Examen VMBO-GL

2022

voorbeeldopdrachten MVI – programmeren in Python

opdrachten

Opdracht 1 Teken een complexe vorm

Programmeer een complexe vorm in Python met een Turtle module.

Werkwijze
- Open de Python shell.
- Importeer de Turtle module.
- Maak nu een scherm om op te tekenen.
- Maak de functie jouwster aan. Let op blok code, gebruik de TAB-toets.

stap 1
Voer de volgende codes in:

```
>>> def jouwster(grootte, gevuld):
        if gevuld == True:
    t.begin_fill()
    for x in range(1, 19):
    t.forward(grootte)
    if x % 2 == 0:
    t.left(175)
    else:
    t.left(225)
    if gevuld == True:
    t.end_fill()
```

stap 2
- De for lus ontbreekt. Maak de code compleet.
- De Turtle laten we vervolgens een gouden ster tekenen
  - t.color(0.9, 0.75, 0)
  - jouwster(120,True)
- Teken een zwart lijntje om de ster.
- Sla het bestand op als gouden_ster_jouwnaam.py .

Opdracht 2 Terugkijken met vragen

Wat is de code om het doek te resetten?

Wat is een *for* lus?

Opdracht 3 Alarmsysteem maken

Jij krijgt op je stageadres een opdracht: je gaat een simpel alarmsysteem maken. Op je stage gebruiken ze daarvoor een Raspberry Pi en het programma Python 3. Een collega is al begonnen. Jij gaat het programma aanpassen, omdat er nieuwe eisen zijn bij gekomen.

Werkwijze

Voorbereiding

- Sluit deze onderdelen aan op de Grove Base HAT:
  - op poort D5 de Red LED Button
  - op poort D16 de Mini PIR Motion Sensor
  - op poort D18 de Buzzer
- Sluit daarna de voeding aan op de Raspberry Pi.
- Bekijk deze code:

```
#!/bin/python
import RPi.GPIO as GPIO # importeer de GPIO bibliotheek.
from time import sleep # importeer de time bibliotheek voor tijdfuncties.

GPIO.setmode(GPIO.BCM) # Zet de pinmode op Broadcom SOC.
GPIO.setwarnings(False) # Zet waarschuwingen uit.

led=5 # GPIO pin als LED variabele declareren
pir=16  # GPIO pin als PIR variabele declareren
buzzer=18  # GPIO pin als BUZZER variabele declareren

GPIO.setup(buzzer, GPIO.OUT) # Zet de BUZZER pin als uitgang
GPIO.setup(led, GPIO.OUT) # Zet de LED pin als uitgang
GPIO.setup(pir, GPIO.IN) # Zet de PIR pin als ingang


while True:

    if GPIO.input(pir)== 0: # als GPIO pin van de PIR poort laag (0 = geen beweging) is
        GPIO.output(led, 1) # zet de GPIO pin van de LED poort hoog (1 = licht aan)
        GPIO.output(buzzer,1) # zet de GPIO pin van de BUZZER poort hoog (1 = piepen)
        sleep(0.01) # 1/100 seconde wachten
        GPIO.output(led, 0) # zet de GPIO pin van de LED poort laag (0 = licht uit)
        GPIO.output(buzzer, 0) # zet de GPIO pin van de BUZZER poort laag (0 = piepen uit)
        sleep(2) # 2 seconden wachten
        print("PIR = 0, geen beweging...") # print tekst

    else:
        GPIO.output(led, 1) # zet de GPIO pin van de LED poort hoog (1 = licht aan)
        GPIO.output(buzzer,1) # zet de GPIO pin van de BUZZER poort hoog (1 = piepen)
        sleep(0.01) # 1/100 seconde wachten
        GPIO.output(led, 0) # zet de GPIO pin van de LED poort laag (0 = licht uit)
        GPIO.output(buzzer, 0) # zet de GPIO pin van de BUZZER poort laag (0 = piepen uit)
        sleep(0.1) # 1/10 seconde wachten
        print("PIR <> 0, wel beweging...") # print tekst
```

