**Python GGH Lesplan 12 Mei**

**Introductie**
-Nabespreking huiswerk:
-Nabespreking Les 3:
functions: arguments, neven-effecten, return value, local vs global, lists, for, for kopieert argument, slice notation, floating-point arithmetic

**Middenstuk**
Objects: Classes & Instances:
Bevatten attributes en methods zoals list.append().
*OOP: Organisation, encapsulation, hergebruik, data-met-de-functies.*

Class maken met method en attributes:
(Gekopieerd uit Python in easy steps:)

```
class Critter:
    count = 0 # Class variable gedeeld met hele class
    def __init__(self, chat): # 1ste variabele in method: geeft altijd instance aan
        self.sound = chat # Instance variable
        Critter.count += 1 # Update class variable
    def talk(self) # Method
        return self.sound()
```

Constructor is vrijwel noodzakelijk: var = Critter().
Attributes can be added on the run.
Gebruik dir() om alle members, zowel attributes als methods te zien, zie ook class __dict__ dictionary.

Maak een destructor met: __del__ → garbage collection.

Del: op variabelen, op lijsten, op slices van lijsten, op object (members), op functies.
Globals() & Locals() dictionary.

Python datamodel:
Objects zijn abstractie voor data.
Alles is een object in Python: variabelen, instances, classes, functies.
Een object kan meerdere aliassen hebben in verschillende namespaces en wanneer het in bijvoorbeeld een list zit of een attribute is.
Elke object heeft een unieke identiteit met id() en type, met type(), deze zijn onveranderbaar. Je kunt met de is controleren of twee aliassen dezelde identiteit hebben.
De value van een object kan wel veranderen:
Mutable & immutable types: sommige objecten kunnen veranderen, zie lists, maar strings, ins, range niet. Dit heet een mutable of immutable type.
***Elke keer dat je de value van een immutable object updatet, wordt een nieuwe object gecreëerd! Dit is niet zo bij lists.***
Wanneer je een immutable object passt naar een function of aliast, zoals in een for loop, wordt een nieuwe kopie gemaakt (pass-by-value).
Wanneer je een mutable object passt of aliast, blijft een referentie instand (pass-by-reference).
Wil je een lijst volledig kopiëren, moet je een deep copy doen.
Dit is de bron van heel veel bugs.

**Einde les:**
-Bespreek wat we geleerd hebben.
-Als de leerlingen willen bijlezen, kunnen ze het best op https://cscircles.cemc.uwaterloo.ca/nl/ lessen 14, 17 doen.
-Vertel dat we Kivy gaan gebruiken om apps te maken. Vraag ze voor volgende les met meer groepjes en voorstellen te komen.