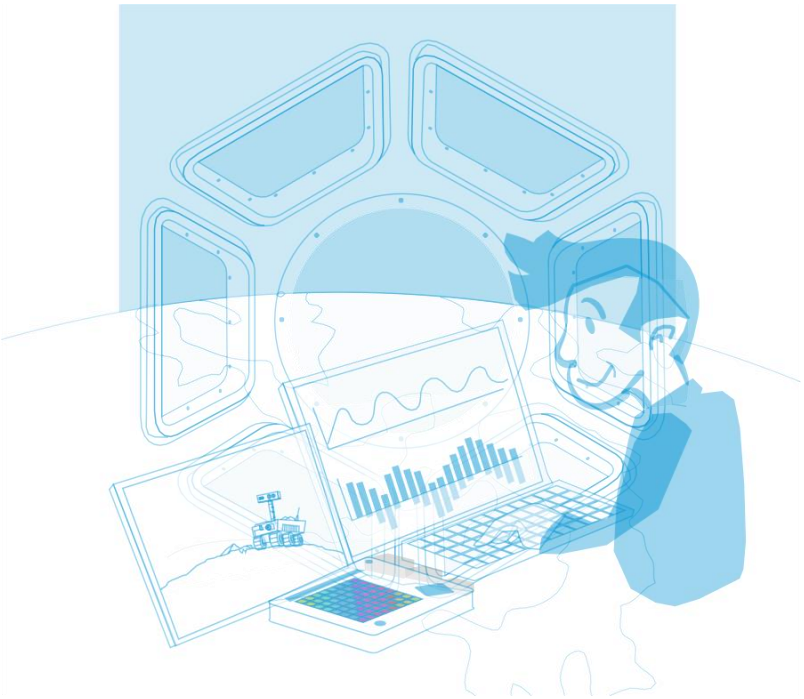
Lehren mit dem All

→ DATENERFASSUNG MIT DEM ASTRO PI

Erfassung von Umgebungsdaten mithilfe von Sense-HAT-Sensoren



Lehren mit dem All – Datenerfassung mit dem Astro Pi | T05.3b
www.esa.int/education

Eine ESA Education Zusammenarbeit mit der Raspberry Pi Foundation, der UK Space Agency, ESERO Poland und ESERO UK

Überarbeitet von ESERO Germany

→ DATENERFASSUNG MIT DEM ASTRO PI

Erfassung von Umgebungsdaten mithilfe von Sense-HAT-Sensoren

Währen der European Astro Pi Challenge wird Astro Pi Ed an Bord der Internationalen Raumstation ISS mit seinen Sensoren eine Reihe von Daten erfassen.
In dieser Lektion werdet ihr die Lebensbedingungen an Bord der ISS untersuchen und sie mit den Bedingungen auf der Erde vergleichen. Dazu werdet ihr mit den Sensoren des Sense-HAT verschiedene Werte eurer Umgebung messen.

Abbildung A1

↑ Astro Pi Logo

Versuchsmaterial

- Astro-Pi-Bausatz
- Monitor
- USB-Tastatur
- USB-Maus

→ AUFGABE 1 – COOL BLEIBEN AUF DER ISS

Für die Astronauten ist es enorm wichtig, dass die Temperatur im Inneren der ISS konstant bei 24 °C liegt - doch das ist leichter gesagt, als getan. Auf der Sonnenseite der ISS herrschen Temperaturen von bis zu 121 °C, auf der Schattenseite können sie aber bis auf −157 °C sinken! Anhand der Sense-HAT-Sensoren werdet ihr die Temperatur in eurem Klassenzimmer messen und sie mit der Temperatur im Columbus-Modul der ISS vergleichen.

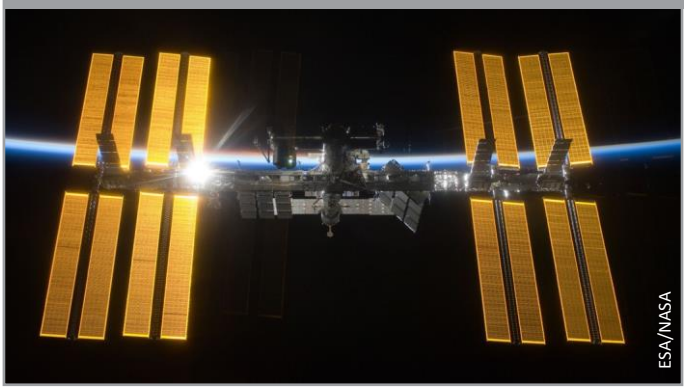
Übung

1. Warum ist es eurer Meinung nach wichtig, dass die Temperatur im Inneren der ISS immer um die 24 °C liegt?

_____

_____

Abbildung A2



↑ Internationale Raumstation (ESA/NASA)

2. Öffnet Python 3, indem ihr oben am Bildschirm auf das Himbeer-Symbol klickt. Dadurch öffnet sich das Menü. Wählt Programming > Python 3 [Programmierung > Python 3]. Daraufhin wird das „Python Shell"-Fenster angezeigt. Klickt auf File > New File [Datei > Neue Datei] und schreibt den nachstehenden Code in das neue Fenster.

