MAE 6291
Internet of Things for Engineers

Prof. Kartik Bulusu, MAE Dept.

Week 2 [01/29/2025]	• Setting up the Edge Lab • What is an edge device • Differences in Cloud and Fog computing • Guest lecture	• Some more programming constructs • RPi skeleton code • Raspberry Pi programming [Blinking LEDs using Thonny]

git clone https://github.com/gwu-mae6291-iot/spring2025_codes.git

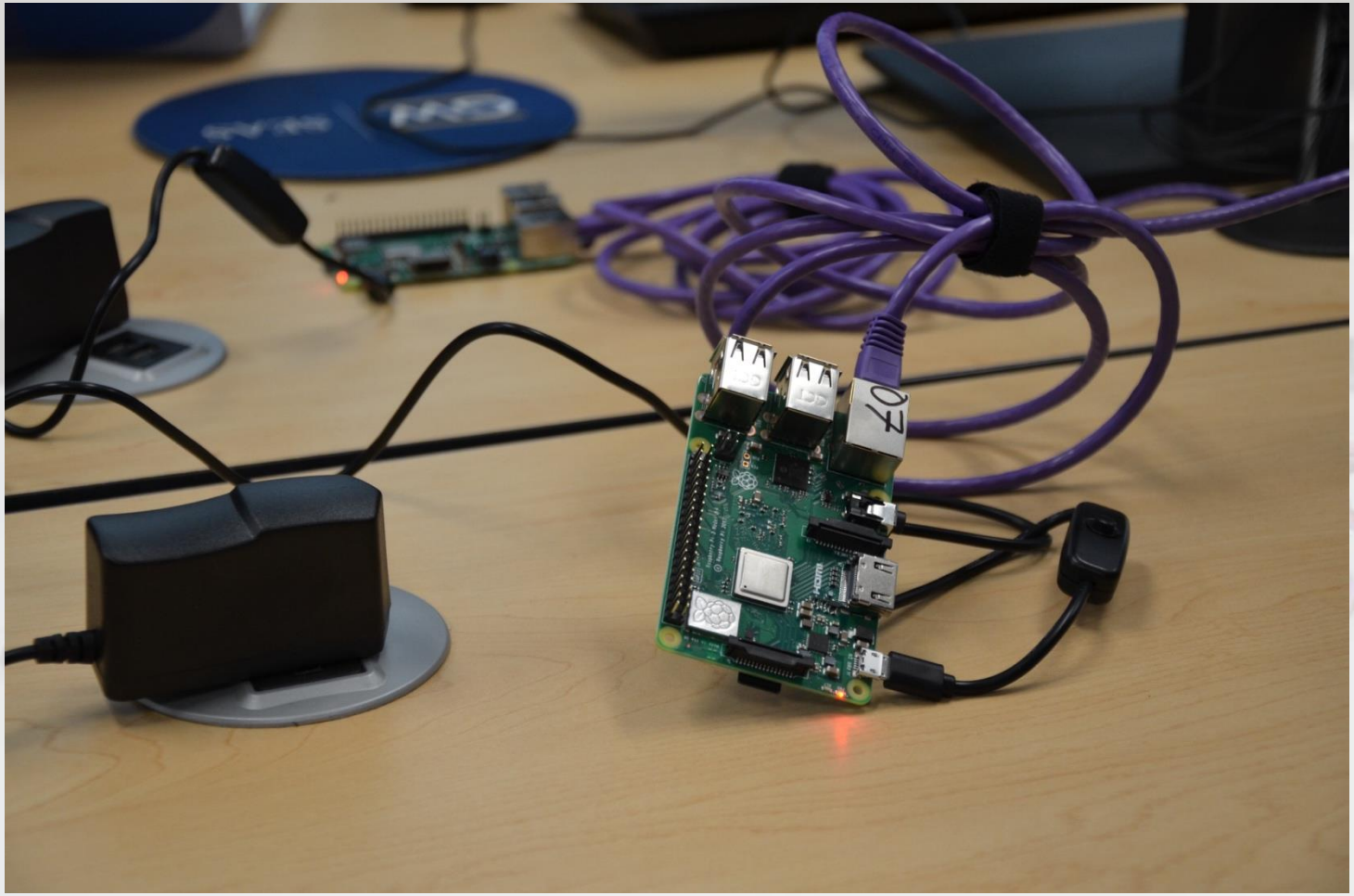
GW School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

Spring 2025

Photo: Kartik Bulusu

Set up lab the Edge-lab

STEP [1]:
Connect the RPis to each desk power outlet as shown



- Make sure there is a microSD card installed in the RPi
- Connect the RPi using the microUSB cable provided
- Connect the purple colored ethernet cables specifically for RPi connections
- LEDs on the RPi will start blinking indicating that it is booting up

STEP [2]:
Access the RPi in the Edge-lab

2.1 Open up remote desktop connection (using the VNC server)

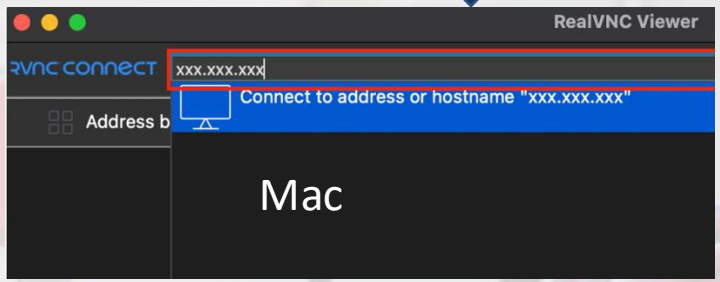
Each RPi has unique alpha-numeric name (e.g., 007, 015 etc)

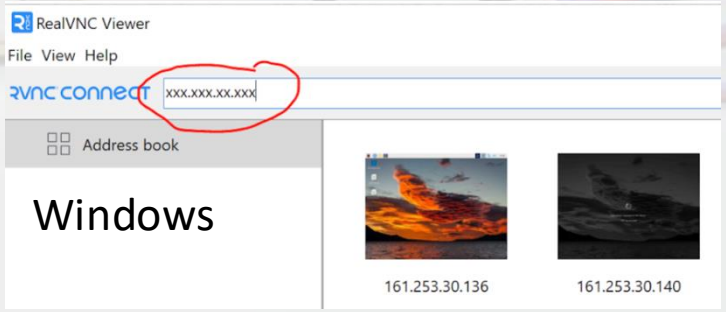
- Locate the Pi-name and the IP address on the <128.164.139.xx>

OR

Each RPi connected using an ethernet cable directly to your laptops

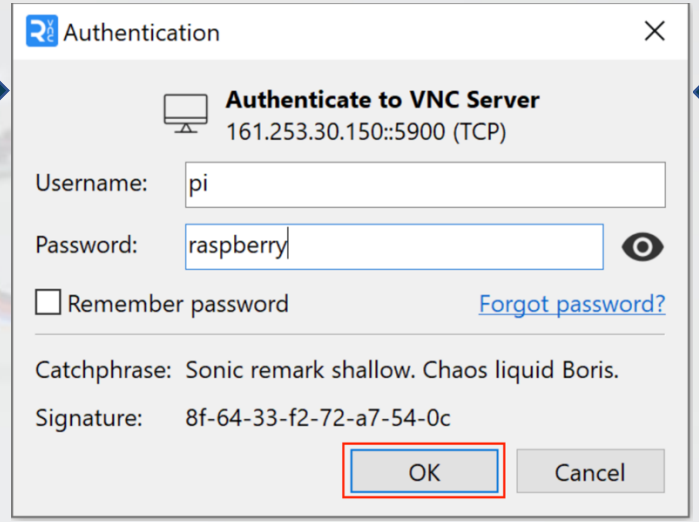
- raspberrypi###.local





2.2 Once you are connected you will

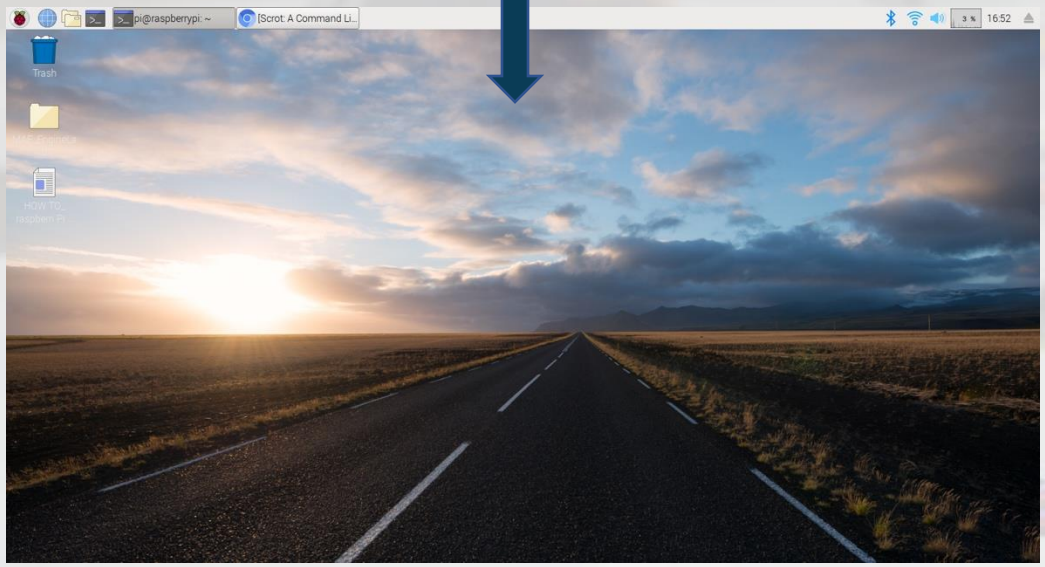
- See Authentication box below
- Type in the Username and Password