of

```
#!/bin/python
import RPi.GPIO as GPIO # importeer de GPIO bibliotheek.
from time import sleep # importeer de time biblotheek voor tijdfuncties.

GPIO.setmode(GPIO.BCM) # Zet de pinmode op Broadcom SOC.
GPIO.setwarnings(False) # Zet waarschuwingen uit.

led=5 # GPIO pin als LED variabele declareren
pir=16 # GPIO pin als PIR variabele declareren
buzzer=18 # GPIO pin als BUZZER variabele declareren

GPIO.setup(buzzer, GPIO.OUT) # Zet de BUZZER pin als uitgang
GPIO.setup(led, GPIO.OUT) # Zet de LED pin als uitgang
GPIO.setup(pir, GPIO.IN) # Zet de PIR pin als ingang


while True:

    if GPIO.input(pir)== 0: # als GPIO pin van de PIR poort laag (0 = geen beweging) is
        GPIO.output(led, 1) # zet de GPIO pin van de LED poort hoog (1 = licht aan)
        GPIO.output(buzzer,1) # zet de GPIO pin van de BUZZER poort hoog (1 = piepen)
        sleep(0.01) # 1/100 seconde wachten
        GPIO.output(led, 0) # zet de GPIO pin van de LED poort laag (0 = licht uit)
        GPIO.output(buzzer, 0) # zet de GPIO pin van de BUZZER poort laag (0 = piepen uit)
        sleep(2) # 2 seconden wachten
        print("PIR = 0, geen beweging...") # print tekst

    else:
        GPIO.output(led, 1) # zet de GPIO pin van de LED poort hoog (1 = licht aan)
        GPIO.output(buzzer,1) # zet de GPIO pin van de BUZZER poort hoog (1 = piepen)
        sleep(0.01) # 1/100 seconde wachten
        GPIO.output(led, 0) # zet de GPIO pin van de LED poort laag (0 = licht uit)
        GPIO.output(buzzer, 0) # zet de GPIO pin van de BUZZER poort laag (0 = piepen uit)
        sleep(0.1) # 1/10 seconde wachten
        print("PIR <> 0, wel beweging...") # print tekst
```

Stap 1

- Kies in het menu: Programmeren, Thonny Python IDE
- Kies in Python: Load en laad het bestand GL_alarm.py
- Run het programma.
- Stop het programma als je de werking kent.

Het programma maakt gebruik van deze functie:

```
def alarm(duur,pauze):
        GPIO.output(led, 1)
        GPIO.output(buzzer,1)
        print("alarm duur" ,duur, "pauze" ,pauze )
        sleep(duur)
        GPIO.output(led, 0)
        GPIO.output(buzzer, 0)
        sleep(pauze)
```

of

```
def alarm(duur,pauze):
        GPIO.output(led, 1)
        GPIO.output(buzzer,1)
        print("alarm duur" ,duur, "pauze" ,pauze )
        sleep(duur)
        GPIO.output(led, 0)
        GPIO.output(buzzer, 0)
        sleep(pauze)
```

Stap 2

- Voeg de coderegels van deze functie toe aan het programma.
- Pas het programma zo aan de functie wordt gebruikt.
- Wis de coderegels die overbodig zijn geworden.
- Sla het bestand op als alarm1_jouw naam.py
- Laat aan je docent zien dat het programma werkt.

Stap 3

Als het alarm gaat, moet een noodknop ingedrukt kunnen worden: het programma pauzeert dan vijf seconden en loopt daarna weer door.

- Pas deze dingen aan:
  - Voeg de declaratie *button=6* toe.
  - Voeg een tweede *if…else* toe aan de bestaande *else* sectie.
  - Zorg dat als de button wordt ingedrukt het een 0 geeft op GPIO 6.
  - Zorg dat als de button niet wordt ingedrukt, het een 1 geeft op GPIO 6.
- Pas het programma zo aan dat de coderegels van de noodknop worden gebruikt.
- Sla het bestand op als alarm2_jouw naam.py
- Laat aan je docent zien dat het programma werkt.