   Hinweis: Die Sätze, die mit # beginnen, sind nur Kommentare. Diese werden vom Programm nicht ausgeführt, deshalb braucht ihr sie nicht zu übernehmen.

```
File  Edit  Format  Run  Options  Windows  Help
#to allow the program to use the Sense HAT hardware
from sense_hat import SenseHat

#to create a sense object which represents the Sense HAT
sense = SenseHat()

#to collect temperature and store it as temp
temp = sense.get_temperature()

#to round the value of the temperature with two decimal points
temp = round(temp, 2)

print (temp)
```

3. a. Wählt File > Save As [Datei > Speichern als] und gebt eurem Programm einen Dateinamen. Klickt dann auf Run > Run module [Ausführen > Modul ausführen]. Notiert die gemessene Temperatur.

_____

b. Messt mit einem Thermometer die Temperatur im Klassenzimmer, um festzustellen, wie genau Astro Pi misst. Welche Temperatur steht auf dem Thermometer?

_____

4. Die in Tabelle A1 angegebenen Temperaturen, wurden von einem der Astro-Pi-Computer im Columbus-Modul der ISS erfasst.

Tabelle A1

Versuchsdaten aus dem Columbus-Modul		Versuchsdaten aus eurem Klassenzimmer	
Temperatur (°C)	**Datum und Uhrzeit**	**Temperatur (°C)**	**Datum und Uhrzeit**
27,53	16.02.16, 10:45 Uhr		
27,52	16.02.16, 10:45 Uhr		
27,54	16.02.16, 10:45 Uhr		
27,55	16.02.16, 10:45 Uhr		
27,53	16.02.16, 10:45 Uhr		
27,55	16.02.16, 10:45 Uhr		
27,54	16.02.16, 10:46 Uhr		
27,54	16.02.16, 10:46 Uhr		
27,53	16.02.16, 10:46 Uhr		
27,52	16.02.16, 10:46 Uhr		
27,53	16.02.16, 10:46 Uhr		
27,53	16.02.16, 10:46 Uhr		

↑ Ein Temperaturvergleich zwischen dem Innern der ISS und eurem Klassenzimmer.

a) Fügt eurem Code eine „while True“-Schleife hinzu. So könnt ihr kontinuierlich Daten vom Sensor ablesen. Euer Code sollte nun in etwa so wie der in nachstehendem Kästchen lauten.

File Edit Format Run Options Windows Help

```
from sense_hat import SenseHat

#to allow the program to use the time module
import time

sense = SenseHat()

#to repeat the code until a condition is met
while True:
    temp = sense.get_temperature()
    temp = round(temp, 2)
    #_____
    time.sleep(10)

    #the strftime function is used to print the date and time
    print (time.strftime('%x %X'),temp)
```

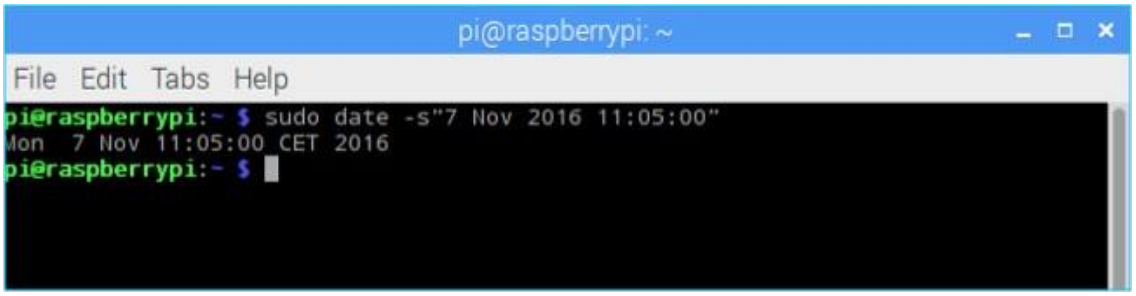
b) Schreibt einen Kommentar in euren Code, um den Befehl time.sleep(10) zu erläutern.

