

Programmieren mit Python in der Sek. 1

20.9.2010

Reinhard Oldenburg

Überblick

- Kurzvorstellung: Was ist Python
- Didaktische Überlegungen zur Stellung des Programmierens
- Gründe für Python
- Beispiele

Was ist Python?

- Kurze Antwort: Eine interaktive, interpretierte Sprache mit einer knappen, lesefreundlichen Syntax
- Python ist....
 - prozedural, funktional und objektorientiert
 - dynamisch typisiert
 - weit verbreitet, auch im professionellen Einsatz
 - gut unterstützt durch Bibliotheken, Literatur, Foren
 - plattformübergreifend (Win, Linux, Mac, JVM, .NET)
 - Unter Windows können *.exe Dateien erzeugt werden
- Ähnliche Sprachen: Ruby, Lua, R, Boo, ...
- IDEs: Kommando-Zeile, IDLE und Spyder
- Ein paar elementare Beispiele:
 - Rechnen, Zeichenketten, Listen, Funktionen; Mathe: Zahlentheorie
- Oh Gott – ist das nicht wie BASIC, Spagetti-Code und Trial&Error?
- Python erlaubt „quick&dirty“
 - Positiv: Projekte der für die Schule typischen Größe lassen sich leicht realisieren, ohne unübersichtlich zu werden
 - Python unterstützt auch systematisches Programmieren

Didaktische Überlegungen zum Programmieren

- Ist Programmieren (noch) wichtig?
- Alte Diskussion, siehe zB Hubwieser: Didaktik der Informatik
- Meine Antwort: Ja, denn...
 - Primärerfahrung, analog zum Zählen in der Mathematik
 - Handlungsorientierung, Produktorientierung
 - Problemlösen
- Aber: Natürlich darf Info-Unterricht nicht zum Programmierkurs werden, der den Blick auf das große Ganze verstellt (zB Wirkung der IKT)
- Eine Programmiersprache erlaubt, Problemstellungen (Problemlösen) präzise auszudrücken (Darstellen) und umzusetzen (Gestaltung), sie sollte dabei so schnell wie möglich unsichtbar werden, genau wie die menschliche Sprache
- Eine gute Sprache für den Informatikunterricht ist also eine, die sich möglichst schnell lernen lässt, so dass danach mit ihr, nicht über sie gesprochen wird

Didaktische Überlegungen zum Programmieren

- Programmieren auch in der Sek I?
- Ja, nach Bruner sollte man sich bemühen, fundamentale Ideen auf jeder Altersstufe intellektuell angemessen zu behandeln
- Eine gute Möglichkeit: Scratch. Defizite daran:
 - Viele interessante Anwendungen, zB Bildverarbeitung, jenseits der Reichweite
 - Keine Vermittlung eines authentischen Bildes von der praktischen Arbeit in der Informatik
- Python kann als erste Sprache oder im Anschluss an Scratch eingesetzt werden
- Python eignet sich auch für die Sek II (Schulversuch: Genetischer IU) und die Uni (MIT, Uni Frankfurt,...)

Gründe für Python

- Interaktivität: Einzelne Funktionen können erprobt werden; unmittelbare Rückmeldung [Lernpsychologie]
- Knappe Syntax, gute Lesbarkeit
 - LINKWEILER, Ingo: *Eignet sich die Skriptsprache Python für schnelle Entwicklungen im Softwareentwicklungsprozess? – Eine Untersuchung der Programmiersprache Python im softwaretechnischen und fachdidaktischen Kontext.*
- Knappe Semantik: Keine Sonderregelungen für bestimmte Datentypen; Funktionen sind auch Objekte
- Modellvorstellung dazu, wie Pythonprogramme abgearbeitet werden, ist relativ einfach
 - Mit Papier und Bleistift ausführen
 - Wertetabellen werden durch Spyder gut illustriert
- Maschinennahe Spezialitäten bleiben meist unsichtbar
 - Bsp: Unterscheidung int, long

Gründe für Python

- Literale, also Sprachmittel um Daten direkt hinzuschreiben, insbesondere Listen
 - Vergleiche: Listenerstellung in Python: `L=[34,1,99]`
 - und in Java: `LinkedList L= new LinkedList(); L.add(new Integer(34)); L.add(new Integer(1)); L.add(new Integer(99));`
- Na ja, ganz praktisch, aber ist das wirklich so wichtig?
 - Praktikerregel: Zahl der korrekten Programmzeilen/Stunde konstant über verschiedene Sprachen: Einfacher ist besser
 - Semiotische Lerntheorie: Abstrakte Objekte werden durch das Hinschreiben erzeugt („Symbolizing mathematical Objects into being“). Historisch, Bsp $\sqrt{2}$
 - Literale erlauben das Operieren mit einer Darstellung.
 - Literale von Listen „zeigen“, was man damit machen kann
 - Davon abstrahiert: Operation mit einem mentalen Objekt
 - Darstellung → mentales Modell
- Beispiele:
 - Modellieren eines Weitsprungwettbewerbs: Datenstrukturen können prototypisch hingeschrieben werden: Demo
- Prototypisches Modellieren als Weg zu abstrakteren Modellierungstechniken
 - Universelle Datenstruktur der Liste durch Literale leicht zugänglich
 - Andere Strukturen werden damit modelliert

Gründe für Python

- Es gibt viele schöne Bibliotheken für Python, u.a.
 - 3D-Grafik mit Visual Python(www.vpython.org)
 - WebCam: VideoCapture (<http://videocapture.sourceforge.net/>)
 - MS Speech SDK (<http://code.google.com/p/pyspeech/>)
 - pygame (<http://www.pygame.org>)
 - Netzwerk- und Internetprogrammierung (im Lieferumfang)
 - GUI (zB TKinter, PyQt)
- Python wird als Skriptsprache in vielen Anwendungen verwendet
 - Gimp, OO, Gnumeric,....
- Große Community, viel Literatur
- Aber: Noch zu wenig für Schüler geeignetes Material

Schluss

- Literaturtipp:
 - G. Lingl: Python for Kids
 - A. B. Downey et al.: Wie ein Informatiker denken lernen ... mit Python,
<http://ada.rg16.asn-wien.ac.at/~python/how2think/index.html>
- Vielen Dank für Ihre Aufmerksamkeit!
- Gibt es noch Fragen oder Wünsche?

Homepage: <http://www.math.uni-frankfurt.de/~oldenbur/>