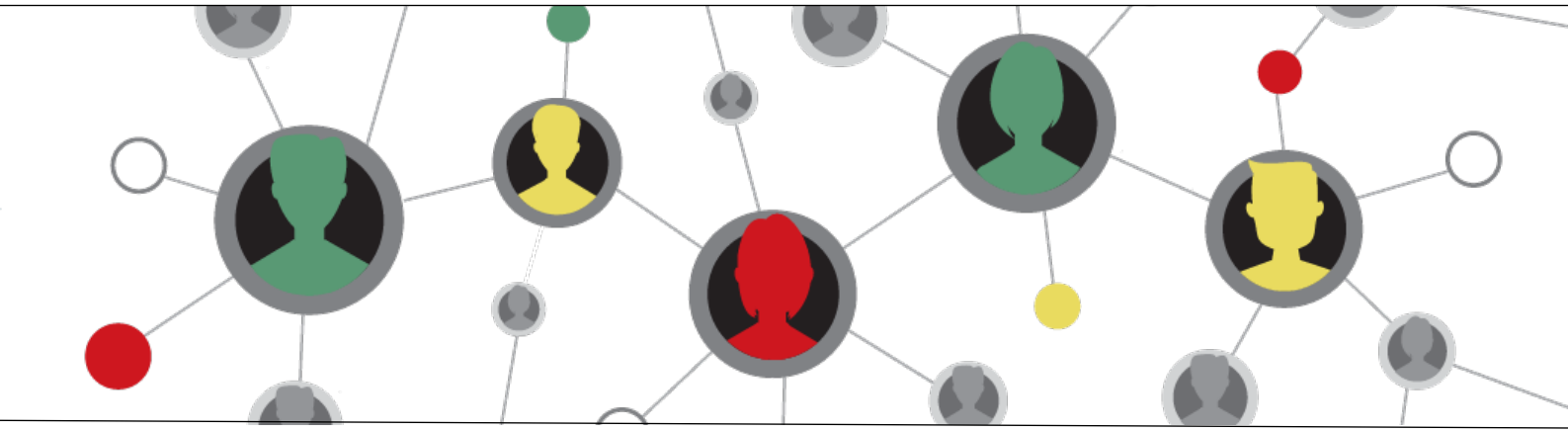
Coderen met Python

Hitte-alarm voor je auto



T³ NEDERLAND

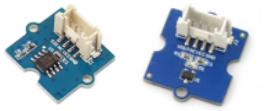
Teachers Teaching with Technology™

T³ VLAANDEREN

Voor dit project bouwen en programmeren leerlingen een eenvoudige feedback- en controlesysteem. Feedback en controle staan centraal in veel industriële systemen en consumentenproducten.

Het hitte-alarmsimulatiesysteem dat we gaan ontwerpen bestaat uit:

- drie input-modules
  - twee temperatuursensoren
  - een Hall-effect magneetveldsensor – simulatie baby/huisdier



- vier outputs
  - twee witte LED-lampjes – simulatie koplampen
  - 1 continue servomotor – simulatie raambediening
  - Luidspreker – simulatie alarm

We coderen het systeem zodat de drie input-parameters gelezen worden en geïnterpreteerd worden i.f.v. kritische waarden om zo te bepalen of het alarm moet ingeschakeld worden.

Het schrijven van de code van het uiteindelijk systeem wordt opgesplitst in een reeks kleinere programmeeropdrachten om stap voor stap vertrouwd te geraken met de Python-syntax die nodig is voor het hele systeem te programmeren.

Wat wetenschappelijke achtergrond.

Licht om te verwarmen
Elke zomer zijn er gruwelijke verhalen over kinderen en huisdieren die in oververhitte auto's achterblijven. Uiteindelijk krijgen de meesten een hitteberoerte en overlijden ze in teveel gevallen.

De binnenkant van een auto warmt door het broeikaseffect veel sneller op dan de buitenkant. Zonnestralen komen door de ramen het voertuig binnen en het licht valt op de oppervlakken van het interieur van de auto.

Het zichtbare licht wordt geabsorbeerd en opnieuw uitgestraald als infrarood licht. Infraroodstraling heeft een grotere golflengte dan zichtbaar licht en kan niet terug ontsnappen via de ramen. Door de ingesloten straling stijgt de temperatuur in de auto veel sneller dan de buitentemperatuur.

Het hoofd koel houden
Zoogdieren zoals mensen, honden en katten hebben manieren om hun lichaamstemperatuur te regelen (thermoregulatie) en hun homeostase onder controle te houden.

