering:

- Keine vorherigen Erfahrungen mit Programmierung oder der Programmierung mit Python notwendig

- Sie sollten lediglich den grundlegenden Umgang mit Ihrem Gerät und die Installation der benötigten Software beherrschen

Nutzung des Kursmaterials

Das Material in diesem Kurs ist geeignet für die Bearbeitung mit PCs/Laptops unter Windows, Linux, und OS-X, sowie für Tablets unter Android. Wir verwenden sogenannte Jupyter Notebooks als interaktive, ausformulierte „Vorlesungsskripte“ und als Möglichkeit, damit Sie das Gelernte instantan selbst ausprobieren können. Die Notebooks werden kombiniert mit Videos, die ausführliche Erklärungen und Code-Walkthroughs bieten.

Der Kurs ist so ausgelegt, dass Sie, je nach Vorkenntnissen und eigenen Lernpräferenzen, verschiedene Möglickeiten der Nutzung haben, welche sich vor allem in der benötigten Bearbeitungszeit unterscheiden:

- Eine videobasierte Nutzung für Teilnehmer*innen ohne Vorkenntnisse, bei der Sie mit dem Studium der Videos starten, mit diesen die zugehörigen Codebeispiele bearbeiten und schlussendlich mit den ausformulierten Notebooks die Übungsaufgaben lösen

- Eine beschleunigte interaktive Nutzung, wobei Sie die Jupyter Notebooks als "Vorlesungsskript" nutzen können, um mit deren Hilfe die Übungsaufgaben zu bearbeiten und die Videos nur selektiv bei Verständnisproblemen hinzuziehen

- Ein Turbo-Modus für Teilnehmer*innen mit Vorkenntnissen in der Programmierung, wobei Sie hier direkt mit den Miniübungen und größeren Übungen starten können und die Videos und ausformulierten Notebooks nur eine Ergänzung bei Fragen darstellen

Mehr Details zu den diversen Nutzungsmöglichkeiten erhalten Sie im Lernmodul A Prolog in der Lerneinheit Gebrauchsanweisung.

Lernmodul A: Prolog

Dieses Lernmodul dient als Einführung in den Selbstlernkurs. Dieses Ziel soll in den folgenden Schritten erreicht werden:

- Vorstellung des Teams rund um und hinter dem Programmierkurs

- Eine Motivation der Programmiersprache Python durch Beispiele aus verschiedensten Bereichen

- Succes Stories vorheriger Absolventen dieses Kurses um Ihnen zu zeigen, was Sie nach Beenden des Kurses implementieren können

- Der erste Gebrauch eines Jupyter Notebooks zur Implementierung von Python-Programmen

- Wir geben dann eine kurze Gebrauchsanleitung für den Ablauf, die Verwendung und Idee des Kurses

- Der Abschnitt schließt mit einer Installationsanleitung für die benötigte Software zur Ausübung des Kurses

Hinweis: Viele Studierende steigen gerne schnell ein. Deshalb ist ein alternativer Weg durch dieses Lernmodul, nach der Motivation zunächst Abschnitt 5 und 6 durchzuarbeiten. So können die Beispiele direkt nachvollzogen werden.

Lernmodul B: Grundlagen

Am Ende dieses Lernmoduls werden wir in der Lage sein:

- Von Hardware-Details innerhalb des Computers zu abstrahieren

- Erste einfache Python-Programme in lesbarer Weise zu schreiben

- Darauf aufbauend unsere Reise ins Python-Universum zu beginnen

Diese Ziele soll hierbei in den folgenden Schritten erlernt werden:

- Diskutieren der grundlegenden Arbeitsweise von Computern

- Kennenlernen des allgemeinen Syntax von Python-Programmen und Nutzung der eingebauten Python-Hilfe

- Verstehen des Konzeptes der Variablen, in denen Computer gewissermaßen Daten zwischenspeichern können

- Eine Einführung in Datentypen erklärt, warum man (wie in der richten Welt übrigens auch) nicht Äpfel mit Birnen vergleichen kann

- Wir beschäftigen uns dann mit der Möglichkeit, Eingaben von Benutzer*innen unserer Programme zu verarbeiten, und lernen umgekehrt auch Möglichkeiten kennen, wie wir die Ausgaben unserer Programme strukturiert und "hübsch" darstellen können

- Das Lernmodul schließt mit der Vorstellung von while-Schleifen, einer ersten Kontrollstruktur

Lernmodul C: Datentypen, Datenstrukturen und Kontrollstrukturen

Wir kennen anekdotisch bereits ausgewählte Datentypen, nämlich ganze und (approximierte) reelle Zahlen. Außerdem kennen wir mit Zeichenketten bereits eine erste Datenstruktur, und mit der while-Schleife eine erste Kontrollstruktur. In dieser Lerneinheit vertiefen, systematisieren

und erweitern wir diesen "Dreisprung" der essentiellen Elemente einer Programmiersprache. Mit der Gleitkomma-Arithmetik betrachten wir zudem eine Frage der Computer-Architektur, deren Verständnis nicht nur für mathematische Berechnungen essentiell ist.

