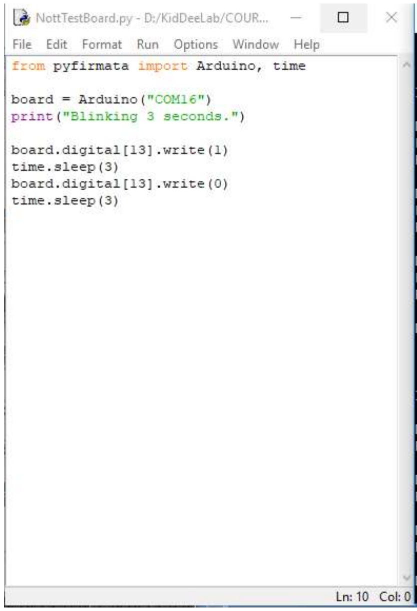
EE3704 Embedded System

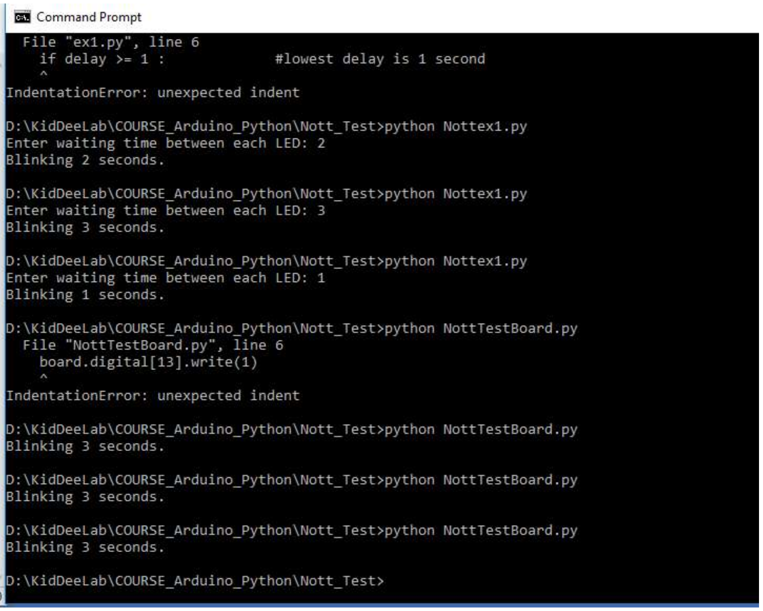
Chapter 10

Presented by
Asst. Prof. Dr.Narong Aphiratsakun

Python IDE


NottTestBoard.py - D:/KidDeeLab/COUR...
File Edit Format Run Options Window Help
Ln: 10 Col: 0

```
from pyfirmata import Arduino, time

board = Arduino("COM16")
print("Blinking 3 seconds.")

board.digital[13].write(1)
time.sleep(3)
board.digital[13].write(0)
time.sleep(3)
```



```
  File "ex1.py", line 6
    if delay >= 1 :            #lowest delay is 1 second
    ^
IndentationError: unexpected indent

D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test>python Nottex1.py
Enter waiting time between each LED: 2
Blinking 2 seconds.

D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test>python Nottex1.py
Enter waiting time between each LED: 3
Blinking 3 seconds.

D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test>python Nottex1.py
Enter waiting time between each LED: 1
Blinking 1 seconds.

D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test>python NottTestBoard.py
  File "NottTestBoard.py", line 6
    board.digital[13].write(1)
    ^
IndentationError: unexpected indent

D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test>python NottTestBoard.py
Blinking 3 seconds.

D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test>python NottTestBoard.py
Blinking 3 seconds.

D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test>python NottTestBoard.py
Blinking 3 seconds.

D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test>
```