Mensen zweten om de afvoer van warmte uit het lichaam te vergroten door middel van verdamping. Honden hijgen meestal om warmte af te voeren, hoewel ze ook een klein aantal zweetklieren in de kussentjes van hun poten hebben.

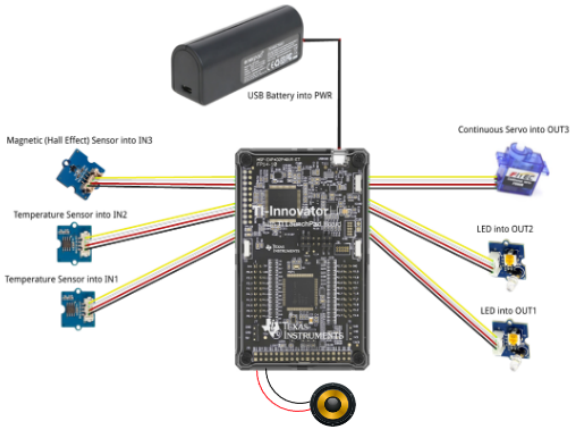
Katten zullen zich uitstrekken op oppervlakken die relatief koel zijn om lichaamswarmte te helpen afvoeren. Ze zullen hun poten likken en het speeksel op warmere delen van hun lichaam wrijven om de verdampingskoeling te vergroten, wat een soortgelijk mechanisme is als zweten bij mensen.

Wanneer het thermoregulatiemechanisme de homeostase niet constant kan houden, zal een zoogdier in hitte-nood komen. Hittestress leidt tot hersenbeschadiging, uitdroging, hartfalen, celzwelling, … en mogelijk tot de dood.

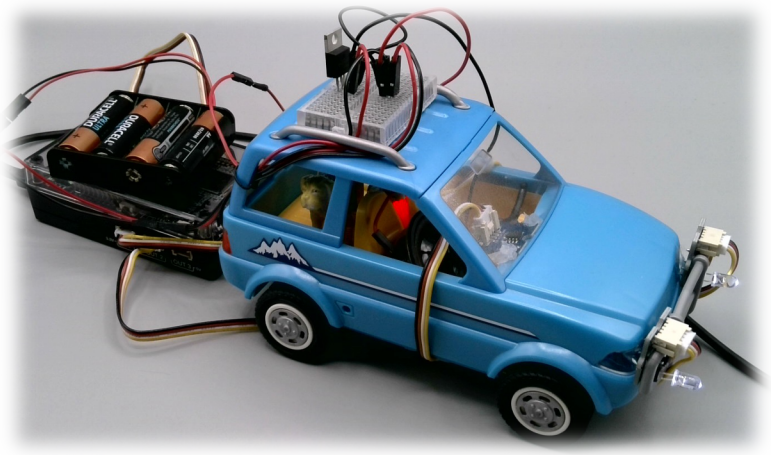
Online is er heel wat data en informatie beschikbaar over waarom zoogdieren zoals mensen, honden en katten niet langdurig in een hete omgeving (zoals gesloten auto's) kunnen overleven. Dit maakt het mogelijk om dit project interdisciplinair uit te breiden.

Het ontwerp van een proto-type van een oplossing met behulp van technologie om auto-inzittenden te beschermen tegen hitte is gebaseerd op de onderstaande configuratie.

Temperatuur buiten	Temperatuur in de auto		
°C	Na 10 min	Na 30 min	Na 60 min
21	31	40	45
24	34	43	47
27	37	45	50
30	40	48	53
32	43	51	56
35	45	54	59



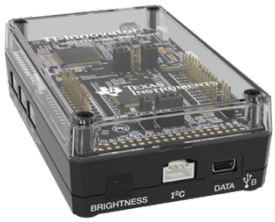
Met de nodige creativiteit en engineering kan het prototype wat meer realistisch gemaakt en verfijnd worden.



Het coderen van dit feedback- en controlesysteem splitsen we op in de volgende deelopdrachten die uiteindelijk leiden tot een simulatie van een hitte-alarm.

- **Challenge 0** – Het schrijven van een Python-programma
- **Challenge 1** – Programmeren van een sirene
- **Challenge 2** – Flikkerende LEDs
- **Challenge 3** – Meten van de temperatuur
- **Challenge 4** – Detectie via een Hall-sensor
- **Challenge 5** – Simulatie van het openen van een venster

Eerst even een introductie van de TI-Innovator™ Hub. De TI-Innovator Hub is een microcontroller-systeem dat klaar is om op een eenvoudige manier in de klas te gebruiken.