STEP [3]:

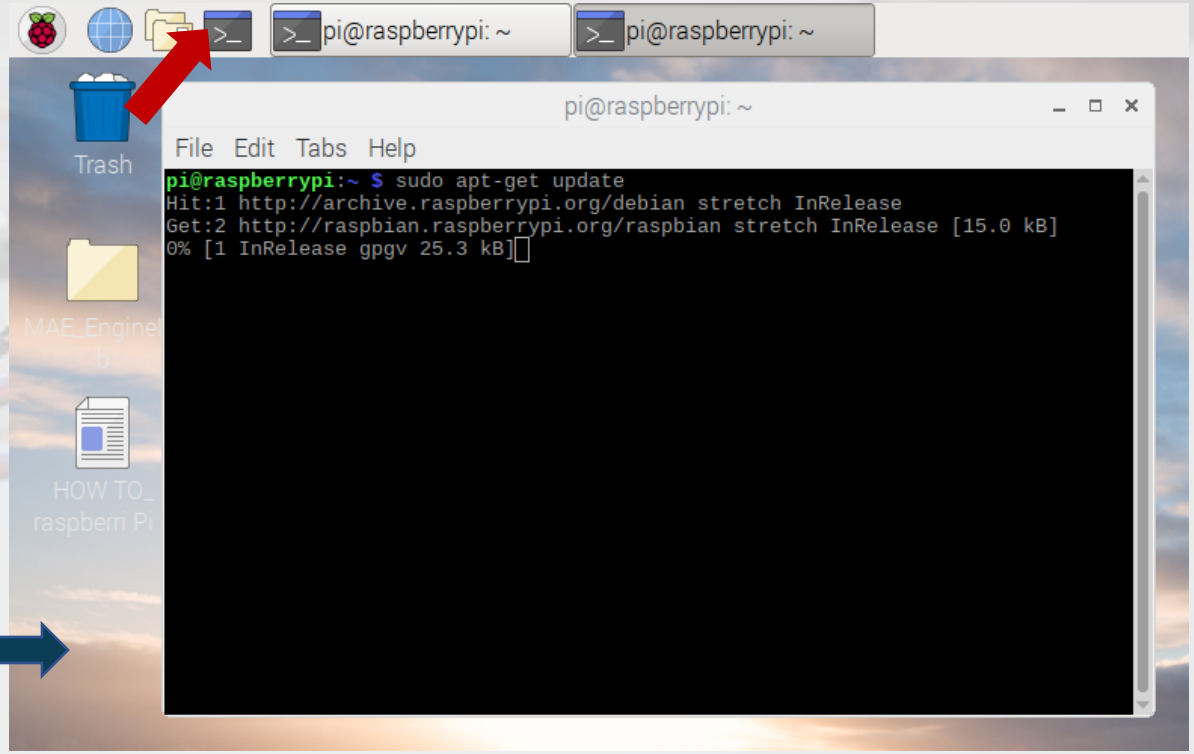
Now that you accessed the RPi…

You will see a screen like the one shown below



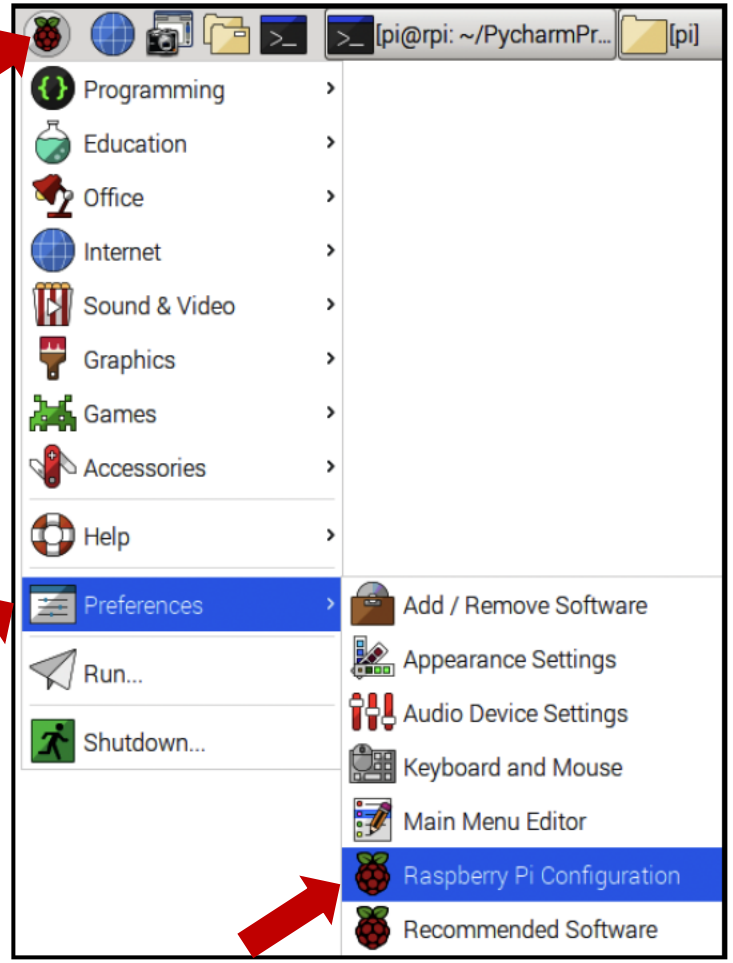
3.1 Click on terminal(shown with a red arrow below)

- At the prompt type: **sudo apt-get update**
- Wait for the updates to complete
- Then type: **sudo apt-get upgrade**
- If you get the following prompt
  - **Do you want to continue [Y/n]**
  - **Type: y**
  - **And hit "Enter" on your keyboard and let the upgrades complete**

