Internet of Things (IoT)

Topic 2: Raspberry Pi and SenseHat

Raspberry Pi is a tiny credit card-sized computer. Because of its small size and low cost, it has been extensively used for teaching kids to code, as well as studying and developing Internet of Things (IoT) applications, thanks to its capability to interface with different sensors and actuators easily.

In this workshop, we will be using Raspberry Pi 4 Model B, which is the latest model to date with onboard Wi-Fi and Bluetooth connectivity. In this introductory exercise, you will go through the basics about Raspberry Pi and get familiar with the procedures of configuring the Pi.

Getting Started

Get a set of equipment, which includes the following:

- Raspberry Pi with micro SD card
- SenseHat
- Power charger
- Video cable
- Portable display with keyboard

Before proceeding with setting up your Pi, you should first understand the structure of the device. Spend a minute to examine the Pi and identify the following. DO NOT PLUG IN ANYTHING YET.

1. How to tell the model of your Pi?
2. Where do you put the microSD card?
3. Where do you plug the Ethernet cable?
4. Where are the HDMI, power, camera, and A/V slots?
5. How many USB ports are there?

One of the most common operating system (OS) options for Raspberry Pi is the “Raspbian”, which is the official Raspberry Pi-optimized variant of the Debian distribution of Linux operating system. To boot your Pi, you should install an OS on your microSD card. In this workshop, we will provide microSD cards with Raspbian pre-installed.

1. Remove the case for the Raspberry Pi

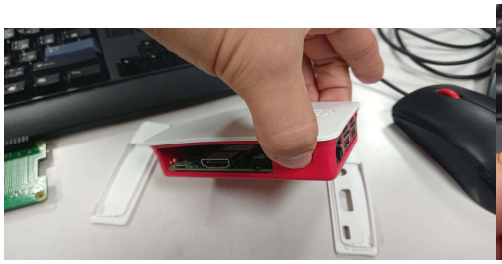
a.

b.

c.

d.

e.



f.

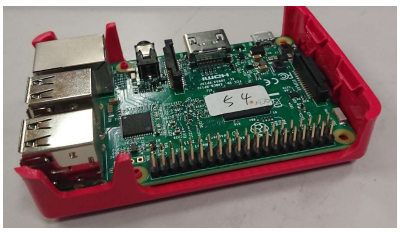
****

In case you accidentally removed the whole Raspberry Pi from the case, please gently put it back by **inserting the side with the SD card inside the appropriate slot first**:

a.

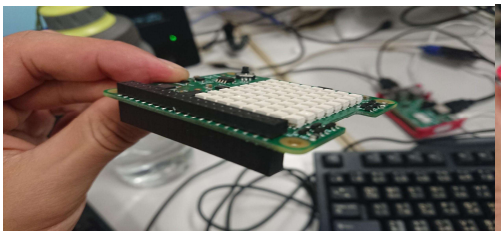
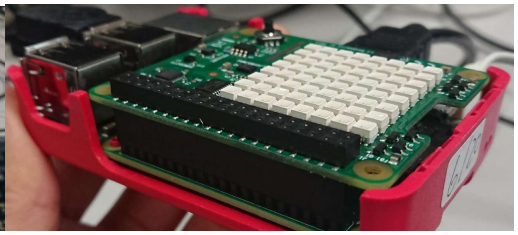
b.

c.



2. Assembly the SenseHat to the Raspberry Pi using the pins
   (make sure the Raspberry Pi is powered off!)

   a. b.

3. Plug in one end of the HDMI cable to the portable display. The other end of the HDMI cable should be connected to your Pi's HDMI port.
4. Connect a USB keyboard and a USB mouse to any of the USB ports of your Pi.
5. Confirm the connection of the micro SD card, VGA cable, keyboard and mouse to your Pi. Connect the Micro-USB cable to the corresponding port next to the HDMI port, and then power up the USB power supply.
6. Your Pi should be booting now with a "rainbow" splash screen, followed by text boot messages. A desktop environment will show up in a few seconds. If you cannot boot your Pi, switch off the power, verify the connections and restart it.
7. (Important! You will run into trouble if you skip this step!) After successful boot up, configure the keyboard layout to US (https://youtu.be/gbJB3387xUw?t=7m50s) and change the time zone to local time zone (Asia/Hong Kong), in order to avoid encountering keyboard input issues
8. Raspberry Pi 3 Model B is equipped with a wired Ethernet port and an 802.11n Wi-Fi adapter as the onboard network interfaces. Connect the Pi using Wi-Fi or Ethernet to the network. (You can select whatever network to join for this exercise.)