c) Raspberry Pi verfügt nicht über eine Echtzeituhr. Um die genaue Uhrzeit ablesen zu können, müsst ihr das System direkt ansteuern. Öffnet ein Terminal-Fenster (Kommandozeilen-Fenster), indem ihr oben am Bildschirm auf das entsprechende Symbol klickt. Daraufhin sollte sich ein neues Fenster mit folgender Eingabeaufforderung öffnen:

pi@raspberrypi ~ $

Schreibt folgenden Befehl (siehe Beispiel in nachstehender Abbildung) und drückt die Eingabetaste:

sudo date -s "Tag Monat Jahr hh:mm:ss"



Hinweis: Eventuell müsst ihr das Terminal-Fenster schließen, um die aktuelle Zeit sehen zu können.

d) Kehrt zurück zum Editor-Fenster und führt euren Code aus. Vervollständigt Tabelle A1 mit den ausgegebenen Daten. Stoppen könnt ihr das Programm mit der Tastenkombination Strg + C.

e) Berechnet anhand der Daten aus Tabelle A1 die Durchschnittstemperatur im Columbus- Modul und die Durchschnittstemperatur in eurem Klassenzimmer. Liegen sie ungefähr bei 24 °C? Warum, glaubt ihr, weichen sie von diesem Wert ab?

Weiterführende Aufgabe

Das Weltall ist ein Ort vieler Extreme. Wie ist es eurer Meinung nach möglich, die Temperatur im Inneren der ISS zu steuern? Plant ein Experiment, um zu bestimmen, welche Materialien verwendet werden könnten, um für die Astronauten die richtige Temperatur zu halten. Dabei könnt ihr euch an nachstehender Vorlage orientieren.

Vorlage für euer Experiment:

1. Formuliert eine Forschungsfrage zum Untersuchungsgegenstand. Stellt eine Hypothese auf, die es zu untersuchen gilt.

   Forschungsfrage:

   Hypothese:

2. Entwickelt einen Plan, wie ihr eure Hypothese mithilfe des Astro Pi überprüfen wollt. Welche weiteren Materialien benötigt ihr?

3. Erfasst und analysiert eure Daten.

4. Zieht Schlüsse und versucht, die Forschungsfrage zu beantworten.

→ AUFGABE 2 – REGELUNG DER LUFTFEUCHTIGKEIT IM INNEREN DER ISS

Auch wenn man es nicht sieht, enthält die Luft um uns herum Wasser. Die Luftfeuchtigkeit ist ein Maß, das angibt, wie viel Wasserdampf sich in der Luft befindet. In der Regel wird sie in der Einheit „Prozent relative Luftfeuchtigkeit“ angegeben. 100 Prozent relative Luftfeuchtigkeit bedeutet, dass die Luft bei der jeweiligen Temperatur vollständig mit Wasserdampf gesättigt ist.

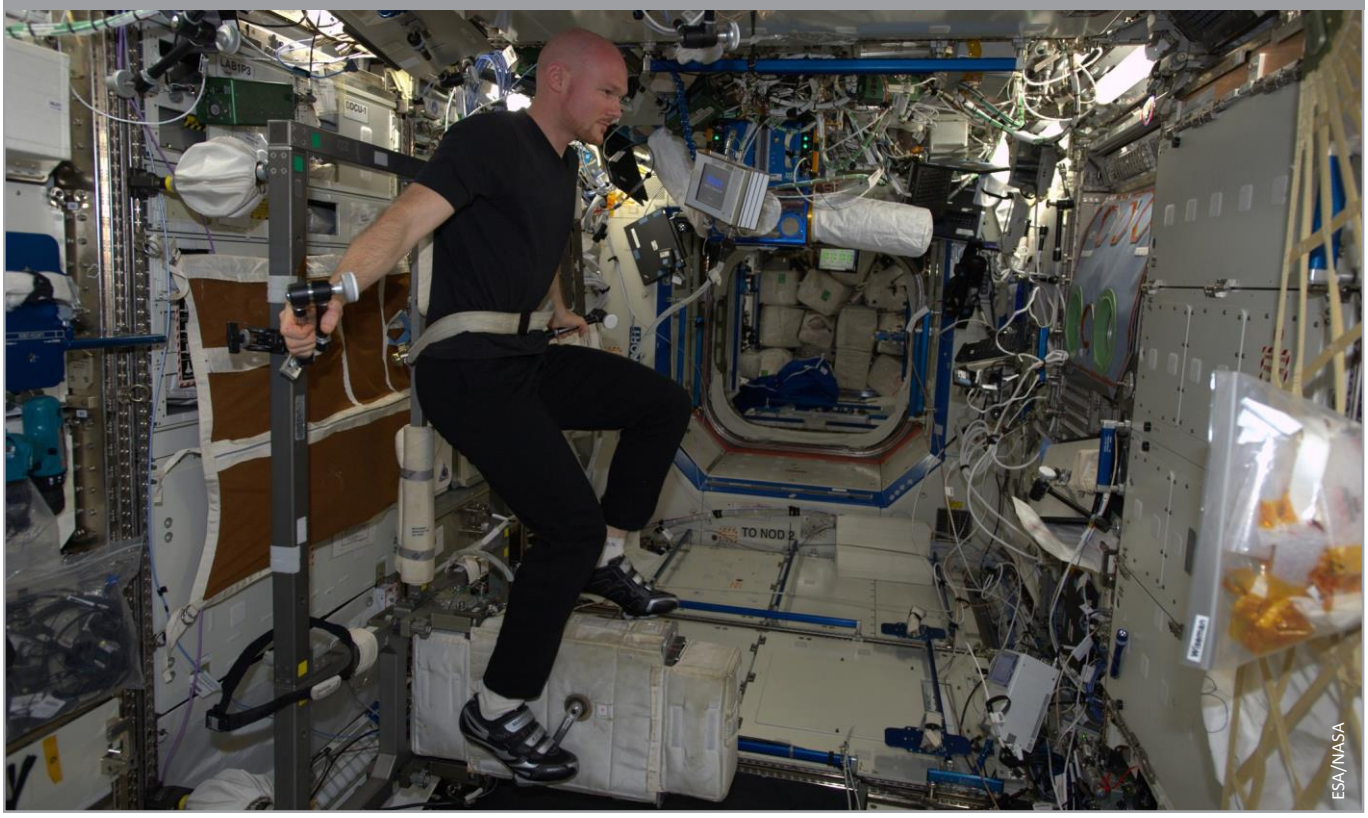
Bei dieser Aufgabe werdet ihr mit dem Astro Pi das Luftfeuchte-Regelungssystem simulieren, das auf der ISS zum Einsatz kommt und lernen, wie ihr die von den Astro-Pi-Sensoren erfassten Daten weiterleiten könnt.

