Cursus Creatief Programmeren in Python

DOCENTENHANDLEIDING – Ethan van Woerkom

Het doel van deze cursus is dat, gespreid over acht lessen van ieder twee uur, middelbare scholieren:

1. De basis van de programmeertaal Python 3 leren. (Focus lessen 1-4)
2. Programmeer-basisvaardigheden leren zodat zij zelfstandig verder kunnen.
3. Creatieve projecten uitvoeren. (Focus lessen 5-8)

Lesstructuur
Elke les wordt verdeeld in:

1. 40-50 minuten instructie.
2. 50-60 minuten “workshop” - programmeren in de les.

- Tijdens (1) worden aan de hand van een live programmeersessie met IDLE en het whiteboard programmeerconcepten onderwezen.
- Tijdens (2) werken de leerlingen in aan een reeks oefenopgaven. Aan het begin bespreken we de vorige huiswerk opdracht en laten mogelijk een oplossing van een leerling zien.
- In de laatste 15 minuten van de workshop bespreken we:
  1. Oplossingen van de opgaven in grote lijnen.
  2. Refereren terug naar de hoofdpunten van de les.
  3. De aankomende huiswerkopdracht. Hierbij kan een moeilijke plus opdracht zijn.

Lesmaterialen
Folder met 4 lesplannen
Folder met (.py / .txt) programmeervoorbeelden genummerd 1.a t/m 8.g.
Folder met 4 workshops met huiswerk opgaven.

Online Resources
Nuttige online cursus voor leerlingen als referentie en oefening:
https://cscircles.cemc.uwaterloo.ca/nl/

Nuttig to-the-point Python boek, ook handig voor de docent als inspiratie:
“Python in Easy Steps”, Mike McGrath.

Online Python Interpreter: https://replit.com/languages/python3

Officiële Python Documentatie: https://docs.python.org/

Officiële Python Tutorial (Zeer uitgebreid): https://docs.python.org/3/tutorial/index.html

Algemene vragen: https://stackoverflow.com/

Handige Youtube-videos:
FreeCodeCamp - Full Python Course:
https://www.youtube.com/watch?v=rfscVS0vtbw
Programming with Mosh: Full Python Course:
https://www.youtube.com/watch?v=_uQrJ0TkZlc

Overzicht te leren vaardigheden:
het is noodzakelijk voor het begrip dat de leerling dit leert: groen
het is goed als de leerling hier kennis van neemt: blauw
informatie nuttig voor leerling, maar niet kritiek: geen kleur

Python - Taal

print(); input(); print() geavanceerde formatting; comments;

variables en toegestante namen; assignment operators (=, -=, += etc.); types, type(), arithmetic (+, -, *, /, //, **, ());

comparison operators (>, <, ==, !=, >=, <=), bool; logical operators (and, or, not)

conditional statements (if, while, break, continue); for–loop en range(a,b,c); list comprehension;

strings & escapes; concatenating strings;

functions(def, return) & BIFs (er zijn 79); local en global scope; import math;

lists (append, remove, +, *, pop); slice notation; del, tuples; dictionaries?.

classes, methods & OOP (gebruiken, definiëren); mutable vs immutable; pass-by-reference en pass-by-value en is keyword (shallow vs deep); File I/O.

Algemeen Programmeren

Wat is Programmeren; Programmeertaal vs markup (Python vs HTML); Compiled vs Interpreted;

expression vs (conditional) statement (vs declaration); betekenis a=b (het is geen wiskundige vergelijking); strongly vs weakly typed floating-point arithmetic

Console (cd, ls/dir, python3); programmeren met IDLE editor, IDLE console, text editor (notepad++) & terminal; packages downloaden met PIP; verschil Windows, Linux, MacOS met betrekking tot Unix, terminal;

flowcharts; algoritmisch denken; efficiency/Big-O notatie.

syntax/runtime errors, bugs; debugging (syntax checker, betekenis error message, debuggen met print statements);

design cyclus: schrijf programma in aparte delen, debug, klaar, herhaal; multi-file project;

ASCII letters, chr(), ord().

Creatief (mogelijke projecten)

Interactieve Website; Mobiele App (Kivy/Beeware); Computer Simulatie (numpy matplotlib); Animaties (matplotlib); Networking (bijv. Chat applicatie); Computer GUI Programma (tkinter); HTML scraping (BeautifulSoup/lxml); Wetenschappelijk (Hisparc/-Sapphire / SDSS-SQL); Games (PyGame).

Lesplan – welke onderwerpen komen aan bod:

Les	Note	Python-Taal	Algemeen
1		print(), input(); comments; variables en toegestante namen; assignment operator (=); types, type(), arithmetic (+, -, *, /, //, **,, % ()); type casting; (concatenating) strings; if; comparison operators (>, <, ==, ! =, >=, <=), bool; typecasting.	Wat is Programmeren; Programmeertaal vs markup (Python vs HTML); Compiled vs Interpreted; ; betekenis a=b (het is geen wiskundige vergelijking); strongly vs weakly typed; .py bestanden, python runnen met IDLE
2	Leerlingen aansporen te denken over les 5-8.	Typecasting; Assignment operators (-=, +=, *=); conditional statements (if, while, break, continue); strings & escapes;	expression vs (conditional) statement (vs declaration); flowcharts; syntax/runtime errors, bugs
3	Leerlingen vragen wat zij willen doen in lessen 5-8.	for & range(n); functions(def, return) & BIFs (er zijn 79); local en global scope; lists maken en accessen met []; lists (append, pop, insert,, remove); range(start, stop, step), slice notation;	floating-point arithmetic, format(float, '.12g');
4		del; classes, methods & OOP (gebruiken, definiëren); objects: mutable vs immutable; pass-by-reference en pass-by-value (naming/binding) en is keyword (shallow vs deep); uitloop ruimte.	
5	Les 5 t/m 8 vrij in te vullen samen met leerlingen	logical operators (and, or, not); tuples	Console (cd, ls/dir, python3); programmeren met IDLE editor, IDLE console, text editor (notepad++) & terminal; packages downloaden met PIP; verschil Windows, Linux, MacOS met betrekking tot Unix, terminal; algoritmisch denken; Big O() notation.
6		Dictionaries?	design cyclus: schrijf programma in aparte delen, debug, klaar, herhaal; multi-file project;
7			debugging (syntax checker, betekenis error message, debuggen met print statements);
8	Waarheen vanaf hier presentatie: talen en applicaties	Terugblik en vooruitblik: OOP stijl, functionele stijl met maps, imperatieve stijl.	