****

Troubleshooting

If the monitor does not show the desktop screen, please ensure that the SD card is intact.

If the SD card is bent or broken, please let the TA or technician know.

Once you hook up to the Internet, get familiar with the user interface. **Finish the UNIX exercise posted on Moodle. Make sure you know how to write and run python programs on the pi.**

Raspberry Pi vs PC (Moodle submission required)

It is well-known that IoT devices are less powerful than desktop computers. You will quantitatively measure the performance difference through running a **merge sort** program in the two different platforms, the Pi and the desktop computer in the laboratory. Merge sort is one of the most efficient sorting algorithms currently exist. To get the most accurate results, you should not execute other programs while collecting the data.

Instructions on running the program on Raspberry Pi

1. On a *Raspberry Pi*, download merge_sort.py. The file should be in the **/home/pi/Downloads** directory.
2. Open the terminal and type **`cd /home/pi/Downloads`** to navigate to it.
3. To run the program, type
   `python3 merge_sort.py <size of array to be sorted>.`
   For example, **`python3 merge_sort.py 1000`** will measure the time taken in seconds to sort an array composed of 1,000 random integers.
   When the size of the array is large, it takes a while to generate the results. Please be patient.

Instructions on running the program on the desktop PC

1. On a *PC*, download merge_sort.py. The file should be in the **Downloads** directory.
2. Open the command prompt (cmd.exe) and type **`cd Downloads`** to navigate to it.
3. To run the program, type
   py merge_sort.py <size of array to be sorted> *or*
   python merge_sort.py <size of array to be sorted>
   Either one should work.

Time performance comparison of the two platforms

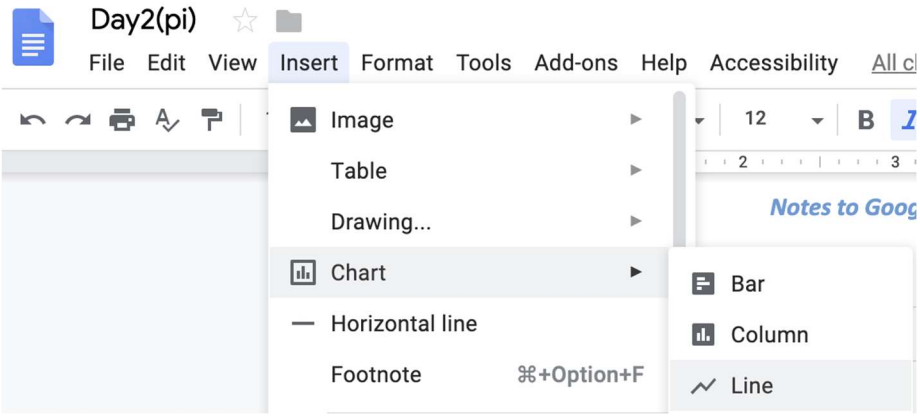
1. Run the program starting from array size 500,000 on the **Pi** with an increment of 100,000, until 800,000. Record the time needed. Plot a **line** of the time **(round to the nearest second)** needed against the size of array to the graph on the next page. It is fine if the program fail due to memory limitation when you increase the array size to a certain number. If that's the case, just stop there.

2. Run the same program in a **desktop computer** using size 500,000 to 800,000. On the same graph as step 1, plot a **line** showing the time needed against the size of array.

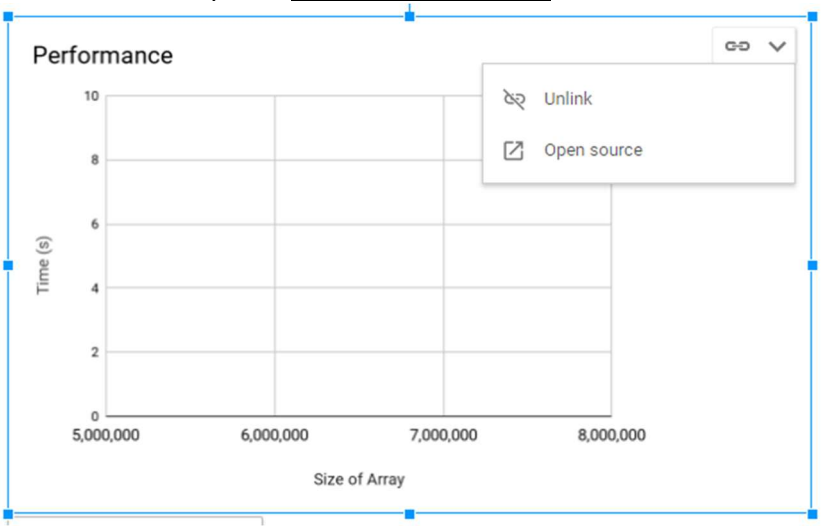
Task 1: *Save the graph you plotted as* ***chart.png***
Submit to Moodle: ***chart.png***