Aufgabe 2.1 – Luftfeuchtigkeit messen

Die Luftfeuchtigkeit auf der ISS wird in der Regel bei etwa 60 % gehalten, doch leicht ist das nicht. Die alltäglichen Routinetätigkeiten von Menschen führen dazu, dass die Luftfeuchtigkeit in der ISS immer wieder ansteigt. Außerdem trinkt und nimmt ein Astronaut mit seiner Nahrung pro Tag ca. 2,7 Liter Wasser auf. Ein Teil dieses Wassers wird durch Verdunstung aus dem Körper ausgeschieden (durch die Poren der Haut oder die Atmung).

Steigt die Luftfeuchtigkeit zu stark an, sorgt das Lebenserhaltungssystem der ISS dafür, dass der überschüssige Wasserdampf herausgefiltert wird. Dazu wird die Luftfeuchte kontinuierlich von präzisen Sensoren in der ISS überwacht. In dieser Übung werdet ihr mit dem Luftfeuchtesensor des Astro Pi den Luftfeuchtigkeitsgehalt in eurem Klassenzimmer messen, so wie auf der ISS.

Abbildung A3



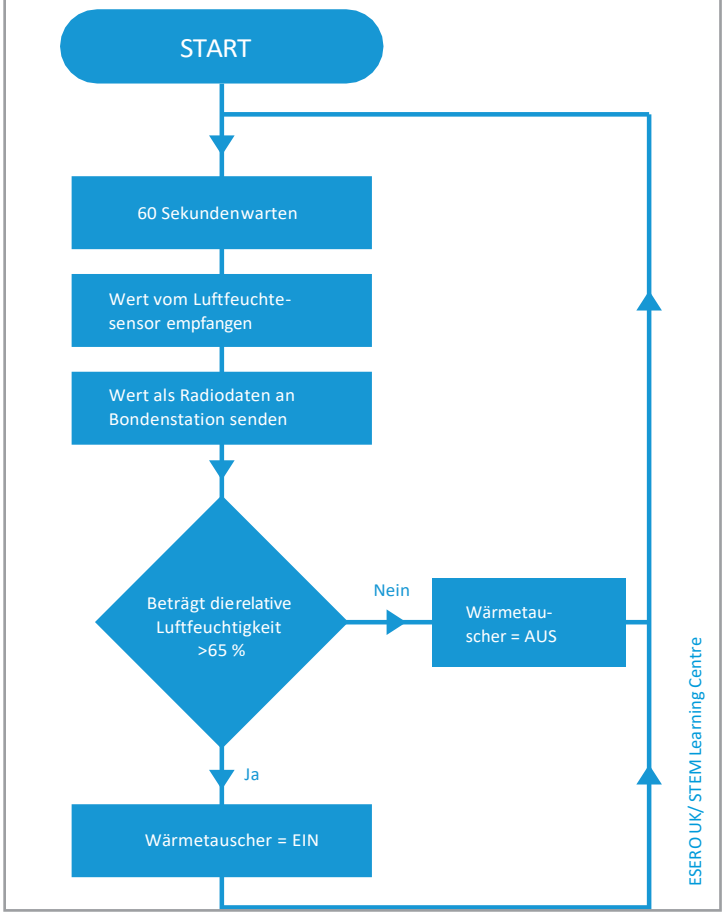
↑ ESA-Astronaut Alexander Gerst in der Internationalen Raumstation auf dem Fitnessbike. Alltägliche Routinetätigkeiten wie Sport führen dazu, dass die Luftfeuchtigkeit in der ISS immer wieder ansteigt.