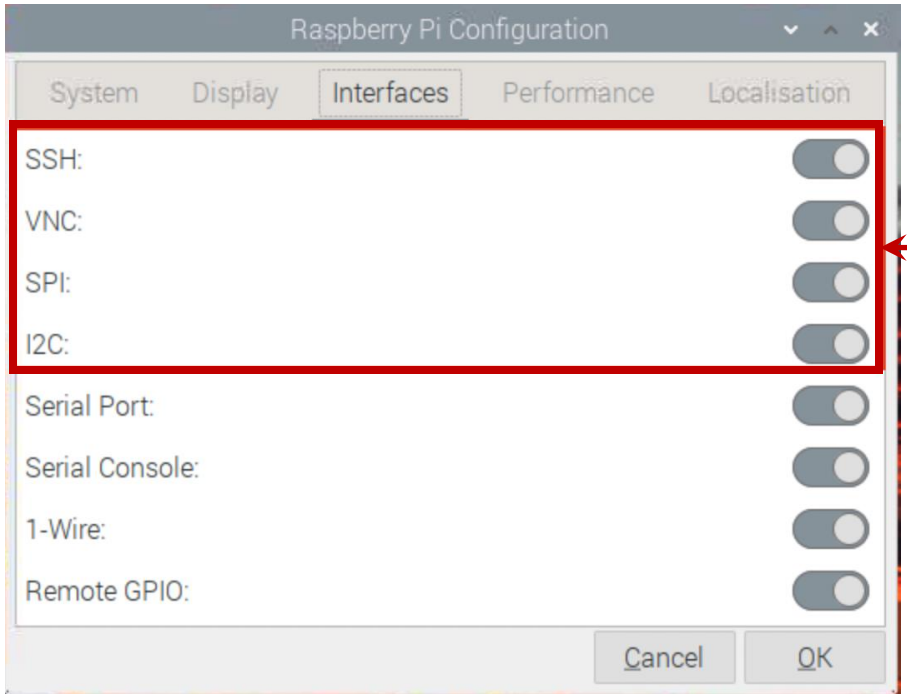


3.2 Students should get the RPis to this step before the laboratory modules begin.

Couple of checks and balances

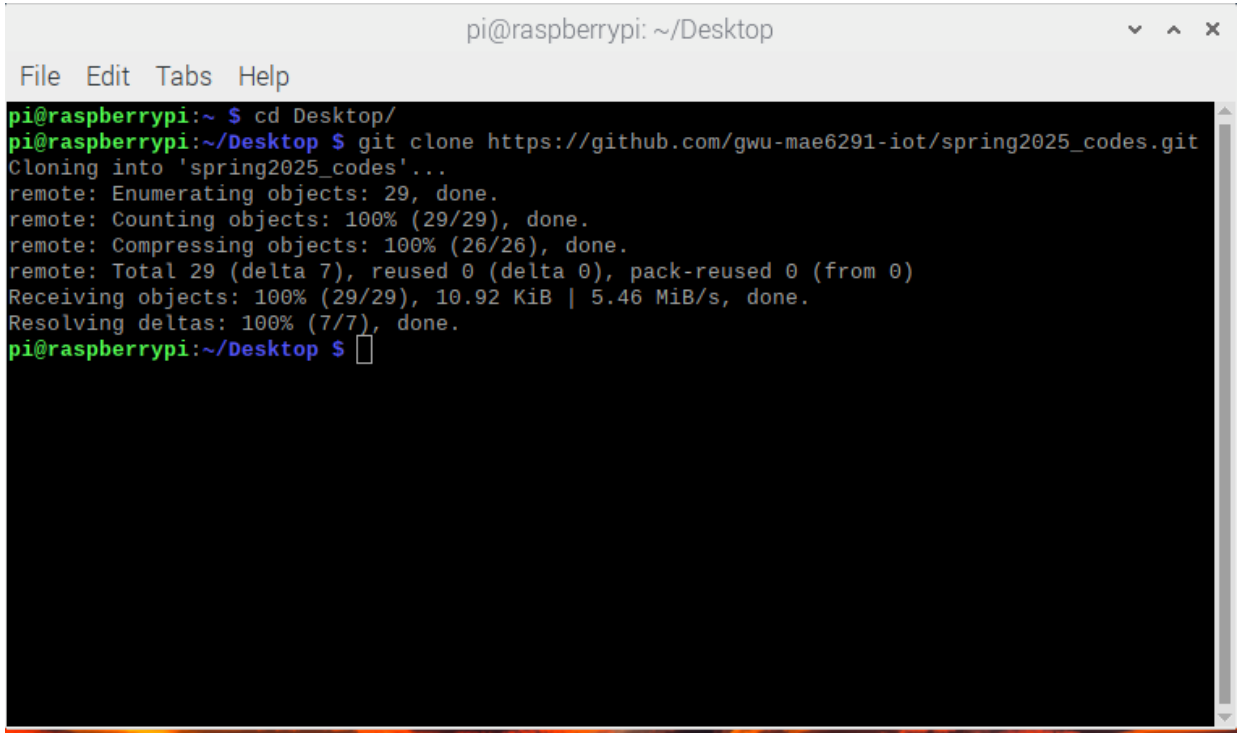


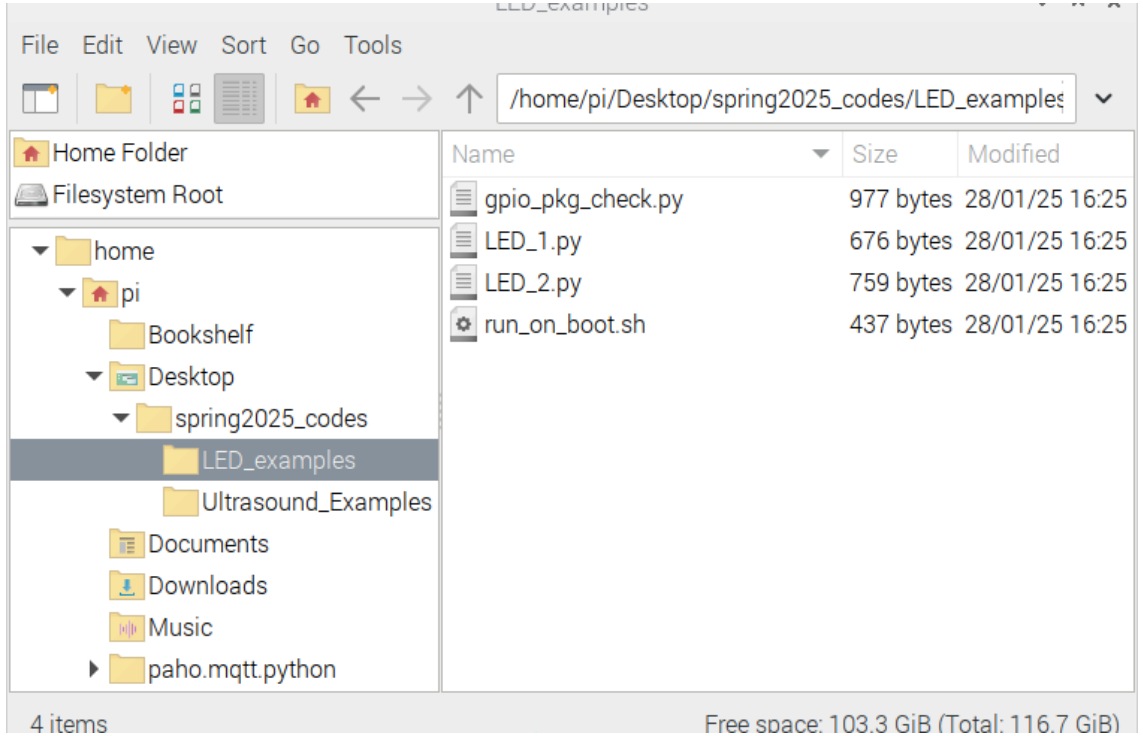
Make sure these are enabled



Downloading folders from course git-repository

$ git clone https://github.com/gwu-mae6291-iot/spring2025_codes.git





“things” with compute

“things” with compute == edge compute

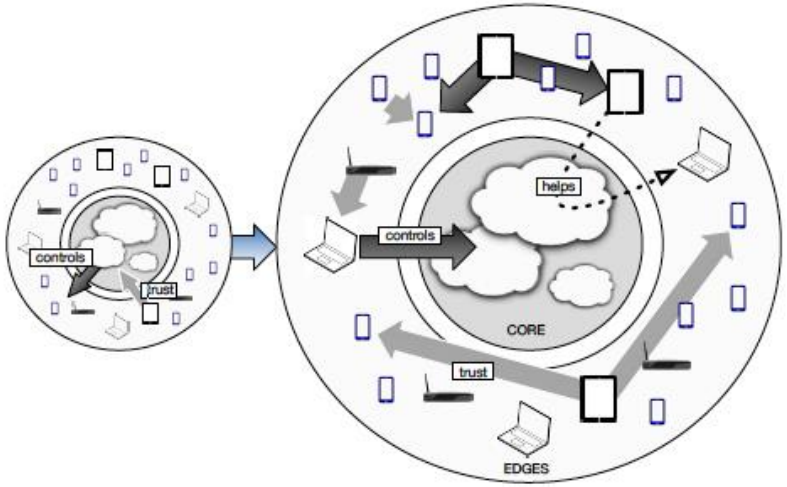
Sources:
Edge computing acts on data at the source: https://www.ibm.com/cloud/what-is-edge-computing
What is Edge computing? https://www.youtube.com/watch?v=3hScMLH7B4o&t=6s

Edge computing is a distributed computing framework that brings enterprise applications closer to data sources such as IoT devices or local edge servers.

This **proximity to data** at its source can deliver strong business benefits, including faster insights, improved response times and better bandwidth availability.