Notes to Google Drive Users

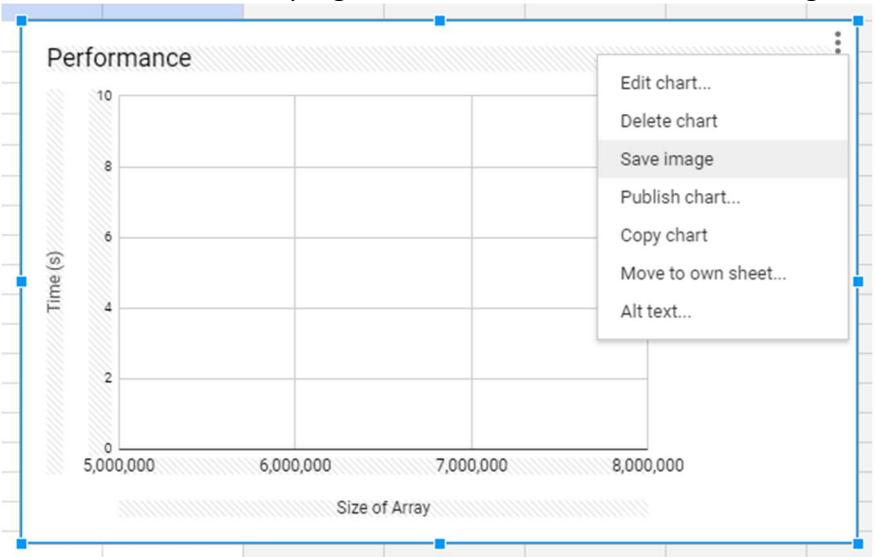
To create a graph in Google Doc:



To edit the graph, click on the graph, then click on the icon on the top right hand corner, then click on "Open Source", it will open a Google Spreadsheet.

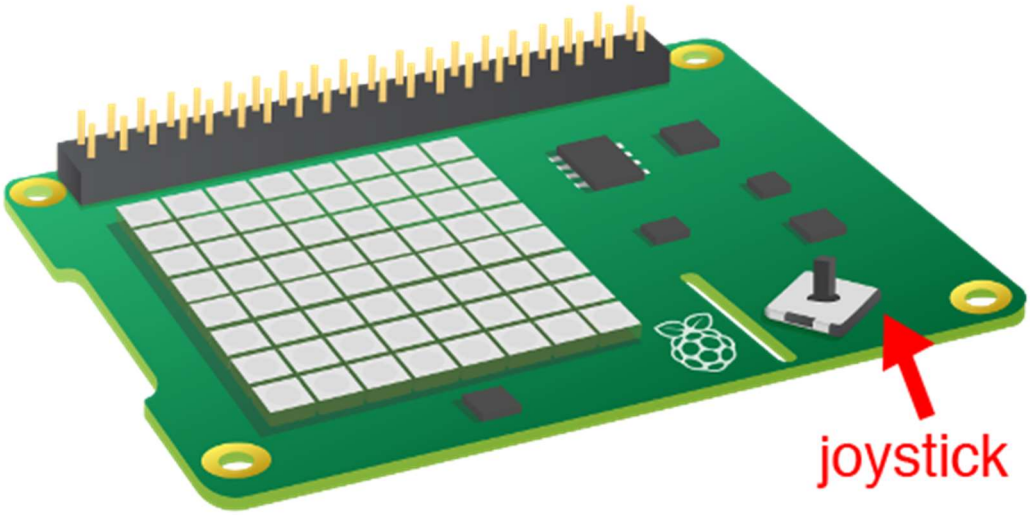


To download the image, go to the Google spreadsheet that stores the chart, click on the chart, and click the 3 dots on the top right hand corner, and click "Save image".