Unterwegs stärken unzählige kleine Beispiele unser Verständnis. Diese Beispiele verdeutlichen zudem, wie viele Dinge wir tatsächlich schon nach zwei Lerneinheiten dieses Python-Einführungskurses programmieren können, und wie viele Probleme wir mit unseren noch elementaren Programmierkenntnissen tatsächlich schlauer, effizienter und besser lösen können als mit "Papier und Bleistift"! Das ist natürlich eine gute Gelegenheit, daran zu erinnern, dass Programmieren lernen unmöglich ist, ohne sich die "Finger selbst schmutzig zu machen" ...

Bis zum Ende dieses Lernmoduls werden wir insbesondere die folgenden Inhalte kennenlernen:

- Der bool Datentyp für logische Variablen
- Der int Datentyp für ganzzahlige Variablen
- Der float Datentyp für reelle Zahlen und Aspekte der Gleitkomma-Arithmetik
- Der complex Datentyp für komplexe Zahlen
- Vergleichsoperatoren für elementare Datentypen
- Bedingte Ausführunge mittels if Abfragen
- Die Datenstruktur Listen (Sequenzen), sowie wichtige Operationen zur Verwendung von Listen
- Wiederholte Ausführungen mittels for Schleifen

Wir behandeln also elementare Datentypen und ihre Verwendung, und mit Listen einen fortgeschrittenen Datentyp beziehungsweise eine Datenstruktur (ähnlich zu Zeichenketten, die wir bereits kennen). Weiterhin lernen wir Kontrollstrukturen kennen.

Lernmodul D: Datenstrukturen, List Comprehensions und Funktionen

Diese Lerneinheit besteht im Wesentlichen aus drei Teilen: Zunächst diskutieren wir drei Datenstrukturen im Detail, nämlich (noch einmal) Zeichenketten, gefolgt von Dictionaries und Mengen.

Dieser Teil wird unterbrochen durch ein kurzes Intermezzo zur Zeitmessung, um die Frage untersuchen zu können, welche Datenstruktur gewählt werden sollte für eine Fragestellung, wenn mehrere Datenstrukturen infrage kommen. Dies führt auf die Diskussion sogenannter Comprehensions, einer sehr effizienten Möglichkeit, über Datenstrukturen zu iterieren. Danach

betrachten wir ausführlich unterschiedliche Tücken und Fallstricke bei den bisher vorgestellten Datenstrukturen.

Im dritten Teil stellen wir Funktionen als Gliederungskonzept für (größere) Programmierprojekte vor, und betrachten die Rekursion, d.h. Funktionen, die sich selbst aufrufen.

Bis zum Ende dieses Lernmoduls werden wir insbesondere die folgenden Inhalte kennenlernen:

- Die Datenstruktur der Zeichenketten zur Arbeit mit Strings
- Dictionaries als weitere Datenstruktur, welche Schlüssel mit Werten verknüpft
- Mengen als alternative Datenstruktur, bei der automatisch sichergestellt ist, dass jedes Element nur einmal vorkommt
- Zeitmessungen und Code-Performance zur Abschätzung von "gutem" Code
- List, Set und Dict Comprehensions zur effizienten Programmierung
- Tücken bei Datentypen: Referenzieren vs. Kopieren, Mutable vs. Immutable
- Funktionen und Rekursionen

Lernmodul E: Funktionen und Module

In dieser Einheit lernen wir zunächst viele fortgeschrittene Aspekte von Funktionen kennen. Insbesondere vereinigen wir die bisher getrennte Diskussion von Datentypen/Datenstrukturen inklusive ihrer (Im)mutabilität mit der Diskussion von Funktionsargumenten und -Rückgabewerten. Im Gegensatz zu den vorherigen Lerneinheiten führen wir in diesem Teil also nicht eine "Flut" an neuen Python-Sprachkonstrukten ein, sondern "lehnen uns zurück" und konsolidieren unser bisheriges Wissen.

Dieses vertiefte Verständnis führt dazu, dass wir beginnen zu verstehen, wie Python-Pakete wie math, cmath, random und time tatsächlich unter der Haube funktionieren. Am Ende sind wir in der Lage, zu verstehen, was der import-Befehl tut, und wie wir eigene Module / Pakete schreiben können.