Übung

1. Warum ist es eurer Meinung nach wichtig, die Luftfeuchtigkeit im Inneren der ISS zu regulieren? Diskutiert diese Frage mit euren Mitschülerinnen und Mitschülern und nennt zwei Gründe für den Einsatz eines Luftfeuchte-Regelungssystems.

2. Abbildung A4 zeigt ein Flussdiagramm zum Luftfeuchte-Regelungssystem der ISS. Schreibt in dem nachstehenden Kasten ein kurzes Python-Programm, mit dem die ersten drei Schritte des Flussdiagramms ausgeführt werden.

Abbildung A4



↑ Luftfeuchte-Regelungssystem: Liegt die relative Luftfeuchte über 65 %, kühlen und trocknen eine Reihe von Lüftern und Wärmetauschern (ähnlich wie die in einem Kühlschrank) die Luft, um den Wasserdampf zu reduzieren. (ESERO UK/STEM Learning Centre)

3. Öffnet ein neues Python-3-Fenster und schreibt euren Code. Wählt File > Save As [Datei > Speichern als] und gebt eurem Programm einen Dateinamen. Klickt dann auf Run > Run module [Ausführen > Modul ausführen]. Notiert die gemessene Luftfeuchtigkeit.

Aufgabe 2.2 Daten erfassen und senden

Ein weltweites Netzwerk aus Kontrollzentren auf der Erde unterstützt die Astronauten, die auf der ISS leben und arbeiten, unter anderem durch die Überwachung der Lebensbedingungen an Bord der Raumstation. Dazu ist es sehr wichtig, die gewonnenen Daten weiterzuleiten. Ihr werdet nun Schritt 4 des Flussdiagramms aus Abbildung A4 nachstellen und die erfassten Daten als CSV-Datei (CSV = comma separated value; durch Trennzeichen getrennte Dateien) senden, die ihr dann analysieren könnt.

Übung

1. Die auf der ISS erfassten Daten werden als Radiodaten zur Erde geschickt. Schreibt auf, was ihr unter „Radiodaten“ versteht.

2. Die Daten von der ISS werden im CSV-Format gesendet, einem Format, mit dem sich Daten in Tabellenform speichern lassen und das sich von Wissenschaftlern leicht auswerten lässt. Öffnet ein neues Python-Shell-Fenster und gebt nachstehenden Code ein.

File Edit Format Run Options Windows Help

```
from sense_hat import SenseHat
import time
sense = SenseHat()

#to open a file named Datafile in which the program will add the collected data
file = open("Datafile.csv", "a")

#to write in the file the name of the table columns. \n is used to break and to create a new line
file.write("Time, Humidity \n")

print ("Time, Humidity")

for n in range(20):
#repetedly gets data from the humidty sensor until adding 20 lines to the file table
    humidity = sense.get_humidity()
    humidity = round(humidity, 2)

#to write in the file the data collected
    file.write(time.strftime('%X'))
    file.write(",")
    file.write(str(humidity))
    file.write("\n")

    print (time.strftime('%X'),humidity)

    time.sleep(1)

file.close()
```

3. Sobald eure Daten in einem nützlichen Format vorliegen, könnt ihr sie auf vielfältige Weise untersuchen. Die vom Luftfeuchtesensor erfassten Daten, wurden im Dateimanager gespeichert (auf den ihr oben auf eurem Desktop Zugriff habt). Öffnet die Datei und zeichnet die Daten zu der Zeit und der Luftfeuchte in nachstehendes Diagramm ein.

4. Führt euren Code erneut aus, pustet dieses Mal aber langsam auf den Sensor. Öffnet die Datendatei erneut. Die neu erfassten Daten wurden hinzugefügt. Zeichnet die neuen Daten zu Zeit und Luftfeuchte als Punkte in obenstehendes Diagramm ein. Vergleicht beide Kurven miteinander. Was schließt ihr daraus?