SenseHat

The **Sense HAT** is an add-on board for the Raspberry Pi. The board has a **joystick** for user inputs. It allows you to make measurements of **temperature**, **humidity**, **pressure**, and **orientation**, and to **output** information using its built-in LED matrix!



First of all, let's install the Sense HAT package using the following command:

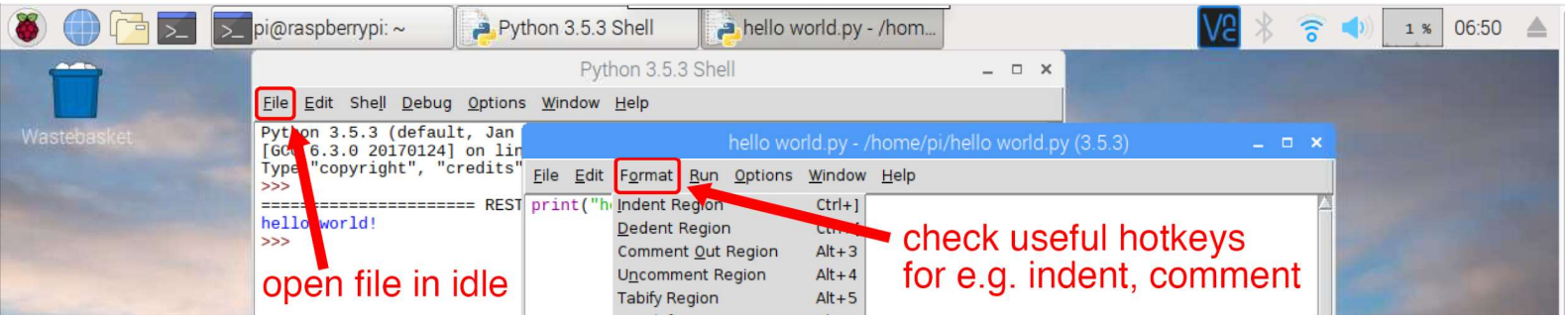
```
sudo apt-get update
sudo apt-get upgrade                # this command may take some time!

sudo apt-get install sense-hat libjpeg8-dev
sudo pip3 install pillow
```

Finally, reboot the Raspberry Pi to complete the installation:

```
sudo reboot
```

Displaying Text (Moodle submission and demonstration required)

1. Open Python3 program editor (IDLE) from the **Terminal** by typing:
   ```
   sudo idle &
   ```
   If the command cannot be found, you can install IDLE (for Python 3) by typing:
   ```
   sudo apt-get install idle3
   ```
2. Download and open file ***display.py*** using **idle**. Steps 3-6 explain the program.



Useful hotkeys (try them by yourself!):

[Ctrl +]] - indent selected lines
[Ctrl + [ ] - dedent selected lines
[Alt + 3] - comment out selected lines
[Alt + 4] - uncomment selected lines

[Ctrl + C] - copy the highlighted text,
[Ctrl + F] - open search box (when this window is opened, you cannot edit code),
[Ctrl + V] - paste in the text and search for it across the python file,
[Ctrl + R] - find & replace (when this window is opened, you cannot edit code, except by replacing)

3. Import the modules and libraries needed for Sense HAT
   ```
   from sense_hat import SenseHat
   ```

4. Instantiate a SenseHat object to use its functions
   ```
   sense = SenseHat()
   ```

5. Create variables for the blue and yellow colors in **(R, G, B)** format
   ```
   blue = (0, 0, 255)
   yellow = (255, 255, 0)
   ```

6. Use a while loop to keep on displaying a text message with the **show_message()** function
   ```
   while True:
       sense.show_message("EEE is awesome!", text_colour=yellow,
       back_colour=blue, scroll_speed=0.05)
   ```

7. Now try to run the program via **Run Module (F5)** and see the results! [IF YOU RUN THE PYTHON PROGRAMS USING COMMAND LINES, MAKE SURE YOU ARE USING THE RIGHT VERSION OF PYTHON. You are recommended to use "python3 display.py" instead of "python display.py" to execute the program using command line.]

