Python GGH Workshop 2 (16 April 2021)

Handige Flowchart info: https://en.wikipedia.org/wiki/Flowchart#Building_blocks

Opgave 1: Priemgetallen

Een priemgetal is een getal dat alleen door zichzelf, en 1 deelbaar is.

(a) Schrijf een programma dat controleert of een input priem is. Dit kun je doen door te controleren of het getal n deelbaar is door een van de getallen tussen 2 en n-1.

(b) Als je weet dat en getal door 2 deelbaar is, is het niet priem, tenzij het getal 2 is. Als je eenmaal weet dat een getal niet door 2 deelbaar is, dan hoef je ook niet meer te controleren of het getal deelbaar is door de andere even getallen. Zo kun je sneller zien of een getal priem is: door alleen te controleren of het deelbaar is door oneven getallen.

Schrijf een programma dat gebruik maakt van dit snellere algoritme, en maak een flowchart.

Opgave 2: Collatz

(Zie de numberphile video op: https://www.youtube.com/watch?v=5mFpVDpKX70)

Het vermoeden van Collatz is een zeer simpele, beroemde vermoeden uit de getallentheorie. Het luidt als volgt:

1. Neem een positief, geheel getal n.
2. -Als n even is, deel het door 2.
   -Als n oneven is, vermenigvuldig het met 3 en tel 1 op.
3. Blijf dit herhalen.

Het vermoeden van Collatz stelt je hierbij altijd op 1 uit zult komen. Dit is NOG NIET bewezen! Noch is het tegenbewijs geleverd. Als jij dus een getal kunt vinden, waarvoor dit niet geld, en je bijvoorbeeld in een eeuwige loep terecht komt, dan wordt je beroemd.

Voorbeeld: 12, 6, 3, 10, 5, 16, 8, 4, 2, 1.

Schrijf een programma dat n als invoert neemt, en vervolgens het Collatz algoritme uitvoert en print totdat je bij 1 uitkomt. *Maak eerst een flowchart van je programma voordat je begint.*

EXTRA: Het vermoeden van Collatz geldt niet voor negatieve getallen. Er zijn 3 bekende negatieve getallen tussen de -1 en -20, waarmee je in een cyclus komt. Kun je deze vinden? (*Vind het antwoord op https://nl.wikipedia.org/wiki/Vermoeden_van_Collatz)*

Opgave 3: Errors

Maak drie programmas met elk een ander soort Syntax error.
Maak drie programmas met elk een ander soort Runtime error.
Kun je een programma bedenken met een bug, die geen error geeft?

Opgave 4 (Extra): Logica Oefeningen

Kun je je rekenmachine aanpassen zodat hij in plaats van berekeningen, de logische operaties and, or uitvoert, en als input een 1 (True) of 0 (False) steeds krijgt?

Huiswerk Opgave 1: Rekenmachine Upgrade

(1) Verbeter je rekenmachine uit les 1 zodat hij steeds opnieuw invoer vraagt, tenzij je C invoert, dan stopt het programma.

(2) Kun je de rekenmachine alle berekeningen met invoer, operatie, en uitkomst laten opslaan in een lijst, en dan aan het einde weer laten outputten wat alle ingevoerde berekeningen waren?

Tip 1: De lijst zou bijvoorbeeld als volgt uit kunnen zien:
`berekeningen = [ [7.0, 9.0, '+', 16.0], [2.0, 3.0, '/', 1.5] ].`
Dan is de uitvoer aan het einde:
```
7.0 + 9.0 = 16.0
2.0 / 3.0 = 1.5
```

Tip 2: Een lijst in een lijst kun je als volgt gebruiken:
`berekeningen[1][0] # Dit heeft waarde 3.0`

Huiswerk Opgave 2: Turtle

Turtle is een leuke Python module waarmee je plaatjes kunt maken d.m.v. een 'turtle' die over het scherm beweegt. Controleer of turtle werkt in jouw Python console door `from turtle import *` in te tikken. Als dit werkt zonder problemen, dan werkt turtle, en kun je doorgaan. Als Python een error geeft `'ModuleNotFoundError: No module named 'tkinter'`, dan moet je eerst `tkinter` installeren. Lees daar meer over op (https://tkdocs.com/tutorial/install.html)

Run het volgende programma om te zien wat voor mooie dingen je met turtle kunt doen:

```
from turtle import *
color('red', 'yellow')
begin_fill()
while True:
    forward(200)
    left(170)
    if abs(pos()) < 1:
        break
end_fill()
done()
```

Het belangrijkst zijn de `forward(n), left(d), right(d)` functies, waarmee je respectievelijk n pixels vooruit gaat, of d graden naar links of rechts kunt draaien.

Wat voor mooie programmas en plaatjes kun je maken met turtle? Probeer te de zaken die je geleerd hebt te gebruiken, zoals `while` en `break`. Maak een flowchart van je turtle algoritme.

Als je inspiratie zoekt, maak een programma dat instructies neemt voor turtle, opslaat in een lijst, en dan deze uitvoert. Dat zou er als volgt uit kunnen zien:

```
Stap 1:
Hoeveel pixels vooruit: 10
Hoeveel graden links: 90
Stap 2:
Hoeveel pixels vooruit: 20
Hoeveel graden links: 90
…
Stap 5:
Hoeveel pixels vooruit: 0

KLAAR MET INPUT, IK TEKEN NU MET TURTLE!
<Programma tekent nu met turtle>
```

Tip: als je wilt dat Python een paar seconden wacht, importeer dan de time module als volgt:

```
import time
```

Dan kun je Python 5.5 seconden laten wachten als volgt:

```
time.sleep(5.5)
```

Dit is handig als je je tekening langzamer wilt maken. Vindt meer informatie over turtle op:
https://docs.python.org/3/library/turtle.html
https://www.tutorialspoint.com/turtle-programming-in-python
https://www.youtube.com/watch?v=p7CiFhiTdvY

Nuttige links:

Nuttige online cursus voor leerlingen als referentie en oefening:
https://cscircles.cemc.uwaterloo.ca/nl/
Officiële Python Tutorial (Zeer uitgebreid): https://docs.python.org/3/tutorial/index.html
FreeCodeCamp.org op Youtube voor Python tutorials.

Officiële Python Documentatie: https://docs.python.org/
Algemene vragen: https://stackoverflow.com/
Online Python Interpreter: https://replit.com/languages/python3