Aufgabe 2.3 – Daten anzeigen

Auf der ISS müssen die Astronauten prüfen, ob alle Geräte und Instrumente ordnungsgemäß funktionieren und Abweichungen der Bodenstation melden. Nun sollt ihr Schritt 5 aus dem Flussdiagramm in Abbildung A4 durchführen, also einen optischen Alarm erstellen, der die Astronauten benachrichtigt, wenn die Luftfeuchtigkeit über 65 % steigt.

Übung

1. Schaut euch folgenden Code an. Was wird eurer Meinung nach passieren? Schreibt eure Antwort nachstehend in die Kommentare.

```
File  Edit  Format  Run  Options  Windows  Help
from sense_hat import SenseHat
import time

sense = SenseHat()

while True:
    humidity = sense.get_humidity()
    humidity = round(humidity, 2)

#_____________________________________
    if humidity<65:
        bg = [0,100, 0]

#_____________________________________
    else:
        bg = [100,0, 0]

    sense.show_message(humidity,scroll_speed=0.05,back_colour=bg)
```

2. Kopiert nun euren Code in das neue Python-Fenster. Wählt File > Save As [Datei > Speichern als] und gebt eurem Programm einen Dateinamen. Klickt dann auf Run > Run Module [Ausführen > Modul ausführen]. Was passiert, wenn ihr den Code ausführt?

Hinweis: Zum Löschen der Matrix gebt den Befehl sense.clear() in Python Shell ein.

3. Der Sense HAT kann nur Daten anzeigen, die er als Zeichenfolgen erkennt. Diese Zeichen sind alle, die ihr auf der Tastatur habt. Zeichenfolgen sind Ketten von Zeichen, also Buchstaben, Zahlen oder Satzzeichen. In einer Zeichenfolge werden alle Zeichen als Text gelesen, auch wenn es sich um Zahlen handelt. In diesem Beispiel war die Luftfeuchtevariable aber eine Gleitkommazahl (also eine Dezimalzahl). Um den Luftfeuchtewert auf dem Sense HAT darzustellen, müsst ihr ihn in eine Zeichenfolge umwandeln. Fügt dazu vor der letzten Zeile des obenstehenden Codes folgende Zeile ein:

   humidity = str(humidity)

Führt euren Code erneut aus. Ist das passiert, was ihr in Übung 1 erwartet habt?

4. Pustet langsam auf die Sensoren, bis ihr eine Luftfeuchte von mehr als 65 % erreicht. Was passiert mit den Daten, die auf der Matrix angezeigt werden?

5. Eine bessere Möglichkeit, die Luftfeuchtigkeit grafisch darzustellen, ist ein einfaches Balkendiagramm. Probiert den nachstehenden Code in einer neuen Datei aus.

```
File  Edit  Format  Run  Options  Windows  Help
from sense_hat import SenseHat

sense = SenseHat()

white = (255,255,255)

for height in range (4):
    sense.set_pixel(3,height, white)
    sense.set_pixel(4,height,white)
```

a) Was passiert, wenn ihr die Zahl in „range(4)“ ändert?

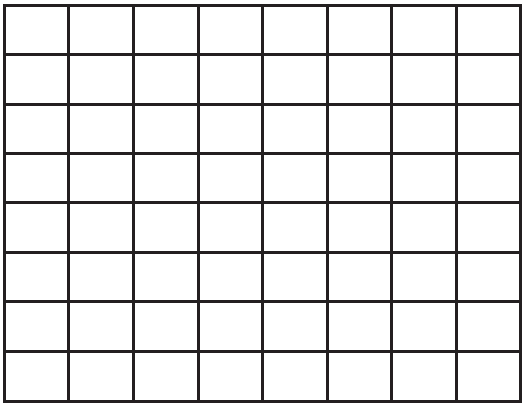
b) Was ist die größte Zahl, die ihr in „range( )“ eingeben könnt, bevor es zu einer Fehlermeldung kommt? Erläutert, warum das passiert.

6. Die Luftfeuchte kann zwischen 0 und 100 liegen. Wenn der Wert auf einen Bereich zwischen 0 und 8 gesenkt werden könnte, so könnte er in „range( )“ eingegeben werden, um die Luftfeuchtigkeit grafisch darzustellen. Dazu sollte es genügen, die Luftfeuchtigkeit durch 12,5 zu teilen.

a) Könnt ihr erklären, warum gerade 12,5?

