Naam: Axel Van Gestel
Bewijsstuk binnen richting 01 - Python

Inleiding:

Als eerste bewijsstuk licht ik een herhalingsoefening toe van Python die ik maakte als voorbereiding voor het examen. Deze oefening was zonder al te veel moeite te maken, het enigste wat moeilijker was een methode vinden om de datums correct te vergelijken.

Het Programma:

Het eerste wat geschreven moest worden waren twee functies die later door het hoofdprogramma opgeroepen kan worden. De eerste hiervan is om het bestand “boomuitleg.txt” uit te lezen en juist te formateren in een dictionary voor optimaal gebruik. In dit geval als:

Boomtype	Karaktertype	Eigenschappen karaktertype

```
def inlezen_bomenuitleg():
    with open("boomuitleg.txt") as bestand_boomuitleg_txt:
        line = bestand_boomuitleg_txt.readline().rstrip()
        split_line = line.split(";")

        dic_boomuitleg = {}

        while line:
            dic_boomuitleg[split_line[0]] = (split_line[1], split_line[2])
            line = bestand_boomuitleg_txt.readline().rstrip()
            split_line = line.split(";")
    return dic_boomuitleg
```

Figure 1 Functie 1: inlezen_bomenuitleg

De tweede functie leest het bestand “bomen.xml” in en controleert of de ingegeven datum valide is. Dit bestand bevat tussen welke datums een boomtype behoort. Deze datums zijn opgesplitst in de aparte dag en maand wat eerst problematisch was.

```
<bomenhoroscoop>
    <boom>
        <dag_begin>1</dag_begin>
        <maand_begin>1</maand_begin>
        <dag_einde>11</dag_einde>
        <maand_einde>1</maand_einde>
        <naam>dennenboom</naam>
    </boom>
```

Figure 2 XML bestand: bomen.xml

Om de datums correct te kunnen vergelijken heb ik ervoor gekozen om de maand en dag te combineren in één getal. Om dit correct te kunnen doen moeten de dagen eenzelfde getal formaat gebruiken, hiervoor controleer ik of elke dag uit twee cijfers bestaan, zo niet dan wordt dat van te voren aangepast via een "{0:0=2d}".format.

Voorbeeld combinatie van het bovenstaande XML bestand:

dag_begin = 01	dag_einde = 11
maand_begin = 1	maand_einde = 1
datum_begin = 101	datum_einde = 111

Daarna wordt de ingegeven dag & maand gecontroleerd en als ie valide is worden ze in hetzelfde formaat als de begin & eind datums geformatteerd.

Voorbeeld combinatie ingegeven datum:

Input_dag = 31
Input_maand = 3
Input_datum = 331

Op deze manier is het zeer gemakkelijk te controleren tussen welke twee begin & eind datums de ingegeven datum licht door te zien of: datum_begin <= input_datum <= datum_einde.

```
14  def zoek_boom(dag, maand):
15      import xml.etree.ElementTree as ET
16      xmldoc = ET.parse("bomen.xml")
17      bomenhoroscoop = xmldoc.getroot()
18
19      for boom in bomenhoroscoop.iter("boom"):
20          # Get begin date from xml file
21          dag_begin = "{0:0=2d}".format(int(boom.find("dag_begin").text))
22          maand_begin = boom.find("maand_begin").text
23          # Get end date from xml file
24          dag_einde = "{0:0=2d}".format(int(boom.find("dag_einde").text))
25          maand_einde = boom.find("maand_einde").text
26
27          # Calculate begin and end date from xml file
28          datum_begin = int(maand_begin + dag_begin)
29          datum_einde = int(maand_einde + dag_einde)
30
31          # Get date from input
32          input_dag = "{0:0=2d}".format(int(dag))
33          if int(input_dag) > 31:
34              return "Onmogelijke dag"
35
36          input_maand = maand
37          if int(input_maand) > 12:
38              return "Onmogelijke maand"
39
40          # Calculate date from input
41          input_datum = int(input_maand + input_dag)
42
43          # Find name for input date
44          if datum_begin <= input_datum <= datum_einde:
45              return boom.find("naam").text
46      print("Geen boom voor gegeven datum. (Foute datum?)")
```

Figure 3 Functie 2: zoek_boom

Na de twee functie kan het hoofdprogramma uitgevoerd worden. Eerst voert het de eerste functie uit om de bomenuitleg in te lezen. Daarna krijg je een welkom scherm en wordt je naam en je verjaardag gevraagd. Met deze gegevens kunnen we de tweede functie uitvoeren om te zien bij welke boomtype we horen.

```
48      # Hoofdprogramma
49      output_dic_bomenuitleg = inlezen_bomenuitleg()
50
51      print("Welkom bij onze bomenastrologie")
52      input_naam = input("Geef je naam: ")
53      print("Wanneer is je verjaardag?")
54      dag = input("dag: ")
55      maand = input("maand: ")
56
57      output_boom = zoek_boom(dag,maand)
58
59      print(input_naam, "jij bent een", output_boom.upper())
60
61      if output_dic_bomenuitleg.get(output_boom):
62          print(output_dic_bomenuitleg[output_boom][0])
63          print(output_dic_bomenuitleg[output_boom][1])
```

Figure 4 Hoofdprogramma

Als de ingegeven datum valide is worden de gevonden boomtype, karaktertype en de eigenschappen ervan afgedrukt. Zo niet wordt er een waarschuwing weergegeven en krijg je terug het welkomscherm te zien.

```
"C:\Users\Axel Van Gestel\Documents\School\Applicatieontwikkeling in Python\Projecten\Hoofdstuk_1\venv\S
Welkom bij onze bomenastrologie
Geef je naam: Bob
Wanneer is je verjaardag?
dag: 31
maand: 03
Bob jij bent een HAZELAAR
De buitengewone
Charmant, niet veeleisend, erg begripvol, weet hoe indruk te maken, actieve strijder voor sociale zaak,

Process finished with exit code 0
```

Figure 5 Resultaten