Sources:
Garcia Lopez, Pedro; *et al*., (30 September 2015). "Edge-centric Computing". *ACM SIGCOMM Computer Communication Review*. **45** (5): 37–42. doi:10.1145/2831347.2831354.
M. Rabinovich, Z. Xiao, and A. Aggarwal. Computing on the edge: A platform for replicating internet applications. In Web content caching and distribution, pages 57{77. Springer, 2004.
Difference Between Edge Computing and Fog Computing, https://www.geeksforgeeks.org/difference-between-edge-computing-and-fog-computing/
Kazaa, https://en.wikipedia.org/wiki/Kazaa#/media/File:Kazaa_(logo).png
Napster, https://en.wikipedia.org/wiki/Napster#/media/File:Napster.png

"Edge Computing" was coined around 2002



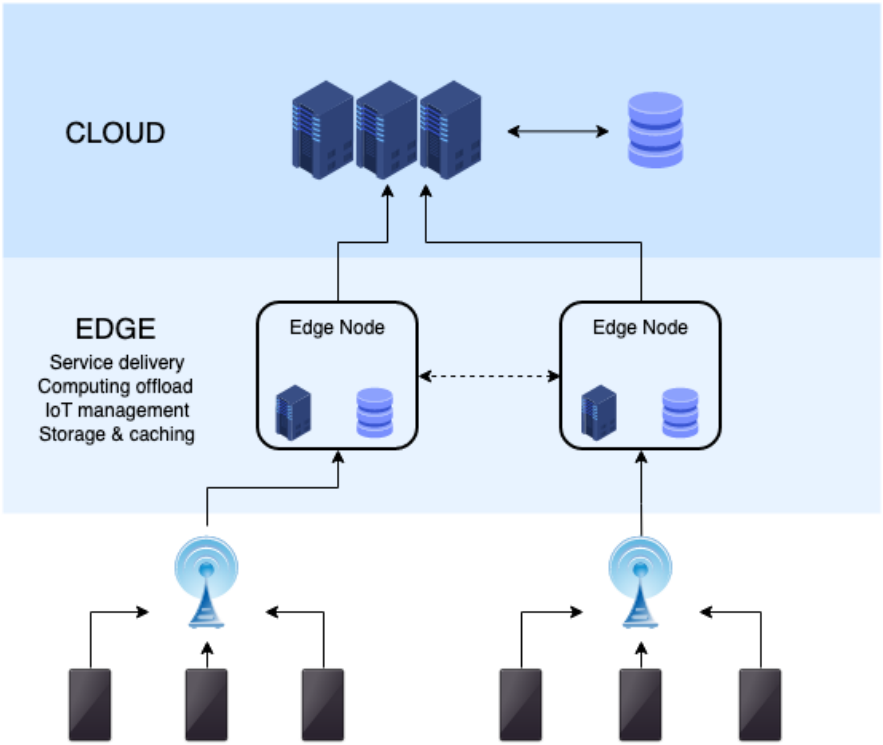
Content Delivery Networks:
It was mainly associated with the deployment of applications over CDNs, when some large companies announced deals to distribute software through CDN edge servers.

P2P computing:
This is another field closely related to edge computing, it is also its main precursor. The term P2P was first introduced around 2000 with the appearance of popular file-sharing systems such as Napster and Kazaa.

Fog computing:
Fog Computing is a recent research field that has substantial overlap with Edge-centric Computing. Proximity to end-users, dense geographical distribution, and support for mobility are the main distinguishing characteristics of Fog Computing.

Sources:
Edge computing, https://en.wikipedia.org/wiki/Edge_computing#/media/File:Edge_computing_infrastructure.png
What is Edge Computing: The Network Edge Explained, https://www.cloudwards.net/what-is-edge-computing/

Edge computing is a form of distributed computing



Classical Paradigm

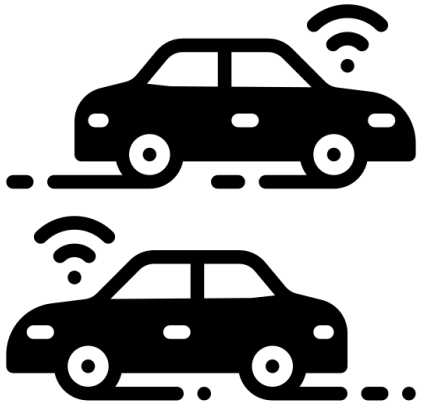
- **Distributed computing**
  - Covers a broad range of technologies
  - Earliest success stories could be considered
    - local area networks and
    - the first internet, ARPANET (1960s).

New Paradigm

- **Decentralized, distributed computing**
  - **Proximity to data:** Moving the computer workload closer to the data source
    - reduces latency
    - bandwidth and
    - overhead for the centralized data center

Four (near future) Edge computing examples

Icon sources:
Autonomous Car by Tippawan Sookruay from https://thenounproject.com/browse/icons/term/autonomous-car/
clean energy by Juicy Fish from https://thenounproject.com/browse/icons/term/clean-energy
Health Care by mynamepong from https://thenounproject.com/browse/icons/term/health-care/
Smart City by dDara from https://thenounproject.com/browse/icons/term/smart-city/



Autononous Cars

- Chevrolet collected 4,220 terabytes of data from customer's cars.

McKinsey forecasts that this could grow into a $450 to 750 billion market by 2030.

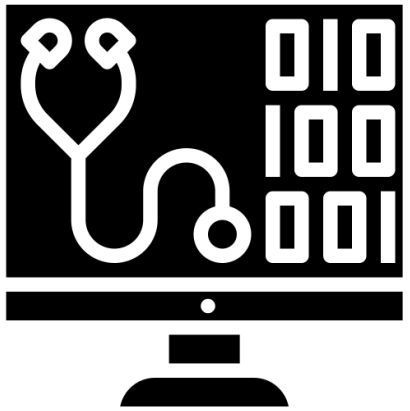
Source: https://www.autoblog.com/2017/02/21/race-for-autonomous-cars-is-over-mcelroy-autoline-opinion/

Clean energy technology

- Data centers use an estimated 200 terawatt hours (TWh) of electricity annually
- ~ 50% of all electricity currently used for all global transport.

Edge computing can significantly reduce the amount of time and power, data centers need to use to process data.

Source:
https://www.forbes.com/sites/forbestechcouncil/2022/03/18/how-machine-learning-and-edge-computing-powers/ustainability/?sh=483ebd025fab

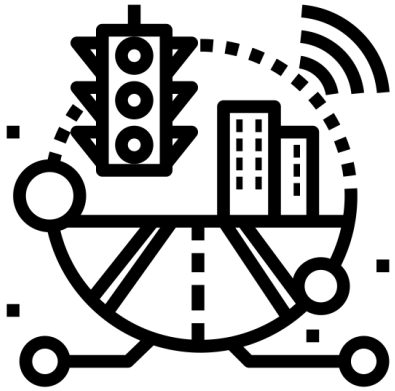


Transforming health care at the edge

- 10 to 15 connected device per US hospital bed
- 3 million data points generated by the average clinical trial
- 30% of all global stored data is from health care

75% of data will be generated at he edge by 2025

Source:
https://www.technologyreview.com/2021/06/10/1026038/transforming-health-care-at-the-edge/



Edge AI: Tackling Traffic Management

- A pilot system deployed at Pittsburgh, Pennsylvania, has reportedly
  - reduced travel time by 26%
  - idling time by 41%, and
  - emissions by 21 %.

The INRIX Global Traffic Scorecard:
World's 20 most congested cities lost between 164 and 210 hours in congestion per capita through 2018.

Source:
https://www.iotforall.com/busting-traffic-woes-with-5g-and-edge-ai

School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

GW

Sources:
Edge computing, https://en.wikipedia.org/wiki/Edge_computing#/media/File:Edge_computing_infrastructure.png
What is Edge Computing: The Network Edge Explained, https://www.cloudwards.net/what-is-edge-computing/
What is IoT Edge computing? https://www.redhat.com/en/topics/edge-computing/iot-edge-computing-need-to-work-together
What is edge computing?, https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-edge-computing/
Secure Edge Computing in IoT Systems: Review and Case Studies , https://par.nsf.gov/servlets/purl/10097337

Benefits of Edge computing

- **Reduced latency of communication** between IoT devices and the central IT networks.
- **Faster response times** and increased operational **efficiency.**
- **Improved network bandwidth.**
- **Continued systems operation offline** when a network connection is lost.
- **Local data processing**, aggregation, and rapid decision making via analytics algorithms and machine learning.

There are concerns!

- **User Privacy** between IoT devices and the central IT networks.
- **Optimization metrics:** There are several layers with various computation abilities in edge computing for choosing an optimal workload allocation.
- **Task-offloading:** Utilizing edge nodes for computation offloading is a concern due to the problem of adequately segmenting computational tasks.
- **Public accessibility of edge nodes:** When an edge device (e.g., a base station, switch, and router) is intended to be used for public access.

What's an "edge" device?

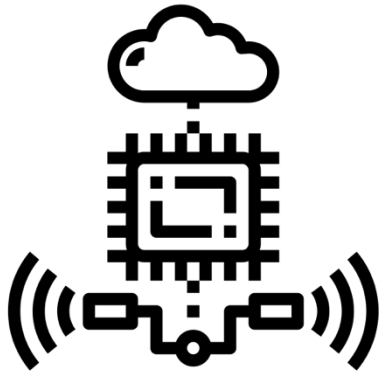
Sources:
Calculator by Markus from https://thenounproject.com/browse/icons/term/calculator/
industrial transformation by dDara from https://thenounproject.com/browse/icons/term/industrial-transformation
Definition: Edge Device, https://www.techtarget.com/searchnetworking/definition/edge-device

Paradigm #1

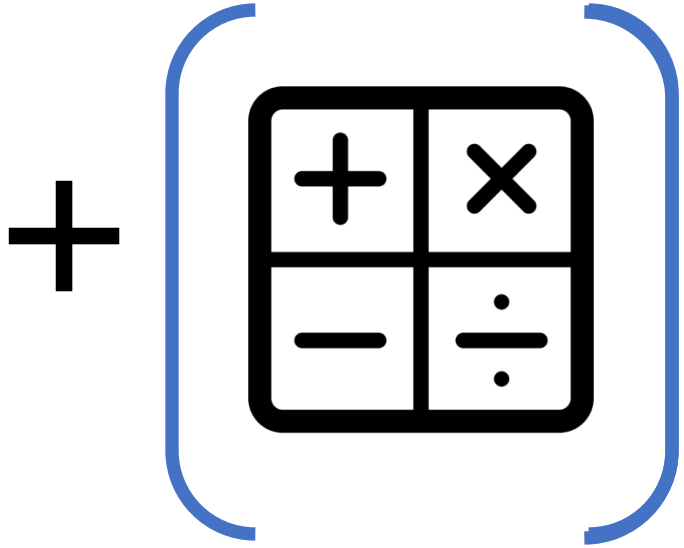
An **edge device** is any piece of hardware that controls data flow at the boundary between two networks.