Im Verlauf dieses Lernmoduls werden wir insbesondere die folgenden Inhalte kennenlernen:

- Die Eingabe von Standardwerte für Argumente von Funktionen
- Benannte Funktionsargumente zur direkten Zuordnung von Variablen und Daten
- Erstellen von Funktionen mit variablen Anzahlen von Argumenten
- Keyword-Argumente zur Übergabe von benannten Funktionsargumenten

- Funktionen als Argumente von Funktionen
- Lokale und globale Sichtbarkeit von Variablen
- Mutable und immutable Funktionsargumente
- Anonyme Funktionen zur lokalen Verwendung
- Dokumentation von Funktionen
- Module, imports und die Standard-Bibliothek

Lernmodul F: Dateien, Fehlersuche und Pakete

In dieser Lerneinheit runden wir bereits unsere Python-Kenntnisse ab. Das mag jetzt etwas verblüffend wirken, aber tatsächlich lernen wir nach dieser Lerneinheit keine neuen Sprachkonstrukte und Eigenschaften der funktionalen Programmierung mit Python mehr kennen: Wir haben alle Grundlagen mindestens einmal gesehen, die notwendig sind, eigene auch komplexe Programme in Python zu schreiben, und die Module aus der Standardbibliothek von Python erst zu verstehen, und dann zu nutzen. Es sollte im Laufe des Kurses klar geworden sein, dass Programmieren letztendlich eine Frage von Erfahrung ist.

Im Verlauf dieses Lernmoduls werden wir insbesondere die folgenden Inhalte kennenlernen:

- Lesen und Schreiben von Dateien
- Ausführliches Beispiel zum Lesen und Schreiben von Dateien: csv-Dateien
- Weitere Dateioperationen, neben dem Arbeiten mit Textdateien
- Fehlerbehandlung und Fehlertypen beim Programmieren
- Unterstützung bei der Fehlersuche mittels Assertions, Exceptions und Debuggern
- Pakete und die Paketverwaltung

Lernmodul G: Die Pakete NumPy, matplotlib und SymPy

Pakete erweitern den Funktionsumfang des Python-Universums um problemspezifische Lösungen. Da Python ein Open Source Projekt ist, gibt es für viele Fragestellungen bereits vorgefertigte Lösungen, die wir mit der Standardbibliothek sonst aufwändig nachprogrammieren müssten.

In dieser Lerneinheit stellen wir drei solche Pakete ausführlicher vor, die in vielen technischen Studiengängen von Bedeutung sein können, nämlich NumPy für Matrix-Vektor Rechnungen und damit verbundene Aufgaben wie lineare Gleichungssysteme, matplotlib zur Erstellung von

graphisch ansprechenden Plots, Animationen und Visualisierungen, und SymPy für symbolische, d.h. exakte Rechnungen und Formelmanipulationen.

Insbesondere möchten wir damit die Befähigung vermitteln, sich eigenständig in unbekannte Pakete einzuarbeiten, mit einer Kombination aus existierenden Tutorials im Internet, üblichen Community-Seiten wie stackexchange, und einer gesunden Portion Grundwissen aus diesem Kurs. Sie werden am Ende hoffentlich zustimmen, dass dies keine Raketenwissenschaft ist, sondern dass Sie im Verlauf des Kurses genau diese Befähigung erworben haben.

Im Verlauf dieses Lernmoduls werden wir insbesondere die folgenden Inhalte kennenlernen:

- Arbeiten mit Paketen und die Paketverwaltung
- Das Paket NumPy zum Arbeiten mit numerischer linearer Algebra
  + Speicherung von Vektoren und Matrizen
  + Rechnen mit Vektoren und Matrizen
  + Das Unterpaket numpy.linalg für die Verwendung hilfreicher Funktionen für Aufgaben der linearen Algebra
- Das Paket matplotlib zum Plotten und Visualisieren von Daten
  + Fortgeschrittene Aspekte des Pakets matplotlib zum Erstellen individueller Plots für gewünschte Anwendungsaufgaben
- Das Paket SymPy für exakte symbolische Berechnungen und Computeralgebra
  + Grundlagen der symbolischen Rechnung
  + Anwendungsbeispiele für die Nutzung des Paketes SymPy

Lernmodul H: Objektorientierte Programmierung

In dieser Einheit lernen wir eine beinahe völlig neue Art der Programmierung mit Python kennen, nämlich die objektorientierte Programmierung (OOP). OOP ist ein deutlich moderneres Programmierparadigma als die Art zu programmieren, die wir bisher betreiben. Tatsächlich sind "unter der Haube" fast alle Python-Pakete und Module der Standardbibliothek objektorientiert implementiert, dies haben wir im Verlauf des bisherigen Kurses bereits mehrfach angedeutet.