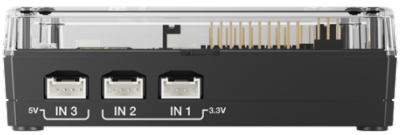


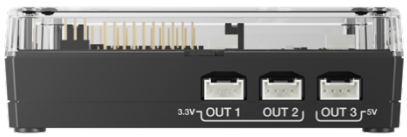
De TI-Innovator is gebaseerd op een evaluatiebord – TI LaunchPad™ – voor de TI-microcontroller MSP432P401R. M.b.v. deze TI LaunchPad-borden testen miljoenen ingenieurs wereldwijd TI MSP432 32 bits microcontrollers voor de ontwikkeling van applicaties.

Door een BoosterPack wordt de TI-Innovator Hub uitgerust met 3 input-poorten en 3 output-poorten voor Grove-apparaten, waaronder tal van sensoren. Grove apparaten worden ontwikkeld door SeeedStudio, www.seeedstudio.com. Meerdere apparaten/sensoren zijn compatible met voorgeprogrammeerde objecten in TI Python.

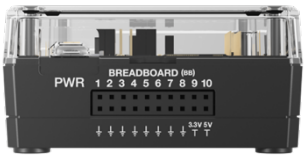
Enkele electronics online stores waar Grove-producten beschikbaar zijn in de Benelux:

- Conrad: www.conrad.be
- KIWI Electronics: www.kiwi-electronics.nl
- DISTRELEC: www.distrelec.be





Bovendien is de TI-Innovator Hub uitgerust met een breadboard connector (om aan de slag te gaan met elektrische circuits), een licht-helderheidsensor, een luidspreker en een I^2C-poort om randapparatuur aan te sluiten gebruikmakend van het I^2C-protocol (zoals de TI-Innovator Hub Rover).



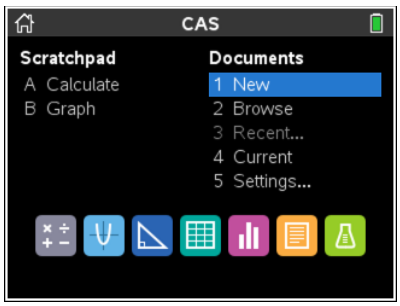
Op het LaunchPad-bord kan bovendien een rode LED en een RGB LED aangestuurd worden.

De DATA-poort (mini USB B) wordt gebruikt om de hub te verbinden met TI-handhelds en computers (Windows & Mac). Via de Micro-USB PWR-poort kan extra vermogen toegevoegd worden met b.v. een powerbank.

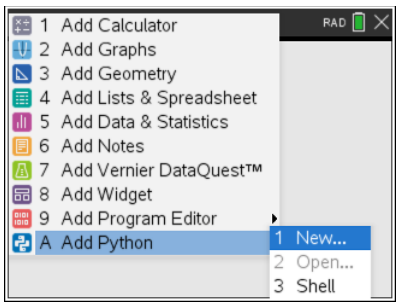
Challenge 0 – Het schrijven van een Python-programma

Een Python-programma schrijven we in een TI-Nspire CX document (tns).

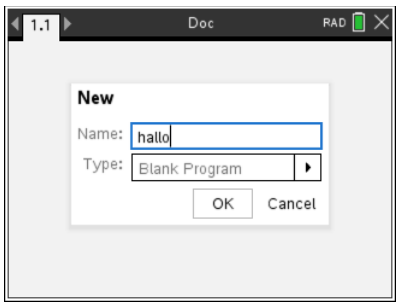
on 1 New.



