Python programming - Labo 8

Doel van dit labo

Deze oefeningen zorgen ervoor dat je van start kan gaan met het ontwikkelen in object-georiënteerd Python. Je gebruikt ook eerder opgebouwde kennis van Python.

Oefening 1: Bolletjes ijs

- In deze oefening maak je een klasse aan en noemt ze Bolletje. Bolletje vertegenwoordigt één enkel bolletje ijs
- Elke bol heeft één enkel attribuut, smaak, een string die je kan initialiseren wanneer je een instance van Bolletje initialiseert
- Als je de klasse hebt gemaakt schrijf je een functie - maak_bolletjes - die drie instances van de Bolletje klasse aanmaakt. Elk met een verschillende smaak
- Deze drie instances stop je - binnen de functie - in een lijst met als naam bolletjes
- Vervolgens *loop* je - binnen de functie - door deze lijst en print je de smaak van elke bol ijs die je hebt gemaakt af
- Tot slot roep je deze functie aan

Oefening 2: bolletjes ijs op een hoorntje

- We herbruiken onze klasse Bolletje, die één bol ijs vertegenwoordigt en maken een nieuwe klasse: Hoorntje, waarin we de bollen kunnen onderbrengen. We zullen dit doen met object compositie
- Maak in Hoorntje een attribuut *bolletjes* aan, met een lege list
- Maak in Hoorntje een method *bolletjes_toevoegen* waarmee je - tegelijk - één of meerdere instances van Bolletje kan toevoegen aan Hoorntje (dus geen string met de smaak ;-))
    - Dat betekent gebruik van de *splat operator* (*nieuwe_bolletjes*) ⇒ zie theorieles dictionaries, laatste slides

    - We kunnen dan itereren (in een for loop) over elk element van *nieuwe_bolletjes en het telkens met list.append toevoegen aan self.bolletjes.

- Tot slot print je de smaken af en je definieert hiervoor de __str__ method met daarin gebruik van een for loop door de smaken met str.join (we gaan dieper in op deze “magic method” in de theorieles later deze week). Voorbeeld: https://www.delftstack.com/howto/python/__str__-vs-__repr__-in-python/

Oefening 3: maximum aantal bolletjes

- We werken verder op de vorige oefening
- Je voegt hier een attribuut toe op het niveau van de klasse Hoorntje, *maximum_bolletjes*, zodat maximum drie bolletjes ijs aan een hoorntje kunnen worden toegevoegd
- Je zorgt ervoor dat in *bolletjes_toevoegen* eens er drie bolletjes zijn toegevoegd elk volgend bolletje wordt genegeerd. We doen dat met een *if* statement waarin we het huidige aantal bolletjes vergelijken met ons eerder aangemaakte klasse attribuut

Oefening 4: reuzehoorntje

- Je werkt verder op de vorige oefening en gebruikt overerving om een nieuwe klasse Reuzehoorntje aan te maken
- Het enige verschil met hoorntje: dit hoorntje kan tot 5 bolletjes ijs aan
- Eventuele minimale aanpassingen in Bolletje() en of Hoorntje() zijn toegelaten

Oefening 5: dierentuin

- Voor de volgende drie oefeningen gaan we een reeks klassen definiëren die alles combineren: klassen, methods, attributen, compositie en overerving
- Je bent medewerker van een dierentuin. Deze dierentuin bevat verschillende diersoorten, waarvan sommigen met elkaar een kooi delen
- Elke soort krijgt zijn eigen klasse. Elk object van een bepaalde klasse deelt een *soort* en *aantal_poten* attribuut. Als attribuut kan je per dier ook nog een

bepaalde kleur meegeven. Je kan dus een bepaalde kleur meegeven: oSchaap1 = Schaap('zwart')

- Je maakt vier dierklassen aan: wolf, schaap, slang en papegaai
- Deze klassen erven van de klasse Dier waarin je zoveel mogelijk functionaliteit stopt
- Deze klasse bevat ook een __str__ functie waarmee je de details over het dier kunt rapporteren
- Soort? self.soort = self.__class__.__name__ binnen de klasse Dier

Oefening 6: dierenkooien

- We moeten onze dieren nog van kooien voorzien. Je werkt verder op de vorige oefening en voorziet nu ook een klasse Kooi
- Een kooi heeft een identificatienummer en je kan er zoveel dieren in stoppen als nodig (denk aan de ijsbolletjes en de splat operator)
- Maak een voeg_dieren_toe method aan waarmee je dierobjecten kunt toevoegen (compositie)
- Zorg ook voor een __str__ method

Oefening 7: dierentuin extended

- We werken verder op de vorige oefening en voegen nu ook een Dierentuin klasse toe. Deze klasse zal de volgende functionaliteiten ondersteunen:
  - Een *voeg_kooien_toe* method via dewelke je kooien kunt toevoegen aan de dierentuin (compositie). Je gebruikt hiervoor een attribuut *kooien* (zoals in de voorgaande oefeningen)
  - Een __str__ method via dewelke je de kooien kunt afdrukken en de dieren die erin zitten
  - Een *dieren_met_kleur* method via dewelke je op basis van een kleurkeuze alle dieren met een bepaalde kleur terugkrijgt
  - Een *dieren_met_aantal_poten* method via dewelke je op basis van een aantal poten alle dieren met dit aantal poten terugkrijgt

- Een *totaal_aantal_poten* method via dewelke je het totaal aantal poten van alle dieren in je dierentuin terugkrijgt