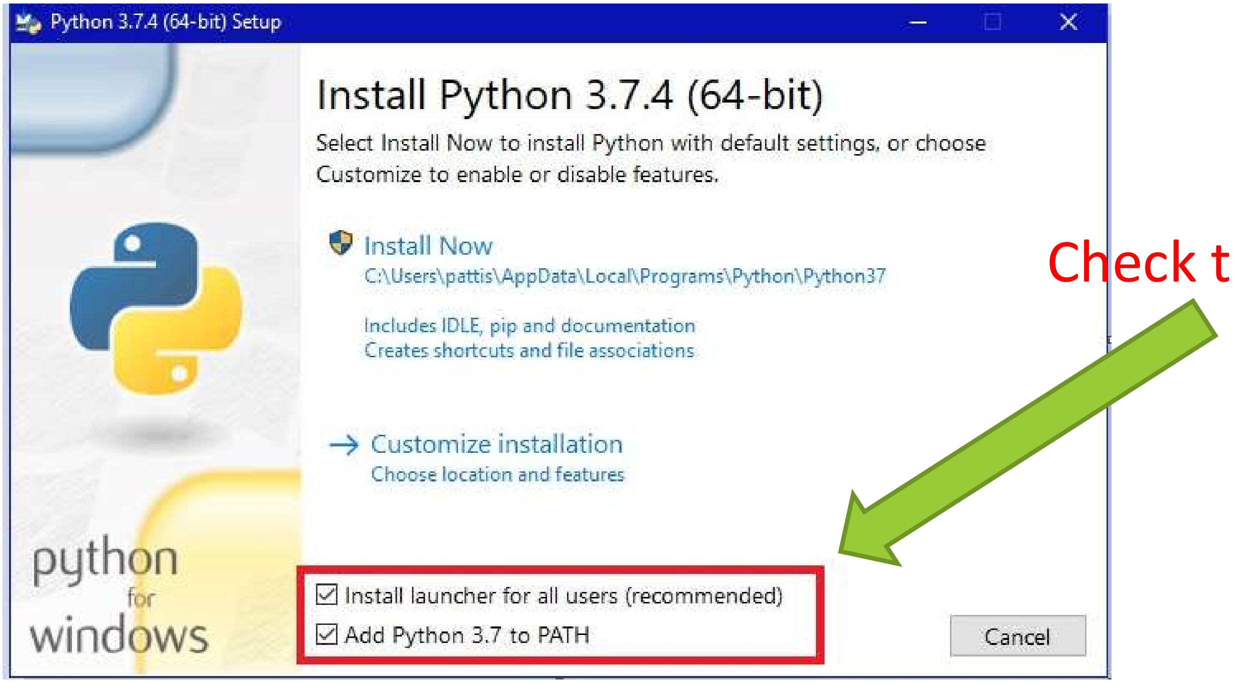
IDLE

- Write and Compile
- Show print output

CMD

- To install some functions

Python IDE



Python 3.7.4 (64-bit) Setup

Install Python 3.7.4 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\pattis\AppData\Local\Programs\Python\Python37

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.7 to PATH

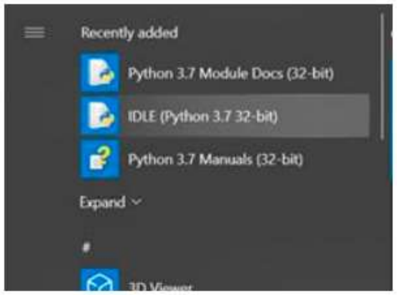
Check these boxes

PIP installed

1

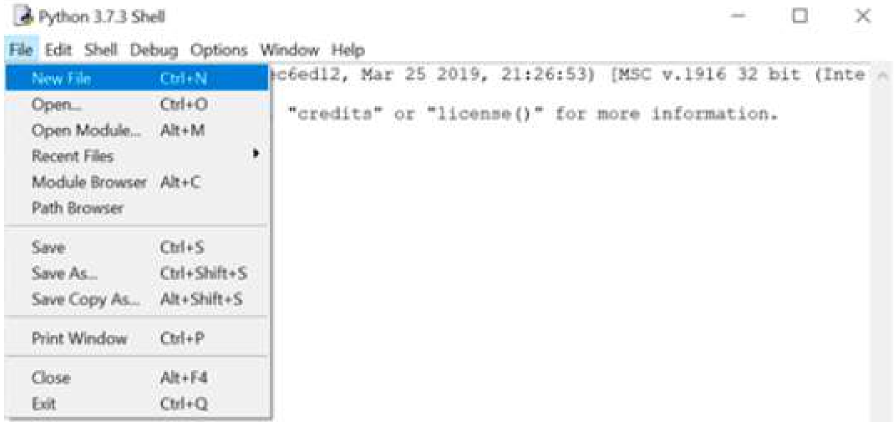
In case of failing installing **pyFirmmata**. This is another way to come over, please follow these steps.