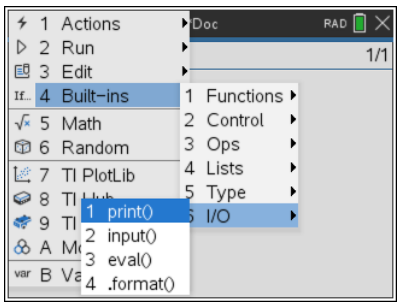
A Add Python > 1 New

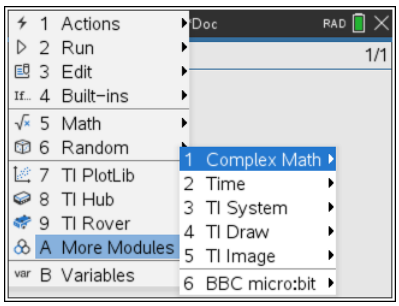


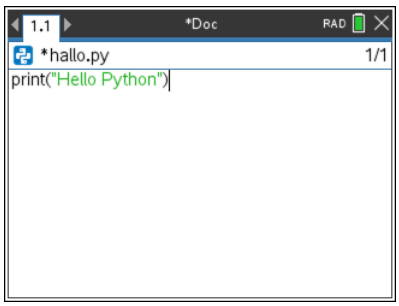
Programmanaam



Via menu kan je via een menustructuur de meest gebruikte commando's, met bijhorende syntax, invoeren in de Python-editor. Het menu geeft ook aan welke Python-modules zijn geïntegreerd voor TI-Nspire CX technologie en ook hier kunnen de gewenste commando's geselecteerd worden en is syntax-hulp beschikbaar.

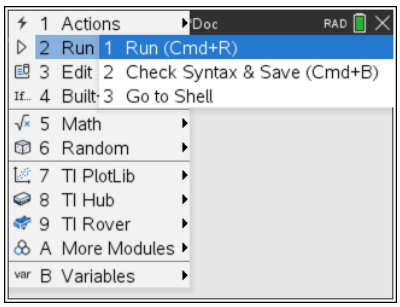


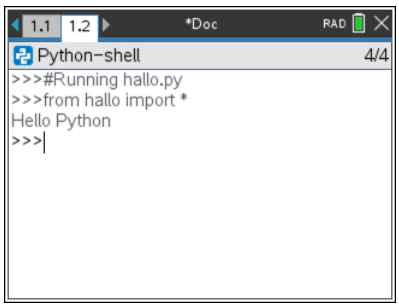




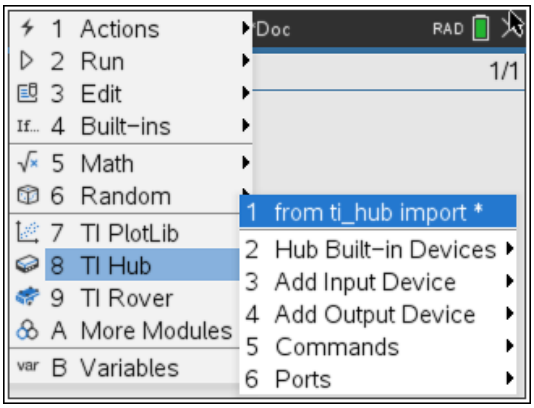
Vanaf versie 5.3.0 (april 2021) zal ook een module toegevoegd kunnen worden die het mogelijk maakt te communiceren met BBC micro:bit.

De asterisk voor de programmanaam geeft aan dat de code nog niet bewaard is. Het runnen van de code (CTRL+R of Cmd+R) checkt de syntax, bewaart de code en voert de code uit in een Python Shell.





Het sturen en lezen van de TI-Innovator Hub in Python is gebaseerd op de TI-module TI Hub waarbij de apparaten/sensoren ingedeeld zijn in de volgende drie categorieën:



- **Ingebouwde apparaten**
  - Rode LED – light
  - RGB LED – color
  - Luidspreker – sound
  - Lichthelderheidsensor – brightness
- **Lezen input**
  - IN 1, IN 2, IN3
  - Breedboard-poorten
- **Sturen output**
  - OUT 1, OUT 2, OUT 3
  - Breed-board-poorten

Het principe voor het programmeren van de TI-Innnovator Hub is vrij eenvoudig. Eerst definiëren we een object voor het aangesloten apparaat en voeren dan beschikbare methodes uit op dit object voor het sturen en lezen van het apparaat.

Voor de ingebouwde apparaten zijn de objecten light, color, sound en brightness voorgedefinieerd.

Enkele voorbeelden van code voor het aansturen van apparaten.

```
from ti_hub import *
```

Ingebouwde rode LED

```
from ti_hub import *

light.on()
sleep(1.5)
light.off()
```

Ingebouwde Luidspreker

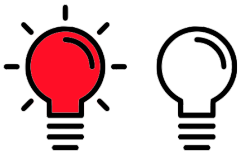
```
from ti_hub import *

# Frequentie 250 Hz - 2 seconden
sound.tone(250,2)
```

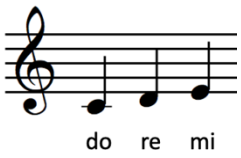
Grove externe LED

```
from ti_hub import *

licht=led("OUT 1")

licht.on()
sleep(1.5)
light.off()
```

Externe Luidspreker

```
from ti_hub import *

geluid=speaker("BB 1")

# Frequentie 250 Hz - 2 seconden
geluid.tone(250,2)
```





Meer info en voorbeelden i.v.m. programmeren in Python met TI-Nspire CX technologie @ www.wil-depython.be en www.wil-destem.be.

