In this class, we will start preparation to the project. Our project will be to develop several games for an arcade game using Raspberry Pi and the Sense Hat. The Raspberry Pi is essentially a small computer, it was developed in the United Kingdom in 2012. The Sense Hat is an add-on board for the Raspberry Pi. It has several sensors (temperature, pressure, humidity, gyroscope, magnetometer, accelerometer), and also an 8 x 8 pixel screen and a joystick.

Actually, there is a Raspberry Pi and a Sense Hat on board of the International Space Station!

We will need to learn the basics of operating with the Raspberry Pi and the Sense Hat. We have two Raspberry Pi's in class, and you can also test your code using an emulator: https://trinket.io/sense-hat.

1. **Show a message on the SenseHat**

   To show a message, you can use the following script.
   ```python
   from sense_hat import SenseHat
   sense = SenseHat()
   sense.show_message("Hello world")
   ```

   We can change how the message is displayed by adding some extra parameters to the `show_message` command.
   - `scroll_speed`: affects how quickly the text moves across the screen. The default value is 0.1. The bigger the number, the lower the speed.
   - `text_colour`: alters the colour of the text and is defined via three values to specify red, green, and blue. These are also called RGB values.
   - `back_colour`: alters the colour of the background.

   We will come back to this exercise later, and now let us learn how to define colours.

2. **Representing colours with numbers**

   ```python
   from sense_hat import SenseHat
   sense = SenseHat()
   white = (255, 255, 255)
   sense.clear(white)
   ```

   If you run this code, the LED matrix will go bright white. The variables r, g, and b represent the colours red, green, and blue. Their values specify how bright each colour should be; each value can be between 0 and 255. In the above code, the maximum value for each colour has been used, so the result is white.

   You can find the r, g, b values for different colours here: https://www.w3schools.com/colors/colors_rgb.asp

   **Exercise 1.** Show a message in red with blue background and speed = 0.5. Put your scrolling message in a while loop to make it repeat forever (remember that you can use 'while True' for this purpose).

   **Exercise 2.** You can show a single letter by replacing `sense.show_message("blablabla")` in the code above with `sense.show_letter('A')`. The same parameters can be used to change
the colour and the background. Your task is to spell your name with a two seconds pause between the letters. To pause the execution of your script, you can call the sleep function (e.g. if you want a 1 second pause, call sleep(1)) from a module called time.

3. Setting pixels

The LED matrix can display more than just text! We can control each LED individually to create an image. Every pixel has coordinates \((x, y)\) where \(x\) and \(y\) are between 0 and 7. The pixel \((0, 0)\) is the top left corner, the pixel \((7, 7)\) is the bottom right corner. You can set pixels (LEDs) individually using the `set_pixel()` method.

```python
from sense_hat import SenseHat
blue = (0, 0, 255)
sense.set_pixel(0, 2, blue)
```

**Exercise 3.** Light up pixels in the four corners of the Sense Hat in colours of your choice.

Setting pixels individually can work brilliantly, but it gets rather complex when you want to set multiple pixels. To change all the pixels in one go with the `set_pixels` command.

```python
from sense_hat import SenseHat
sense = SenseHat()
# Define some colours
g = (0, 255, 0) # Green
b = (0, 0, 0) # Black
# Set up where each colour will display
smiley = [g, g, g, g, g, g, g, g,
g, b, b, b, b, b, b, g,
g, b, b, b, b, b, g, g,
g, g, g, g, b, g, g, g,
g, b, g, b, b, g, b, g,
g, b, g, g, g, g, b, g,
g, b, b, b, b, b, b, g,
g, g, g, g, g, g, g, g]
# Display these colours on the LED matrix
sense.set_pixels(smile)
```

4. Detecting movement

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)

We all know the Earth rotates around an axis that runs between the North and South Poles. All objects have three axes around which they can rotate. These are:

- Pitch
Roll

The image shows where the axes are in relation to the Sense HAT.

You can access the data as follows.

```python
from sense_hat import SenseHat
sense = SenseHat()
o = sense.get_orientation()
pitch = o["pitch"]
roll = o["roll"]
yaw = o["yaw"]
print("pitch {0} roll {1} yaw {2}".format(pitch, roll, yaw))
```

The `sense.get_accelerometer_raw()` method tells you the amount of G-force acting on each axis (x, y, z). If any axis has ±1G, then you know that axis is pointing downwards. In this example, the amount of gravitational acceleration for each axis is extracted and is then rounded to the nearest whole number:

```python
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))
```

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.

**Exercise 4.** Display a letter of your choice on the Sense Hat so that it always looks upwards. You can rotate the screen by 0, 90, 180, or 270 degrees using the `set_rotation` method. For example, to rotate your screen by 180 degrees, you'd use the following code:

```python
sense.set_rotation(180)
```

Here is some pseudocode to get you started:

```
If the x axis has -1 G, rotate 180 degrees
Else if the y axis has 1 G, rotate 90 degrees
Else if the y axis has -1 G, rotate 270 degrees
Else rotate 0 degrees
```

**5. Using the joystick**

You can detect when the Sense HAT’s joystick is pressed, held, and released in five different directions: up, down, left, right, and middle.
Exercise 4. Depending on which way the joystick was pressed, display one of the letters U, D, L, R or M on the LED matrix.

The Sense HAT joystick can be used to trigger function calls in response to being moved. For instance, you can tell your program to continually 'listen' for a specific event, such as the joystick being pushed up (direction_up), and to then trigger a function (called pushed_up in this example) in response. The function triggered by the event can either have no parameters, or it can take the event as a parameter. In the example below, the event is simply printed out.

```
from sense_hat import SenseHat
sense = SenseHat()
def pushed_up(event):
    print(event)
    # Tell the program which function to associate with the upward direction
    sense.stick.direction_up = pushed_up
while True:
    pass  # This keeps the program running to receive joystick events
```

You can use similar methods for other directions. Another useful example is the direction_any method:

```
sense.stick.direction_any = do_thing
```

If you use this method as seen in the example, the do_thing function will be triggered in response to any joystick event.

Exercise 5. Create functions to fill the LED matrix with four different colours. Add triggers to call one function for each possible direction in which the joystick can be pressed.