Uitwerking opdracht

Opdracht :	*Game of Life*
Weeknummer :	*4*
Studentnummer :	*s1060679*
Naam student :	*Dion van den Berg*
Specialisatie :	*MEDT*
Pogingnummer :	*1*

1. Vraagstelling

Er moet een kleine visualisatie gemaakt worden van cellen die leven of dood gaan.

2. Specificatie

Invoer

De gebruiker geeft een grid aan X en O tjes mee.

Uitvoer

De gebruiker krijgt te zien hoe het grid eruit ziet en een aantal generaties van game of life

Verband tussen in- en uitvoer

De uitvoer komt voort uit het ingevoerde grid aan cellen.

Beperkingen

Coordinaten kunnen verkeerd ingevoerd worden.

Voorbeelden (testscenario's)

```
>>> generatie = [[True] + [False] * 7 for _ in range(6)]
>>> toonGeneratie(generatie)
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O

>>> aantalBuren(generatie, 0, 0)
1
>>> aantalBuren(generatie, 1, 1)
3
>>> aantalBuren(generatie, 2, 2)
0

>>> volgende = volgendeGeneratie(generatie)
>>> toonGeneratie(volgende)
O O O O O O O O
X X O O O O O O
X X O O O O O O
X X O O O O O O
X X O O O O O O
O O O O O O O O

>>> volgende = volgendeGeneratie(volgende)
>>> toonGeneratie(volgende)
O O O O O O O O
X X O O O O O O
O O X O O O O O
O O X O O O O O
X X O O O O O O
O O O O O O O O

>>> volgende = volgendeGeneratie(volgende)
>>> toonGeneratie(volgende)
O O O O O O O O
O X O O O O O O
O O X O O O O O
O O X O O O O O
O X O O O O O O
O O O O O O O O

>>> volgende = volgendeGeneratie(volgende)
>>> toonGeneratie(volgende)
O O O O O O O O
O O O O O O O O
O X X O O O O O
O X X O O O O O
O O O O O O O O
O O O O O O O O
```

3. Ontwerp

Hoe krijg ik het grid binnen ?
Als een string aan X en O tjes

Hoe ga ik een volgende generatie berekenen?
Door te checken wat er om hem heen zit.

Hoe laat ik alles uiteindelijk zien ?
Door het hele grid te printen.

4. Pseudocode

```
Functie aantalBuren ( generatie , x , y )
        Loop door de posities om de cell heen
                Controleer of posities in de cell zitten en of ze een X hebben
                        ja: buren_aantal + 1
        return buren

Functie toonGeneratie ( generatielijst )
        Return Loop door generatielijst en join ze tot een string per rij.

Functie volgendeGeneratie ( generatielijst )
        Nieuwe generatie = loopje van de lijst

        Loop door generatielijst
                Cell is X en aantalBuren is 2 of 3 → levend
                Cell is O en aantalBuren is 3 → levend
        Return nieuwe generatie

Geef generatie op voor het gehele grid
Loop door 4 generaties heen en print ze.
```

5. Code

```python
def aantalBuren(generatie: [[]], x: int, y: int) -> int:

    # Berekent het aantal nabijgelegen levende cellen.

    buren = 0
    # Alle posities om een punt.
    modifiers = [[-1, -1], [-1, 0], [-1, 1],
                 [0, -1], [0, 1],
                 [1, -1], [1, 0], [1, 1]]

    for modifier in modifiers:
        posX = x + modifier[1]
        posY = y + modifier[0]

        # Controleer of een positie binnen het veld valt
        if 0 <= posY < len(generatie) and 0 <= posX < len(generatie[0]):
            # Telt de levende cellen
            if generatie[posY][posX] == "X":
                buren += 1

    return buren


def toonGeneratie(generatie: [[]]) -> str:

    # Geeft het veld op een bepaalde manier weer.
    return "\r\n".join([str(" ".join(row)) for row in generatie])


def volgendeGeneratie(generatie: [[]]) -> [[]]:

    # Berekent de volgende generatie

    # Een veld met dode cellen
    nieuwe_generatie = [["O" for o in range(len(generatie[0]))] for i in
range(len(generatie))]

    for y, row in enumerate(generatie):
        for x, cell in enumerate(row):
            buren = aantalBuren(generatie, x, y)

            # Alleen als een cel 2 of 3 buren heeft
            if cell == "X" and (buren == 2 or buren == 3):
                # Wordt een levende cel terug gezet op het bord
                nieuwe_generatie[y][x] = "X"

            # Of als er nog geen cel is en er 3 buren zijn
            if cell == "O" and buren == 3:
                # Wordt een levende cel terug gezet op het bord
                nieuwe_generatie[y][x] = "X"

    return nieuwe_generatie

# geef alles weer zoals de voorbeeld uitvoer
generatie = [["X", "O", "O", "O", "O", "O", "O", "O"] for x in range(0, 6)]
print(toonGeneratie(generatie))
print(aantalBuren(generatie, 0, 0))
print(aantalBuren(generatie, 1, 1))
print(aantalBuren(generatie, 2, 2))

for i in range(0, 4):
    print(toonGeneratie(generatie) + "\r\n")
    generatie = volgendeGeneratie(generatie)
```

6. Test

Test 1

```
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O
1
3
0
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O
X O O O O O O O

O O O O O O O O
X X O O O O O O
X X O O O O O O
X X O O O O O O
X X O O O O O O
O O O O O O O O

O O O O O O O O
X X O O O O O O
O O X O O O O O
O O X O O O O O
X X O O O O O O
O O O O O O O O

O O O O O O O O
O X O O O O O O
O O X O O O O O
O O X O O O O O
O X O O O O O O
O O O O O O O O
```