- After finishing installing Python, open IDLE



2

- Go to tab File > New File

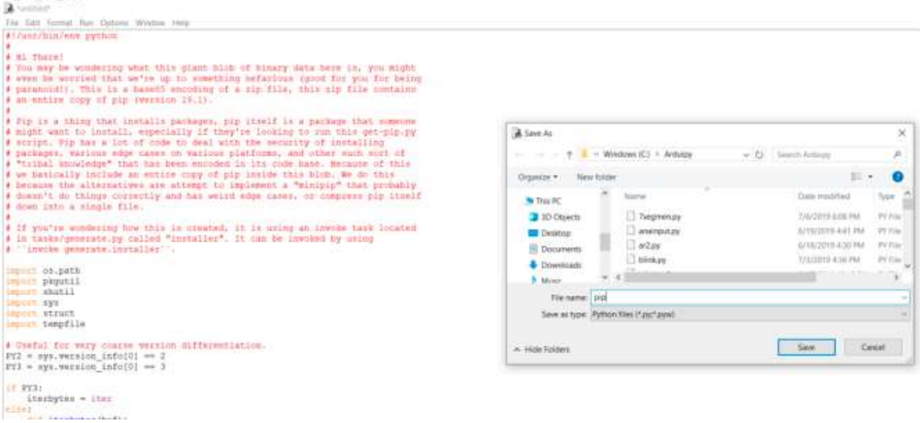


```
c6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Inte
 "credits" or "license()" for more information.
```

PIP installed

3

- Go to https://bootstrap.pypa.io/get-pip.py and copy the code to IDLE program. Save it as **pip.py**



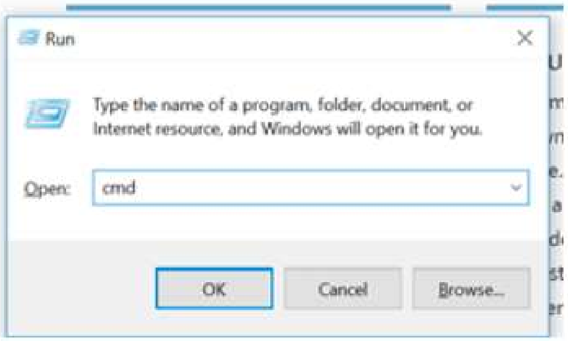
```
#!/usr/bin/env python
#
# Hi There!
# You may be wondering what this giant blob of binary data here is, you might
# even be worried that we're up to something nefarious (good for you for being
# paranoid!). This is a base85 encoding of a zip file, this zip file contains
# an entire copy of pip (version 19.1).
#
# Pip is a thing that installs packages, pip itself is a package that someone
# might want to install, especially if they're looking to run this get-pip.py
# script. Pip has a lot of code to deal with the security of installing
# packages, various edge cases on various platforms, and other such sort of
# "tribal knowledge" that has been encoded in its code base. Because of this
# we basically include an entire copy of pip inside this blob. We do this
# because the alternatives are attempt to implement a "minipip" that probably
# doesn't do things correctly and has weird edge cases, or compress pip itself
# down into a single file.
#
# If you're wondering how this is created, it is using an invoke task located
# in tasks/generate.py called "installer". It can be invoked by using
# ``invoke generate.installer``.

import os.path
import pkgutil
import shutil
import sys
import struct
import tempfile

# Useful for very coarse version differentiation.
PY2 = sys.version_info[0] == 2
PY3 = sys.version_info[0] == 3

if PY3:
    iterbytes = iter
else:
```

File name: pip

Save as type: Python files (*.py;*.pyw)

4

- Open command prompt by ⊞ + **R** "and type **cmd** and then click **OK**



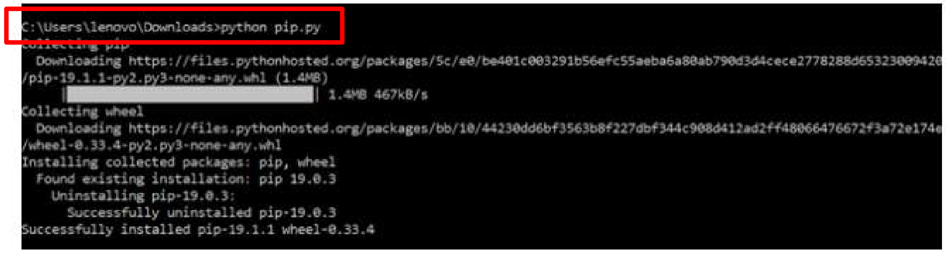
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: cmd

pyfirmata installed

- Before typing '**python pip.py**'. Make sure that you're already at the same as directory of pip.py