Challenge 1 – Programmeren van een sirene

Schrijf een programma dat twee verschillende tonen speelt; telkens 1 seconde voor iedere stap in een lus om dit 5 keer te herhalen.

Dit kan zowel voor de interne speaker als voor een externe speaker aangesloten op een BB-poort.

Interne Luidspreker

```
from ti_hub import *

for i in range(5):
    sound.tone(500,1)
    sleep(1)
    sound.tone(375,1)
    sleep(1)
```

Externe Luidspreker

```
from ti_hub import *

luidspreker=speaker("BB 1")
for i in range(5):
    luidspreker.tone(500,1)
    sleep(1)
    luidspreker.tone(375,1)
    sleep(1)
```

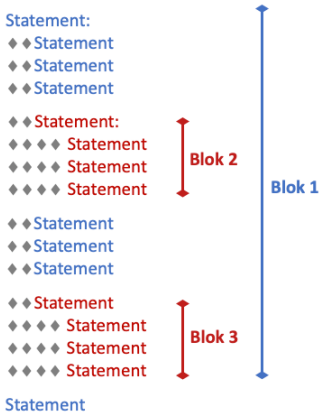
In Nederland is sinds maart 2009 de regel vastgesteld dat alle voorrangsvoertuigen een tweetonige hoorn voeren met frequenties 375 Hz en 500 Hz.

TI Python tips

- De bovenstaande for-lus wordt 5 keer doorlopen, van i = 0 t.e.m. i = 4 m.a.w. 5 niet inbegrepen. De volgende syntax is beschikbaar voor een for-lus via menu > 4 Built-ins > 2 Control; telkens de stop-waarde niet inbegrepen.

```
4 for index in range(size):
5 for index in range(start, stop):
6 for index in range(start, stop, step):
7 for index in list:
```

- Merk op dat de for-lus niet wordt afgesloten met een end-statement. De statements na het dubbele punt - die 5 keer worden uitgevoerd - moeten in een blok staan. In Python-termen geven spaties aan tot welk blok statements behoren. Begin en einde van een blok is gebaseerd op gelijke inspringingen. In TI-Technologie wordt een spatie aangeduid met ♦.



- Het sleep-commando tussen de twee tonen zorgt ervoor dat je de eerste toon hoort zonder dat de tweede instructie de eerste stopt vooraleer uitgevoerd te zijn.

Challenge 2 – Flikkerende LEDs

Schrijf een programma dat twee grove-LEDs laat flikkeren, aangesloten op OUT 1 en OUT 2. Hiermee simuleren we het aan en uit gaan van de koplampen van een auto.

on() / off()

```
from ti_hub import *

led1=led("OUT 1")
led2=led("OUT 2")

for i in range(10):
    led1.on()
    led2.on()
    sleep(1)
    led1.off()
    led2.off()
    sleep(0.5)
```

blink()

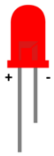
```
from ti_hub import *

led1=led("OUT 1")
led2=led("OUT 2")

for i in range(10):
    led1.blink(2,10)
    led2.blink(2,10)
```

TI Python tip

- De syntax van de blink-methode is object.blink(frequency,time) met
  - frequency: 0.1 – 20 Hz
  - time: 0.1 – 100 s

  Met de methode blink() is het makkelijker om de LEDs gelijktijdig te laten flikkeren waar met on/off er een klein tijdverschil tussen het aan- en uitgaan van de LEDs.

- Bij gebruik van losse LEDs (i.p.v. Grove LEDs) aangesloten op de breadboard-poort, verander je de string met de poortaanduiding door b.v. led1=led("BB 1").



Met de combinatie van een for-lus en een lijst programmeer je vrij eenvoudig een looplichtje.

```
from ti_hub import *
led1=led("BB 1")
led2=led("BB 2")
led3=led("BB 3")
led4=led("BB 4")
ledarray=[led1,led2,led3,led4]
for i in ledarray:
    i.on()
    sleep(0.5)
    i.off()
```

Challenge 3 – Meten van de temperatuur

Schrijf een programma dat de temperatuur meet en weergeeft totdat we dat esc-knop indrukken. Sluit de temperatuursensor aan op poort IN 1.