- Essentially serve as network entry -- or exit -- points.
- Common functions of edge devices are the transmission, routing, processing, monitoring, filtering, translation and storage of data passing between networks.

Paradigm #2

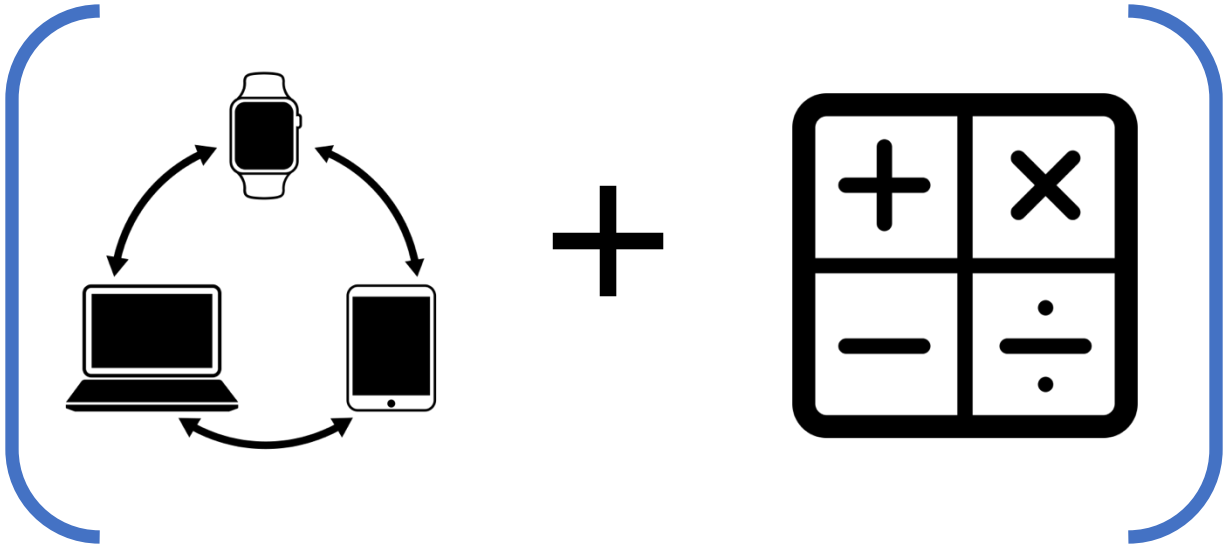


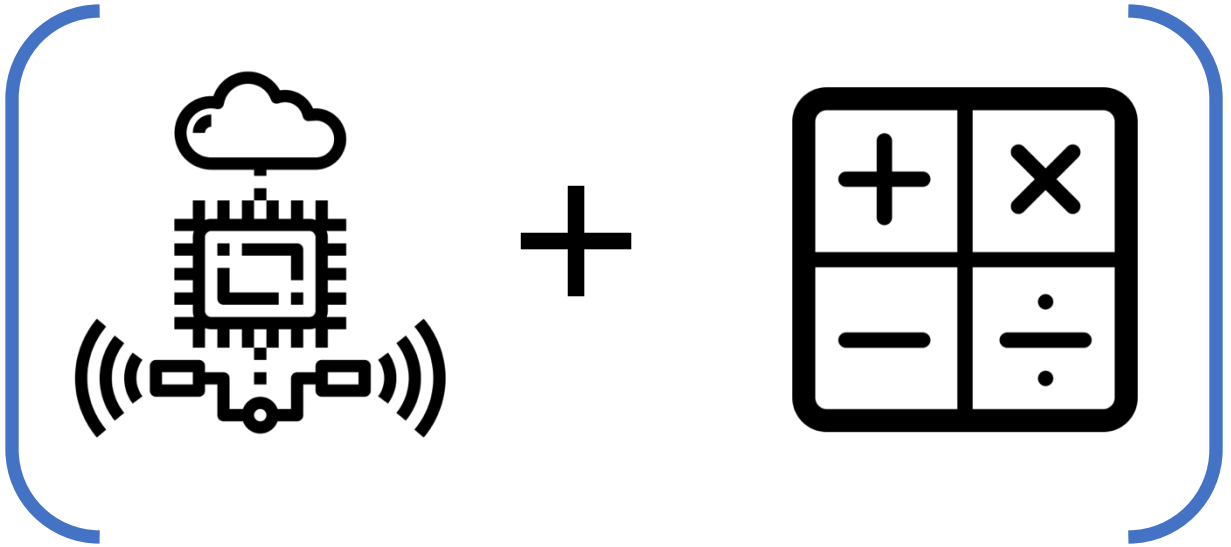
Paradigm #3



Overview of Computing in IoT

Sources:
internet of things by Davo Sime from https://thenounproject.com/browse/icons/term/internet-of-things/
Calculator by Markus from https://thenounproject.com/browse/icons/term/calculator/
industrial transformation by dDara from https://thenounproject.com/browse/icons/term/industrial-transformation
What is IoT Edge computing?, https://www.redhat.com/en/topics/edge-computing/iot-edge-computing-need-to-work-together





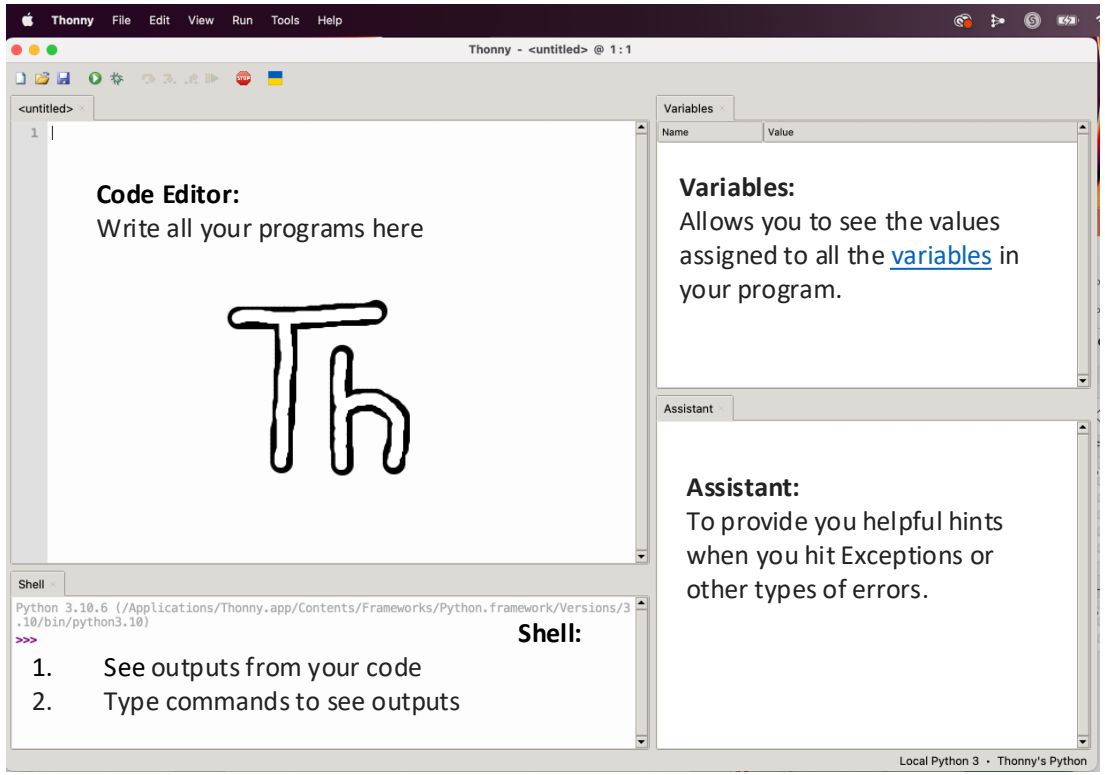
The Internet of Things (IoT) is made up of smart devices connected to a network—sending and receiving large amounts of data to and from other devices—which produces a large amount of data to be processed and analyzed.