b) Zeichnet in nachstehendem Raster ein, was eurer Meinung nach auf der LED-Matrix angezeigt wird, wenn ihr den unten angegebenen Code eingebt (nehmt dabei an, dass die Luftfeuchtigkeit dieselbe wie in Übung 3 ist). Begründet eure Abbildung.

```
File  Edit  Format  Run  Options  Windows  Help
from sense_hat import SenseHat

sense = SenseHat()

white = (255,255,255)

humidity = sense.get_humidity()
# int is used to convert data to an integer number
humidity = int(humidity/12.5)

for height in range (humidity):
    sense.set_pixel(3,height, white)
    sense.set_pixel(4,height,white)
```



c) Um eure Antwort zu überprüfen, gebt euren Code in die neue Python-Datei ein. Wählt File > Save As [Datei > Speichern als] und gebt eurem Programm einen Dateinamen. Klickt dann auf Run > Run module [Ausführen > Modul ausführen].

d) Fügt eurem Code eine „whileTrue“-Schleife hinzu und lasst Astro Pi kontinuierlich die Luftfeuchte messen und sie als Balkendiagramm darstellen. Schreibt euren Code in nachstehendes Feld.

Weiterführende Aufgabe

Versucht, euren Astro Pi in eine Mini-Wetterstation zu verwandeln, die die Temperatur, die Luftfeuchte und den Luftdruck in eurem Klassenzimmer misst. Ermittelt dazu zunächst, welche Bedingungen ideal für ein Klassenzimmer sind. Programmiert dann ein Alarmsystem, das euch benachrichtigt, wenn diese idealen Bedingungen überschritten werden.

→ AUFGABE 3 – WO IST UNTEN?

Alles auf der Erde unterliegt der Schwerkraft bzw. Gravitation. Das ist die Kraft, die euch wieder zurückzieht, wenn ihr in die Luft springt. Deshalb ist es auf der Erde sehr leicht, herauszufinden, wo oben und unten ist. „Unten“ ist die Richtung, in der die Schwerkraft euch zieht, und „oben“ ist die entgegengesetzte Richtung.

Auf der Internationalen Raumstation gibt es weder ein Oben noch ein Unten. Es besteht kein Unterschied zwischen dem Boden und der Decke. Bis man sich an diese Desorientierung gewöhnt, muss man als Raumfahrer mit dem ein oder anderen Unwohlsein rechnen („Raumkrankheit“).

Bei dieser Aufgabe lernt ihr, wie man den Beschleunigungssensor (engl. accelerometer) verwendet, um sich räumlich zu orientieren.

Übung

1. Die Internationale Raumstation umkreist die Erde in einer Höhe von 400 km. Da die Gravitation mit zunehmender Entfernung abnimmt, beträgt sie in dieser Höhe nur 90 % der Schwerkraft auf der Erde. Schaut man sich jedoch an, wie die Astronauten durch die Internationale Raumstation schweben, so scheinen sie kein Gefühl für oben oder unten zu haben. Könnt ihr erklären, warum das so ist?

↑ Auf der ISS gibt es kein oben und unten

2. Der Sense HAT ist mit einem Bewegungssensor ausgestattet, dem sogenannten IMU (Inertial Measurement Unit; Trägheitsmesseinheit). Diese Einheit vereint eigentlich drei Sensoren in einem:

- Ein Gyroskop (Kreiselgerät), welches die Rotation und den Impuls (Schwung) misst;
- Ein Beschleunigungssensor, der verwendet werden kann, um die Richtung der Gravitation bei einem ruhenden Objekt zu ermitteln;
- Ein Magnetometer (misst das Magnetfeld der Erde, ähnlich wie ein Kompass)

Beschleunigungssensoren messen in Meter pro Quadratsekunde (m/s2) oder in g-Kraft (g). „g“ ist die Einheit für die durchschnittliche Beschleunigung (9,8 m/s2) durch die Gravitation, gemessen an der Erdoberfläche (auf der Höhe des Meeresspiegels). 1 g auf der Erde entspricht 9,8 m/s2.

Abbildung A6

↑ IMU-Sensor im Sense HAT

Öffnet ein neues Python-Fenster und gebt nachstehenden Code ein.

```
File  Edit  Format  Run  Options  Windows  Help
from sense_hat import SenseHat

sense = SenseHat()

acceleration = sense.get_accelerometer_raw()

print(acceleration)
```

3. Führt euren Code aus und notiert die Messungen des Beschleunigungssensors.

---

4. Der Sense-HAT-Beschleunigungssensor erfasst Daten auf 3 Achsen (also in 3 Dimensionen). Die Daten lassen sich in dem Format, in dem ihr sie gerade ausgegeben habt, nur schwer lesen. Um sie in einem besser lesbaren Format auszugeben, übernehmt folgenden Code und führt ihn aus.