Meet & Print

```
from ti_hub import *

temp=temperature("IN 1")

t=temp.measurement()
print("Temperatuur = ",t)
```

Meet & Print tot esc

```
from ti_hub import *

temp=temperature("IN 1")

while get_key() != "esc":
    t=temp.measurement()
    print("Temperatuur = ",t)
    sleep(0.5)
    get_key()
```

Basic output-dashboard

```
from ti_hub import *

temp=temperature("IN 1")

while get_key() != "esc":
    t=temp.measurement()
    text_at(7,"Temperatuur = {}".format(t),"center")
    sleep(0.5)
    cls()
    get_key()
```

TI Python tips

- Het get_key()-commando checkt welke toets er wordt ingedrukt van de handheld of het toetsenbord van een computer: b.v. “0”, “a”, ”esc”, “enter”, ...

  Met “left”, “right”, “up”, “down” worden de pijltjestoetsen gedetecteerd en met “center” een muisklik of het indrukken van de touchpad van de handheld.

  Het blok van de while-lus wordt uitgevoerd totdat de esc-toets wordt ingedrukt.
- De stringmethode format() maakt het mogelijk om o.a. numerieke waarden toe te voegen aan een string (of woord).
- Met het statement text_at() van de TI Hub-module kan tekst getoond worden in het grafische venster van de Python Shell. Er zijn 13 lijnen/rijen beschikbaar.
- Met de TI Draw-module (www.wil-depython.be) kunnen meer geavanceerde en grafisch dashboard geprogrammeerd worden.
- In een logische, booleaanse uitdrukking betekent
  - != verschillend van
  - == gelijk aan

Challenge 4 – Detectie van een passagier

Een Hall-sensor checkt of de zuidpool van een magneet dicht bij de sensor is. We simuleren de passagier met een magneet.

Schrijf een programma dat weergeeft of een magneet aanwezig of niet, gebruikmakend van de meetwaarde van de Hallsensor. Sluit de Hallsensor aan op poort IN 3.

De Hall-sensor heeft minimaal een voedingsspanning van 3.8V nodig en wordt meestal aangesloten op een poort met 5V; wat voor de TI-Innovator Hub poort IN 3 (5V) is.

measurement()

```
from ti_hub import *
mag=magnetic("IN 3")
while get_key() != "esc":
    m=mag.measurement()
    text_at(6,"Hallsensor = {}".format(int(m)),"center")
    if m<100:
        text_at(7,"Magneet aanwezig","center")
    else:
        text_at(7,"Magneet niet aanwezig","center")
    sleep(0.5)
    cls()
    get_key()
```

magnet_close()

```
from ti_hub import *
mag=magnetic("IN 3")
while get_key() != "esc":
    m=mag.magnet_close()
    text_at(6,"Magneetdetectie","center")
    text_at(7,"----------","center")
    if m==1:
        text_at(7,"Magneet aanwezig","center")
    else:
        text_at(7,"Magneet niet aanwezig","center")
    sleep(0.5)
    cls()
    get_key()
```

TI Python tip

Met if-statements kunnen voorwaarden gecontroleerd worden en beslissingen genomen worden op basis van waargenomen waarden.

De volgende structuren van conditionele if-statements zijn beschikbaar via menu:

- if BooleanExpression:
  ♦♦Block
- if BooleanExpression:
  ♦♦Block
  else:
  ♦♦Block
- if BooleanExpression:
  ♦♦Block
  elif BooleanExpression:
  ♦♦Block
  else:
  ♦♦Block

4 Built-ins > 2 Control

1 if..
2 if..else..
3 if..elif..else..
4 for index in range(size):
5 for index in range(start, stop):
6 for index in range(start, stop, step):
7 for index in list:
8 while..

Challenge 5 – Roteren van een servomotor

Schrijf een programma dat een continue servomotor eerst in wijzerzin en dan in tegenwijzerzin laat roteren. Sluit de servomotor aan op poort OUT 3.

Hiermee simuleren we het openen en sluiten van de vensters van de auto.

Voor het aansturen van de servomotor is er extra vermogen nodig bovenop wat de TI-Innovator Hub kan leveren. Het extra vermogen kan toegevoegd via de Micro-USB PWR-poort met b.v. een USB-batterij of door de Micro-USB PWR-poort te verbinden met een USB-poort van een computer.