Edge computing, a strategy for computing on location where data is collected or used, allows IoT data to be gathered and processed at the edge, rather than sending the data back to a datacenter or cloud.

Let’s get to know some programming paradigms

Thonny integrated development environment (IDE)

Sources:
Thonny (IDE): https://en.wikipedia.org/wiki/Thonny
Thonny: The Beginner-Friendly Python Editor: https://realpython.com/python-thonny/



Code Editor:
Write all your programs here

Variables:
Allows you to see the values assigned to all the variables in your program.

Assistant:
To provide you helpful hints when you hit Exceptions or other types of errors.

Shell:
1. See outputs from your code
2. Type commands to see outputs

Allows you to run the code, i.e., "Do what I told you do do!".

Allows you to debug your code.
A bug is another name for a problem.

Allows you to open a file that already exists on your computer

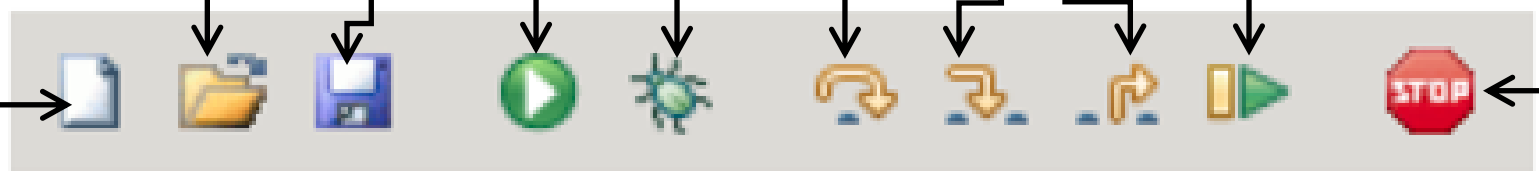
Create a new file

Save your code. Press this early and often.

The arrow icons allow you to run your programs step by step.
These icons are used after you press the bug icon

The resume icon allows you to return to play mode from debug mode.

The stop icon allows you to stop running your code



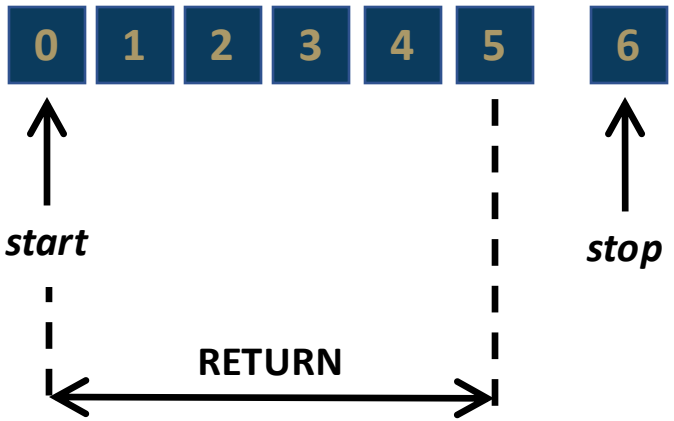
Built-in function range()

Python's range()function
- returns a sequence of numbers
- works only with integers

start:	at the value (default = 0)
step:	up or down at the increment value (default = 1)
stop:	at the value but not including it

range(stop)

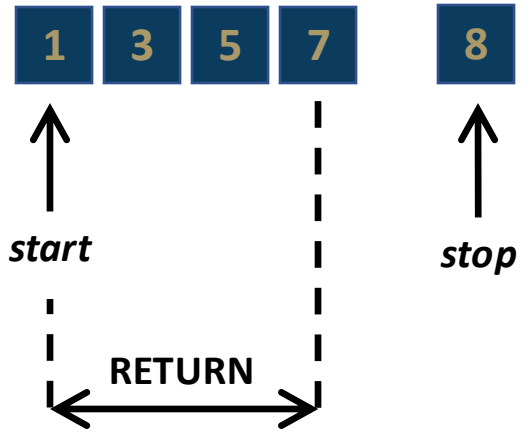
```
>>> range(6)
```



range(start, stop)

range(*start*, *stop*, *step*)

```
>>> range(1,8,2)
```

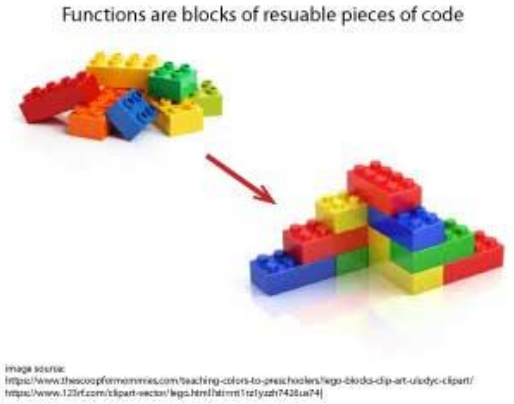


range()
- can be utilized in (for) loops
- to specify a range to iterate or do repetitions

Demos

Syntax and Skeleton of a user-defined function

```
def name(parameters):

        statement
        statement
        . . .

        return value
```



Function name: Identifier by which it is called in the program

(Optional) **Arguments: values passed to the function**

Function Declaration: Starts with "def" that is not indented

```
def func_name(parameters):
```

Colon; Don't miss it!

Indentation: Tab or 4 spaces for each statement

```
        statement
        statement
        . . .

        return value
```

Body: Statements executed each time a function is called

(Optional) **return value: Can end function call and send data back to the main program**

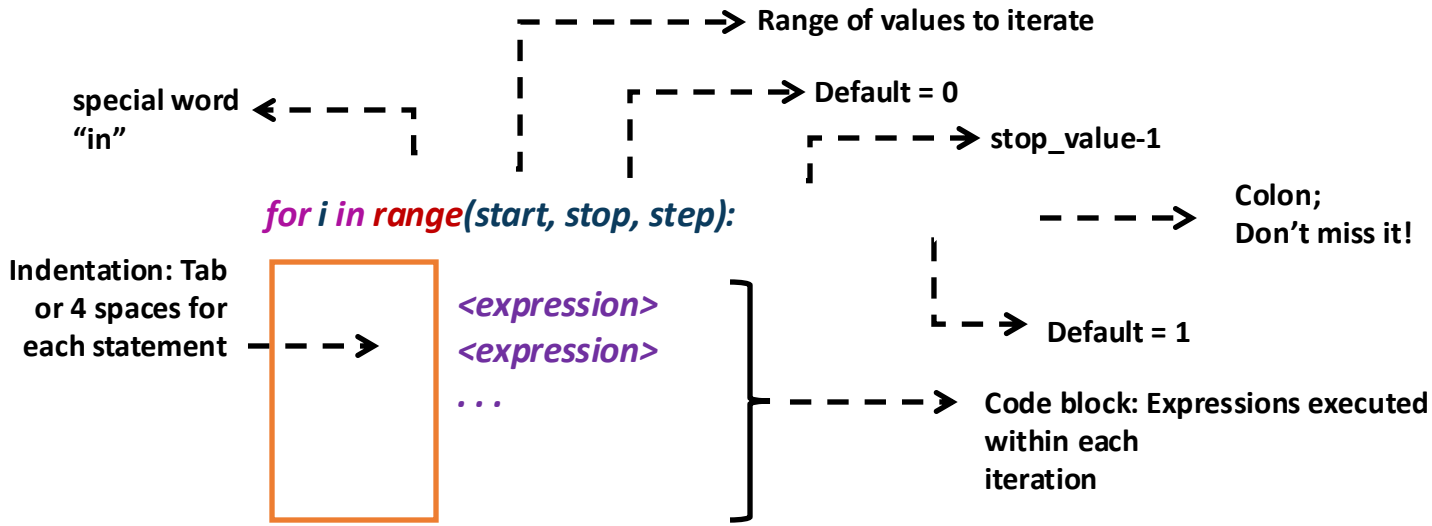
Function definition

```
func_name()
```

Function call

Skeleton of the for-loop

```
for i in range(start, stop, step):

        <expression>
        <expression>
        . . .
```



Sources:
Exceptions in Python || Python Tutorial || Learn Python Programming: https://youtu.be/nlCKrKGHSSk
Python Try Except: https://www.w3schools.com/python/python_try_except.asp

Skeleton of
try:
except:
else:
finally:

```
try:
    statement
    function
except KeyboardInterrupt:
    statement
    function
else:
    statement
    function
finally:
    statement
    function
```

Exception Clauses

```
try:
  # Runs first
  < code >
except:
  # Runs if exception occurs in try block
  < code >
else:

finally:
```

Exception Clauses

```
try:
  # Runs first
  < code >
except:
  # Runs if exception occurs in try block
  < code >
else:
  # Executes if try block *succeeds*
  < code>
finally:
```

Exception Clauses

```
try:
  # Runs first
  < code >
except:
  # Runs if exception occurs in try block
  < code >
else:
  # Executes if try block *succeeds*
  < code>
finally:
  # This code *always* executes
  < code >
```

The try block lets you test a block of code for errors.
The except block lets you handle the error.
The else block lets you execute code when there is no error.
The finally block lets you execute code, regardless of the result of the try- and except blocks.