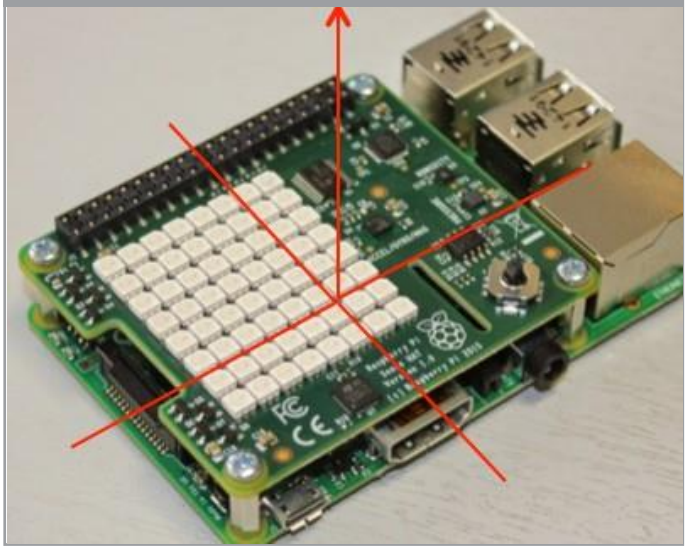
```
File  Edit  Format  Run  Options  Windows  Help
from sense_hat import SenseHat

sense = SenseHat()


acceleration = sense.get_accelerometer_raw()
x = acceleration['x']
y = acceleration['y']
z = acceleration['z']

x=round(x, 0)
y=round(y, 0)
z=round(z, 0)

print("x {0} y {1} z {2}".format(x,y,z))
```

5. Notiert die Ergebnisse. In welche Richtung zeigt die Gravitationskraft eurer Meinung nach? Diskutiert eure Antworten mit euren Mitschülerinnen, Mitschülern und der Lehrkraft.

6. Dreht euren Astro Pi um 90 Grad. Notiert die Ergebnisse und erläutert die Unterschiede zu den vorherigen Werten.

7. Vervollständigt nachstehendes Schema mit den Richtungen der X-, der Y- und der Z-Achse eures Astro Pi. Wenn nötig, könnt ihr den Astro Pi noch einmal drehen.

Abbildung A7



8. Tabelle A2 zeigt die Messwerte des Beschleunigungssensors für die X-, Y- und Z-Achse des Astro Pi auf der ISS an einem einzigen Arbeitstag. Warum misst der Beschleunigungssensor eurer Meinung nach, einen Wert nahe null? Könnte es sich um eine Anomalie handeln? Vergleicht zur Beantwortung der Frage die auf der ISS erfassten Daten mit den Daten, die ihr gemessen habt. Schaut euch auch eure Antwort aus Frage 1 dieser Aufgabe an. Diskutiert eure Antworten mit euren Mitschülerinnen, Mitschülern und der Lehrkraft.

Tabelle A2: Messwerte des Beschleunigungssensors vom Astro-Pi-Computer auf der ISS

accel_x	accel_y	accel_z	Zeitstempel
-0,00057	0,019359	0,014357	10:45:00 Uhr
-0,00044	0,019405	0,014425	11:45:00 Uhr
-0,00056	0,019531	0,014597	12:45:00 Uhr
-0,00056	0,019506	0,014432	13:45:00 Uhr
-0,00058	0,019464	0,014569	14:45:01 Uhr
-0,00056	0,01939	0,014578	15:45:00 Uhr
-0,00053	0,019384	0,014389	16:45:00 Uhr
-0,00046	0,01926	0,01444	17:45:00 Uhr
-0,00053	0,019266	0,014568	18:45:01 Uhr

Weiterführende Aufgabe

1. Die ISS verliert pro Tag etwa 50 bis 100 Meter an Höhe. Ohne Korrektur würde sie aus der Umlaufbahn geraten und immer weiter sinken (Orbit-Verfall). Dann wäre die Raumstation in großer Gefahr. Der Grund für dieses Absinken ist, dass in 400 km Höhe noch immer eine geringe Atmosphäre vorhanden ist. Die Luft bildet also einen Widerstand, der an der ISS zieht. Dadurch kommt es zu einem langsamen Orbit-Verfall. Um dem entgegenzuwirken, muss die ISS mittels Triebwerkzündungen wieder angehoben und zurück in ihre Umlaufbahn gebracht werden (Bahnanhebung oder Bahnkorrektur). Solche Bahnkorrekturen finden etwa drei- bis viermal im Monat statt. Mit welchen Messungen kann ein Astro Pi diese Ereignisse erfassen?

2. Ladet euch die im Columbus-Modul zwischen dem 16.02.2016 und 29.02.2016 erfassten Daten herunter. Könnt ihr erkennen, ob die ISS in diesem Zeitraum Bahnkorrekturen vorgenommen hat und wenn ja, wie lange sie angehalten haben? Vergleicht oder bringt auch die aktuelle Höhenkurve mit den Daten in der Datei in Zusammenhang. Notiert eure Schlussfolgerungen.