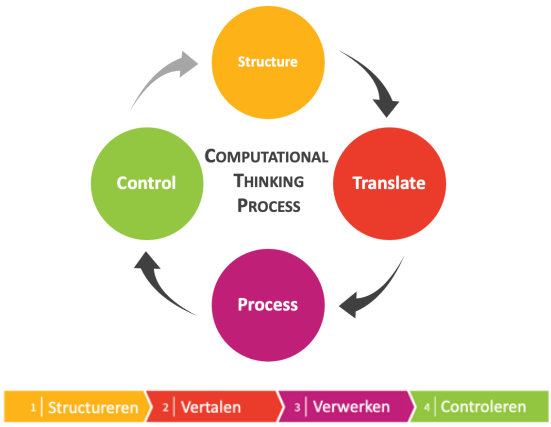
```
from ti_hub import *
venster=continuous_servo("OUT 3")
for i in range(3):
    venster.set_cw(100,2)
    print("Openen")
    sleep(2)
    venster.set_ccw(100,2)
    print("Sluiten")
    sleep(2)
```

Hitte-Alarmcode

Vooraleer de simulatie van een hitte-alarm te bespreken eerst even kort wat over Computational Thinking.

We kunnen Computational Thinking als volgt beschrijven:

> *Computational Thinking is een denkproces voor het (her)formuleren van problemen op een zodanige manier dat het mogelijk wordt om ze met ICT-technieken en technologie op te lossen.*

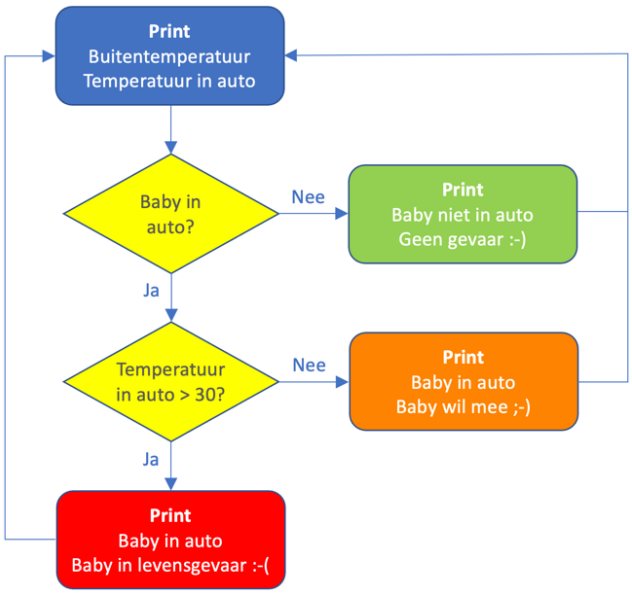


Voor de simulatie van een hitte-alarm kunnen we dit proces opspitsen in de volgende stappen:

- Structureren = het algoritmitiseren van het probleem
- Vertalen = het programmeren van het algoritme in Python
- Verwerken = het runnen van de Python code
- Controleren = de output controleren en eventueel het algoritme en/of de code aanpassen

Structureren

Het hart van de simulatie kan gebaseerd zijn op het volgende algoritme, een while-lus die start met het blijven uitvoeren van een groep van statements zolang de auto op slot is en we afbreken met het indrukken van de knop “esc”.



Vertalen

Dit algoritme kan als volgt geprogrammeerd worden in Python.

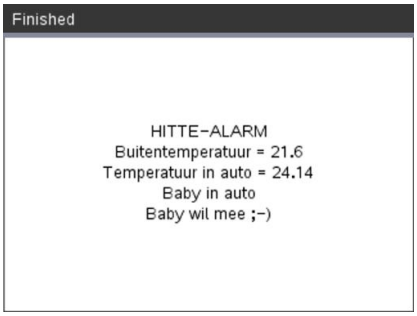
```
from ti_hub import *
geluid=speaker("BB 1")
led1=led("OUT 1")
led2=led("OUT 2")
temp1=temperature("IN 1")
temp2=temperature("IN 2")
mag=magnetic("IN 3")
while get_key() != "esc":
    t1=round(temp1.measurement(),2)
    t2=round(temp2.measurement(),2)
    m=mag.magnet_close()
    text_at(5,"HITTE-ALARM","center")
    text_at(6,"Buitentemperatuur = {}".format(t2),"center")
    text_at(7,"Temperatuur in auto = {}".format(t1),"center")
    if m != 1:
        text_at(8,"Baby niet in auto","center")
        text_at(9,"Geen gevaar :-)","center")
    else:
        text_at(8,"Baby in auto","center")
        if t1 > 25:
            text_at(9,"Baby in levensgevaar :-(","center")
            led1.blink(2,2)
            led2.blink(2,2)
            geluid.tone(440,1)
            sleep(0.5)
            geluid.tone(880,0.5)
            sleep(0.5)
        else:
            text_at(9,"Baby wil mee ;-)","center")
    sleep(1)
    cls()
```