Skeleton of the updated Python program written for Raspberry Pi

```
import library1 as name1
import RPi.GPIO as GPIO
import time
```

Import libraries that are relevant for interaction with the Raspberry Pi hardware such as GPIO pins, camera ports etc.

```
GPIO.setmode(GPIO.BOARD)

GPIO.setup(12, GPIO.OUT)
```

Set up GPIO pins as data outputs or inputs for sensors and actuators

```
def function_name(arguments):
        statement
        statement
        . . .
        return value
```

Create user-defined functions to modularize your code and make it easy to work.

Create user-defined functions to release the GPIO pins

```
if __name__ == "__main__":
        try:
                function_name1()
        except KeyboardInterrupt:
                function_name2()
```

Create entry point into the program and pull all functions in

Create a keyboard-interrupt exception clause

Let’s blink some LEDs

Preliminaries:

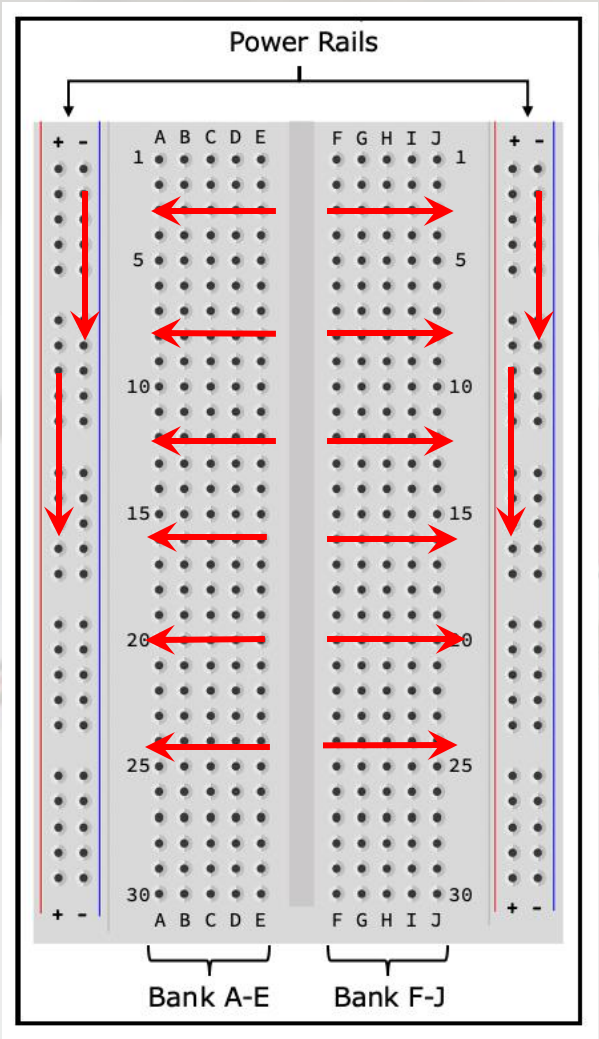
1. **git-clone the repo preferably on the desktop**

   git clone https://github.com/gwu-mae6291-iot/spring2025_codes.git

2. **I have the files on a USB in case there is a slowdown!**

Connect the Raspberry Pi Model 3 B+ (RPi) to a bread board

Sources of the breadboard image:
Gary Smart, Practical Python Programming for IoT



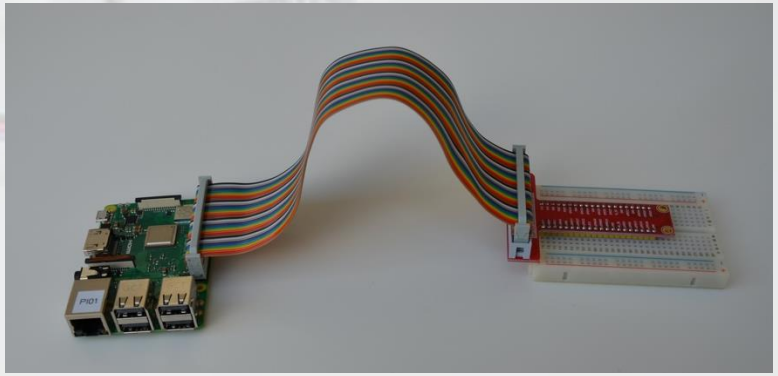
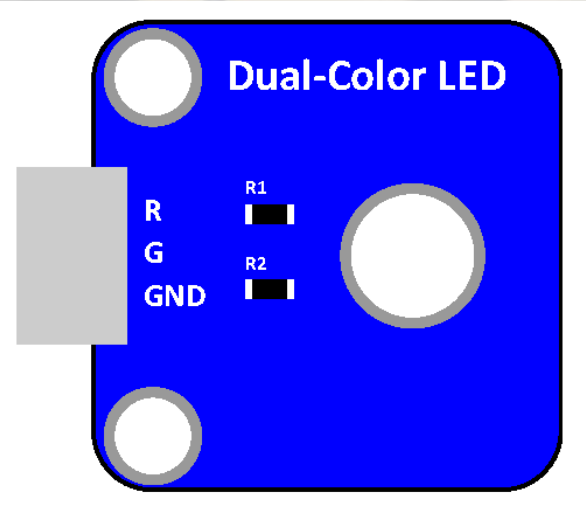


Photo: Kartik Bulusu

Know your Light Emitting Diode (LED)



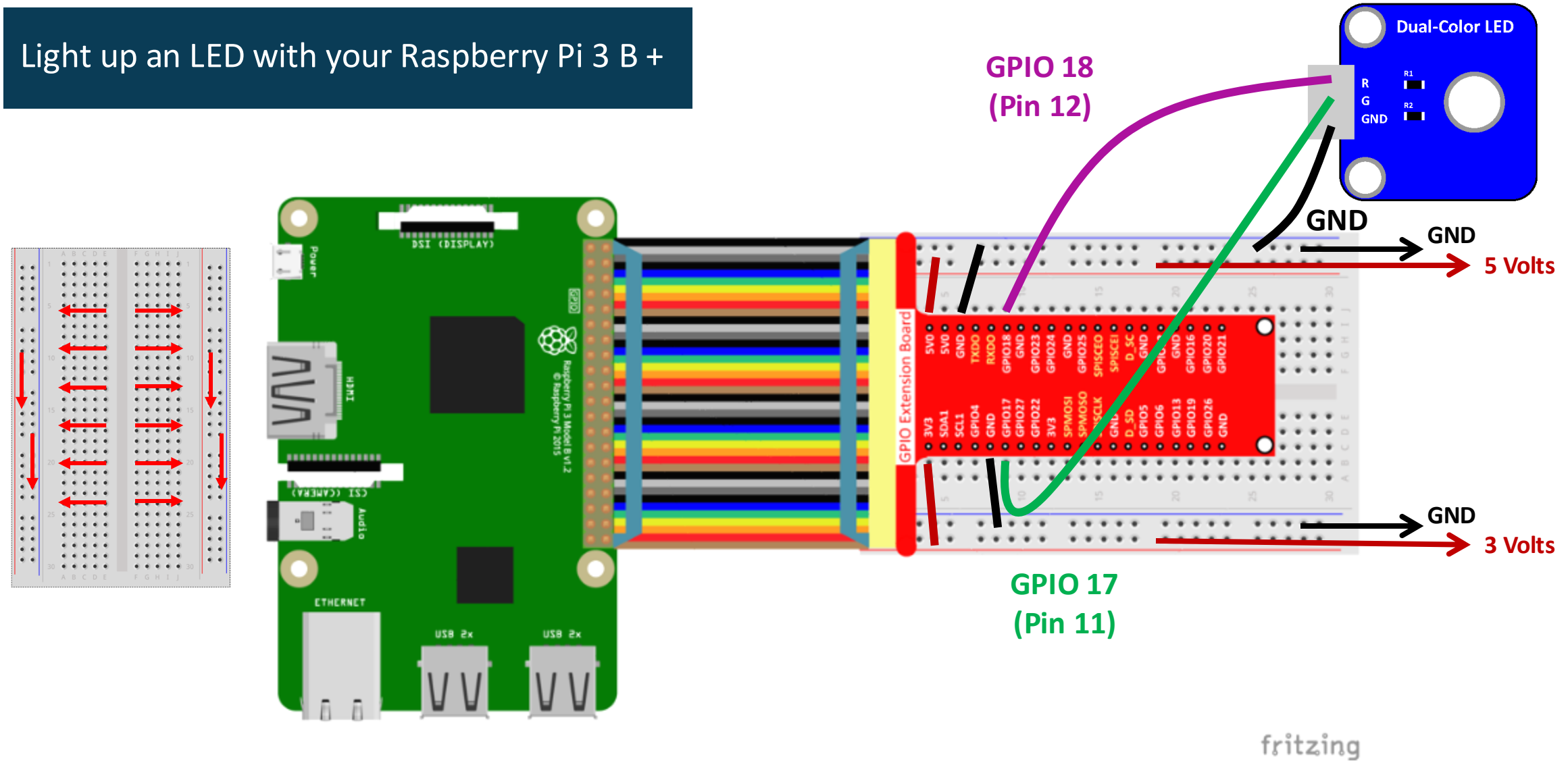
Source:
https://www.sunfounder.com/learn/lesson-1-dual-color-led-sensor-kit-v2-0-for-b.html

A dual-color light emitting diode (LED) is capable of emitting two different colors of light, typically red and green.