Diese Einheit kann nur einen ersten Einblick geben. Wir konzentrieren uns dabei auf Klassen und Kapselung als das zentrale Konzept der objektorientierten Programmierung, und diskutieren weitere Aspekte wie Polymorphie, Vererbung etc. nur oberflächlich.

Im Verlauf dieses Lernmoduls werden wir insbesondere die folgenden Inhalte kennenlernen:

- Klassen und Objekte als ein grundlegendes Konzept für objektorientierte Programmierung
- Anlegen von Klassen und Objekten mit Attributen und Methoden
- Vererbung um Unterklassen bereits bestehender Klassen abzuleiten
- Ein "Schmankerl" zum Schluss: Die Mandelbrotmenge

Lernziele

Der Kurs zielt darauf ab, dass Sie sich im Selbststudium die folgenden grundlegenden Fähigkeiten und Kompetenzen bezüglich Python aneignen können:

- Beherrschung der Sprachelemente von Python
- Fähigkeit zur eigenständigen Erstellung von Python-Programmen zu gegebenen Fragestellungen und die Übersetzung einer Aufgabe in ein Programm
- Beherrschung der Modularität von Python, wie z.B. das Finden eines Python-Moduls für eine gegebene Fragestellung vs. eigene Implementierung
- Übliche Programmier-Skills: Fehlersuche, Datenstruktur-Auswahl etc.

Zusätzlich gibt es übergeordnete Lernziele, welche im Verlauf des Kurses vermittelt werden sollen:

- Programmieren kann Spass machen
- Programmieren ist sehr relevant auch in Studiengängen in denen dies nicht Teil des Curriculums ist
- Ohne grundlegende Kenntnisse der Programmierung und der Arbeitsweise von Computern ist Digital Literacy unmöglich

Für Lehrende

Für Lehrende bietet dieser Kurs zudem eine Vorlage, die beliebig durch "remixen" an eigene Bedürfnisse in der Lehre angepasst werden kann:

- Dazu sind beispielsweise Erläuterungen von Beispielen getrennt (in Notebooks und in Videos)

- Das Grundgerüst des Kurses kann hergenommen werden und durch eigene Beispiele und Verständnisübungen ergänzt werden

- Durch die Lizenzierung als CC-BY ist tatsächlich jede Form des "Remixens" erlaubt und erwünscht

Zum empfohlenen Aufbau des eigenen Kurses ist wichtig zu beachten:

- Das Lernmodul "Prolog" bietet vier verschiedene Einstiegsmöglichkeiten in den Kurs, zwischen denen ausgewählt werden kann oder durch eine eigene zielgruppenspezifische Motivation ersetzt werden kann.

- Das Lernmodul "Grundlagen" sollte vermutlich 1:1 übernommen werden, auch wenn hier die Möglichkeiten zur Individualisierung auf eigene Studiengänge und Zielgruppen bereits angedeutet sind am Ende des Moduls

- Die folgenden Module werden immer weniger in sich geschlossen, und erfordern bei einer konsequenten Anpassung jenseits von einer allgemeinen Schlüsselqualifikation immer mehr Individualisierung

- Ein "Durchscrollen" durch die Lernmodule verdeutlicht das Baukasten-Prinzip

- Ebenso einfach sollte es möglich sein, den Stoff zu remixen, so dass die empfohlene wöchentliche Stoffmenge für die Studierenden angepasst werden kann

- Ab Modul 2 existieren Abschnitte "Globale Verständnisfragen", die einem inverted classroom Szenario entsprechen. Die Inhalte in den Abschnitten "Übungsaufgaben" entsprechen klassischen wöchentlichen Übungen.

Generell gilt, dass die Videos und Notebooks mit dem Ziel der Remixbarkeit erstellt wurden. Die Einbettung in einen ILIAS-Kurs oder ein ähnliches eLearning-System ist jedoch immer Studiengangs- und Standort-spezifisch, um Studierenden einen roten Faden zu bieten.

Rückmeldungen und konstruktive Kritik an dominik.goeddeke@ians.uni-stuttgart.de sind immer herzlich willkommen.

Hinweis: Der Inhalt dieser Datei wurde aus vier Textdateien kopiert. Nicht alle Daten des Python-Kurses sind in der HubbS-Mediathek zu finden. Die Jupiter-Notebook-, Scorm- und Python-Übungsdaten finden sie auf https://zoerr.de.