Verwerken

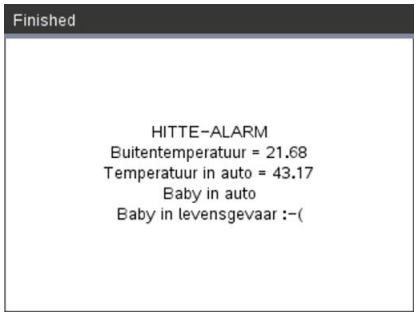
De bovenstaande code geeft de volgende output, afhankelijk van de ingelezen waarden van de sensoren.

m=0

m=1 & t1<=30



m=1 & t1>30



Controleren

Na het runnen van de code, kunnen o.a. de volgende problemen optreden:

- Foute boodschappen worden weergegeven of boodschappen overlappen,
- Bepaalde sensoren meten niet en/of resultaten worden niet/verkeerd weergeven,
- Het alarm (geluid/lampen) werkt niet zo alles gepland,
- ...

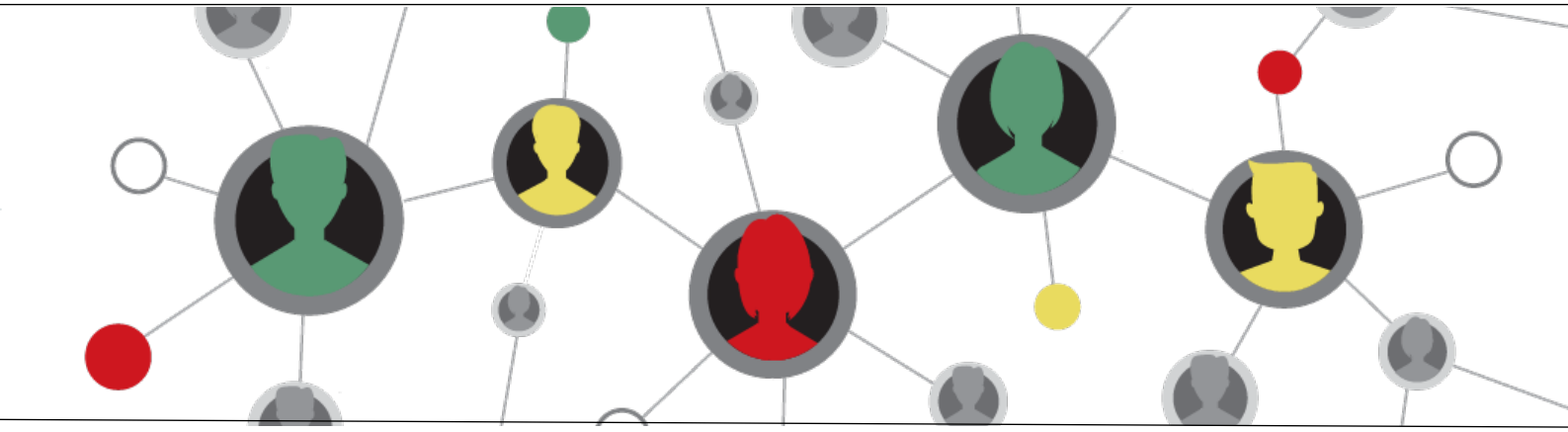
Afhankelijk van de resultaten van de controle dient het algoritme en/of de code aangepast.

Extra programmeeropdrachten

- Voeg een continue servomotor toe aan de code van het hitte-alarm om als volgt een venster te simuleren:
  - Definieer een controlevariabele om te checken of het venster open (1) of toe (0) is.
  - Simuleer het openen van het venster met een rotatie in wijzerzin.
  - Simuleer het sluiten van het venster met een rotatie in tegenwijzerzin.
  - Codeer een check of het venster open of gesloten is en geef dit tekstueel weer.
  - Indien er geen baby in de auto is en het venster is open, sluit het venster.
  - Indien een baby in de auto is en de temperatuur hoger dan 25 °C is, open het venster.
  - Indien bij het afsluiten van het alarm het venster open is, sluit het venster

- Programmeer/Simuleer een parkeeralarm met een Grove Ultrasonic Ranger.
  Hoe kleiner de afstand tussen een object en de Ranger:
  - hoe hoger de toon,
  - hoe sneller de pieptoon
  - Code:

```
dist=ranger("IN 1")
afstand=dist.measurement()
```



T³ NEDERLAND

T³ VLAANDEREN

Texas Instruments