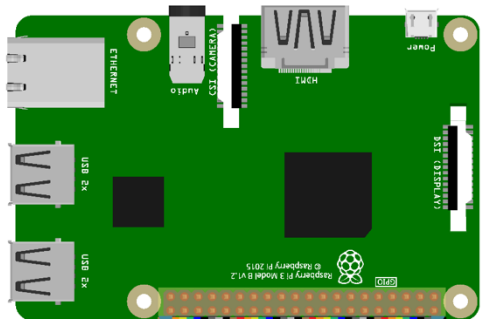
Application:
Variety of devices, such as televisions, digital cameras, and remote controls deploy these type LEDs.

Connector:
3-pin anti-reverse cable

Light up an LED with your Raspberry Pi 3 B +

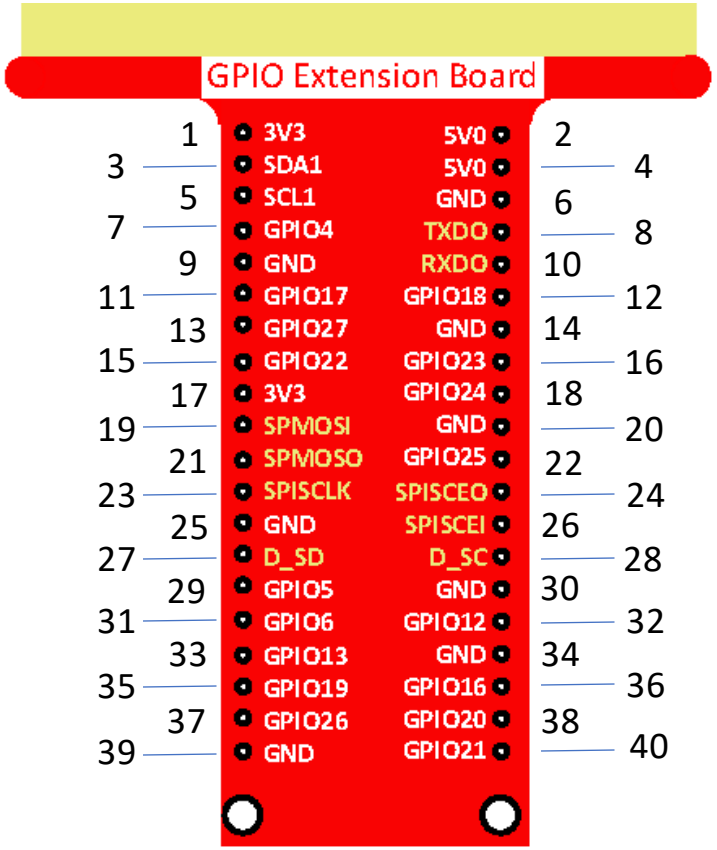


How a python code can light up your LED with Raspberry Pi Model 3 B+ (RPi)

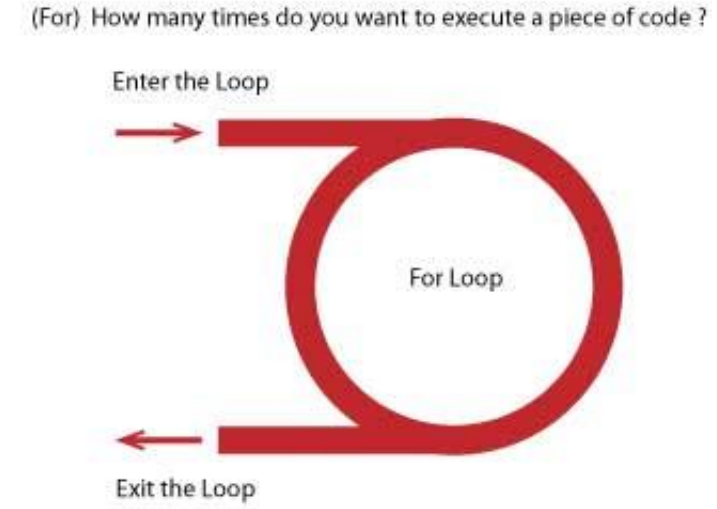


```
import RPi.GPIO as GPIO
import time
```

```
GPIO.setmode(GPIO.BOARD)
```

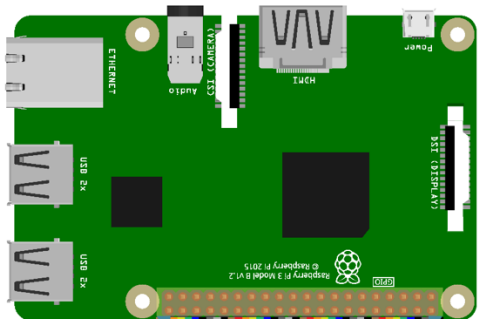



```
GPIO.setup(12, GPIO.OUT)
```



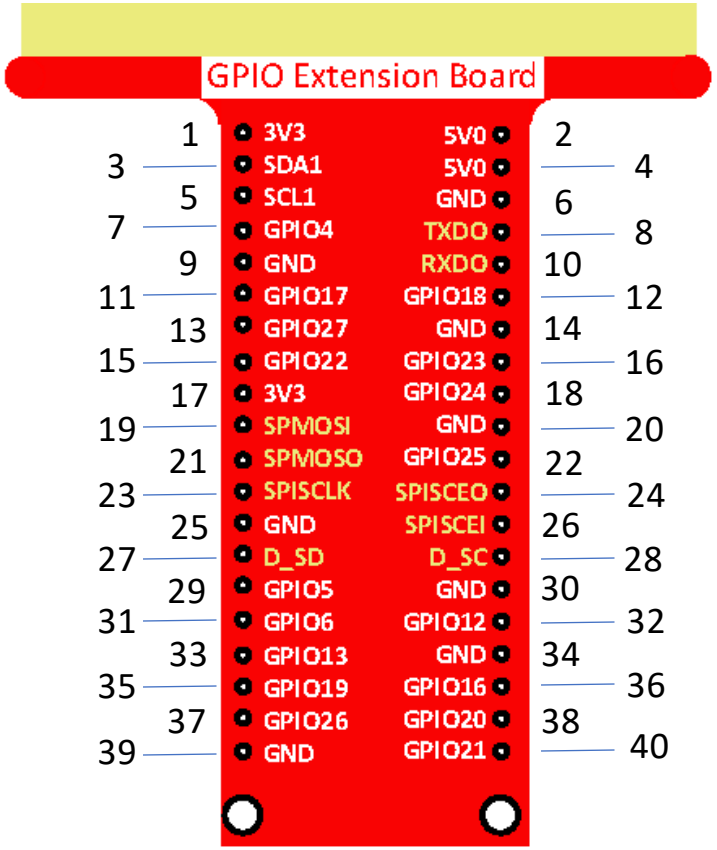

```
for i in range(0,15):

    GPIO.output(12, GPIO.HIGH)
    time.sleep(0.5)
    GPIO.output(12, GPIO.LOW)
    time.sleep(0.5)
    print(i)

GPIO.cleanup()
```

Another python code to kick start your Raspberry Pi Model 3 B+ (RPi)



```
import RPi.GPIO as GPIO
import time
```

```
GPIO.setmode(GPIO.BOARD)
```




```
GPIO.setup(12, GPIO.OUT)
```

```
def loop():
        while True:
                GPIO.output(12, GPIO.HIGH)
        time.sleep(0.5)
            GPIO.output(12, GPIO.LOW)
        time.sleep(0.5)
```

```
def destroy():
    GPIO.output(12, GPIO.LOW)
            # Turn off all leds
    GPIO.cleanup()
```

```
if __name__ == "__main__":
    try:
        loop()
    except KeyboardInterrupt:
        destroy()
```


Photo: Kartik Bulusu
Someone should summarize what we learned today