display.py

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

blue = (0, 0, 255)
yellow = (255, 255, 0)

while True:
   sense.show_message("EEE is awesome!", text_colour=yellow, back_colour=blue,
scroll_speed=0.05)
```

Exercises –

Task 2: *Show the message and background using the rainbow (VIBGYOR) colors. The sequence is as follows:*

text color: Red	*background color: White*
text color: Orange	*background color: Red*
text color: Yellow	*background color: Orange*
text color: Green	*background color: Yellow*
text color: Blue	*background color: Green*
text color: Indigo	*background color: Blue*
text color: Violet	*background color: Indigo*

Hint: find the RBG codes of the colors on the web.

Submit to Moodle: ***a video demonstrating task2_display.py***

Task 3: *Extend Task 1 so that the message will be shown continuously using a random combinations of VIBGYOR and the white colors.*

Submit to Moodle: ***a video demonstrating task3_display.py***

Task 4: *Simulate the process that a ball is moving on the display as shown in the ball.mp4 video in Moodle. In each second, the ball moves to a new position with a different color. The new position can be any of the surrounding 8 pixels. You may want to sleep for 1 second before changing the position of the ball.*

The ball.py in Moodle displays the white ball in a random position in each second.

Submit to Moodle: ***a video demonstrating task4_ball.py***

Task 5: *Download game.py. Understand the program and run the program. You can use the joystick to control the bat and bounce the ball. Extend the game by introducing different levels of difficulties (ball moves faster, bat is shorter; you can decide the levels by yourselves). The color of the bat should follow the rainbow colors as the level is higher. Lowest level is red.*

Submit to Moodle: ***a video demonstrating task5_ball.py***

Sensing the Environment (Moodle submission and demonstration video required)

The Sense HAT has a set of environmental sensors for detecting the surrounding conditions: it can measure **pressure**, **temperature**, and **humidity**.

1. Download file ***sensor.py***. Steps 2-7 explain the codes.

2. Import the modules and libraries needed for Sense HAT
   ```
   from sense_hat import SenseHat
   ```

3. Instantiate a SenseHat object to use its functions, and initiate the sensors
   ```
   sense = SenseHat()
   sense.clear()
   ```

4. To get and print the pressure reading, use the **get_pressure()** function
   ```
   pressure = sense.get_pressure()
   print(pressure)
   ```

5. To get and print the temperature reading, use the **get_t*emperature*()** function
   ```
   temp = sense.get_temperature()
   print(temp)
   ```

6. To get and print the humidity reading, use the **get_humidity()** function
   ```
   humidity = sense.get_humidity()
   print(humidity)
   ```

7. Now try to **Run** the program and see the results!

sensor.py

```
from sense_hat import SenseHat

sense = SenseHat()
sense.clear()

pressure = sense.get_pressure()
print(pressure)

temp = sense.get_temperature()
print(temp)

humidity = sense.get_humidity()
print(humidity)
```

Exercises –

Task 6: *Modify the program so that when the joystick on the senseHat is pressed, the current temperature is displayed on the console (not on senseHat) with two decimal places (**Hint:** Use the round method for the decimal places requirement).*

Submit to Moodle: ***a video demonstrating task7_sensor.py***

Task 7: *Measure the temperature every 1 second and measure 50 readings. Log the temperature information, together with the timestamp that the temperature is measured, in a text file named **record.txt** (sample file posted on Moodle). (**Hint:** explore the **time** and **datetime** libraries in python.)*

Submit to Moodle: ***record.txt*** *&* ***task8_record.py***

Detecting movement (Moodle submission required)

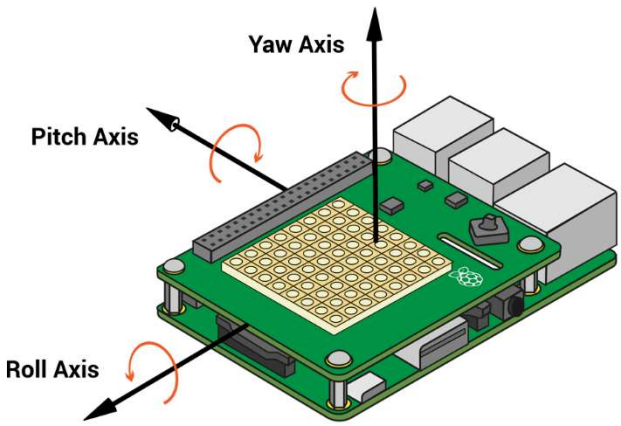
The Sense HAT has an **IMU** (Inertial Measurement Unit) chip which includes a set of sensors that detect movement:

- A gyroscope (for detecting which way up the board is)
- An accelerometer (for detecting movement)
- A magnetometer (for detecting magnetic fields)

You may wonder why is a movement sensor important? For example, when you are in space, there is one question of absolute importance to which you must always know the answer: "Which way am I pointing?" If you don't know your orientation, you are in big trouble!!! In case you don't know, IMU sensors are actually used on all manned and unmanned spacecraft to track movements and maintain an understanding of orientation!

You will also need to learn about **pitch**, **roll**, and **yaw**. All objects have three axes around which they can rotate.

- Pitch — imagine a plane taking off
- Roll — imagine a plane doing a victory roll
- Yaw — imagine steering a plane like a car



If you know how much rotation has happened on each axis of an object, then you will know which way the object is pointing.

Let's try to get the orientation values from the Sense HAT using the **get_orientation()** function:

```
pitch, roll, yaw = sense.get_orientation().values()
```

orientation.py

```
from sense_hat import SenseHat

sense = SenseHat()
sense.clear()

while True:
   pitch, roll, yaw = sense.get_orientation().values()
   print("pitch = %s, roll = %s, yaw = %s" % (pitch, roll, yaw))
```

Another way to detect orientation is to use the **get_accelerometer_raw()** function, which tells you the amount of g-force acting on each axis. If any axis has ±1G, then you know that axis is pointing downwards.

In the following example (***orientation2.py***), the amount of gravitational acceleration for each axis is extracted and is then rounded to the nearest whole number:

orientation2.py

```
from sense_hat import SenseHat

sense = SenseHat()

while True:
        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))
```

Run the program and try to rotate the Sense HAT. You should see the values for x and y change between -1 and 1. If you place the Pi flat or turn it upside down, the value for the z axis will be 1 and then -1.

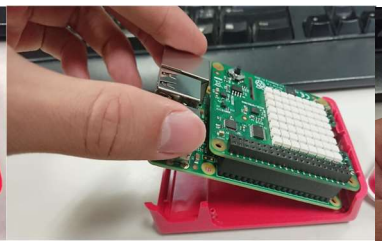
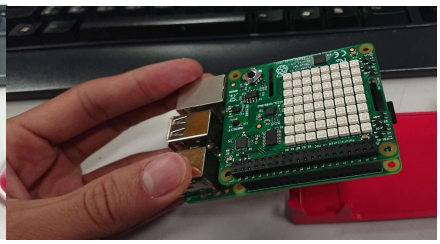
Imagine a raspberry pi is installed in a car. Suggest how to use a pi to determine how many left turns and right turns the car has made. You do not have to develop the program but have to explain the program logic. Your description should be no more than half a page. ***Task 8: Submit your file (car.docx or car.txt) to Moodle.***

Congratulations! That's the end of this session! Have fun programming! ☺

To remove the SenseHat:

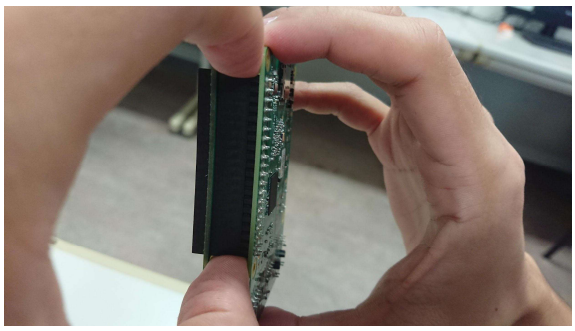
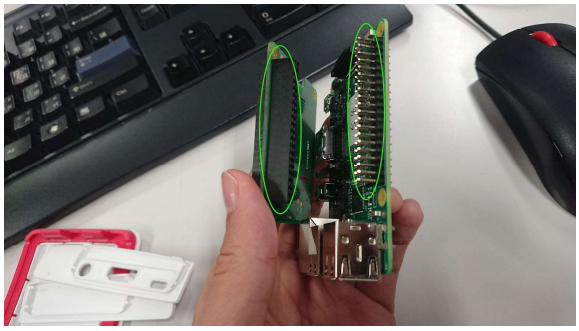
1. Turn off and **cut the power** of the Raspberry Pi
2. Take the Raspberry Pi out of the case

a. b. c.

3. You are suggested to remove the SenseHat using this method to avoid damaging the pins of the Raspberry Pi:

a. b.

References

1. https://projects.raspberrypi.org/en/projects/getting-started-with-the-sense-hat