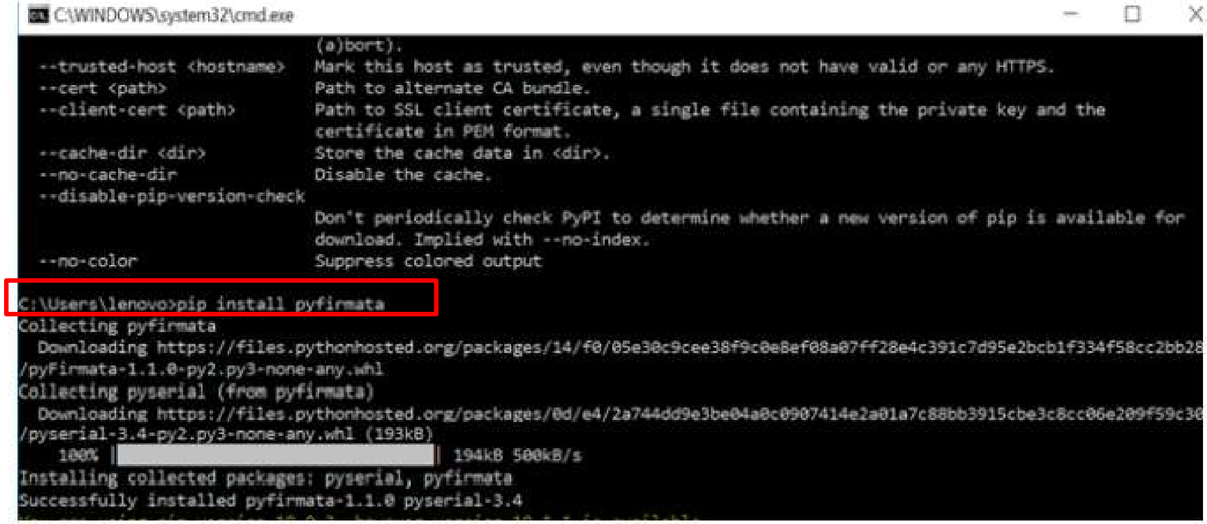
5



```
C:\Users\lenovo\Downloads>python pip.py
Collecting pip
  Downloading https://files.pythonhosted.org/packages/5c/e0/be401c003291b56efc55aeba6a80ab790d3d4cece2778288d65323009420
/pip-19.1.1-py2.py3-none-any.whl (1.4MB)
     |████████████████████████████████| 1.4MB 467kB/s
Collecting wheel
  Downloading https://files.pythonhosted.org/packages/bb/10/44230dd6bf3563b8f227dbf344c908d412ad2ff48066476672f3a72e174e
/wheel-0.33.4-py2.py3-none-any.whl
Installing collected packages: pip, wheel
  Found existing installation: pip 19.0.3
    Uninstalling pip-19.0.3:
      Successfully uninstalled pip-19.0.3
Successfully installed pip-19.1.1 wheel-0.33.4
```

- Finally, you can install **pyfirmata**

6



C:\WINDOWS\system32\cmd.exe

```
                            (a)bort).
  --trusted-host <hostname>   Mark this host as trusted, even though it does not have valid or any HTTPS.
  --cert <path>               Path to alternate CA bundle.
  --client-cert <path>        Path to SSL client certificate, a single file containing the private key and the
                              certificate in PEM format.
  --cache-dir <dir>           Store the cache data in <dir>.
  --no-cache-dir              Disable the cache.
  --disable-pip-version-check
                              Don't periodically check PyPI to determine whether a new version of pip is available for
                              download. Implied with --no-index.
  --no-color                  Suppress colored output

C:\Users\lenovo>pip install pyfirmata
Collecting pyfirmata
  Downloading https://files.pythonhosted.org/packages/14/f0/05e30c9cee38f9c0e8ef08a07ff28e4c391c7d95e2bcb1f334f58cc2bb28
/pyFirmata-1.1.0-py2.py3-none-any.whl
Collecting pyserial (from pyfirmata)
  Downloading https://files.pythonhosted.org/packages/0d/e4/2a744dd9e3be04a0c0907414e2a01a7c88bb3915cbe3c8cc06e209f59c30
/pyserial-3.4-py2.py3-none-any.whl (193kB)
    100% |████████████████████████████████| 194kB 500kB/s
Installing collected packages: pyserial, pyfirmata
Successfully installed pyfirmata-1.1.0 pyserial-3.4
```

Basic Python

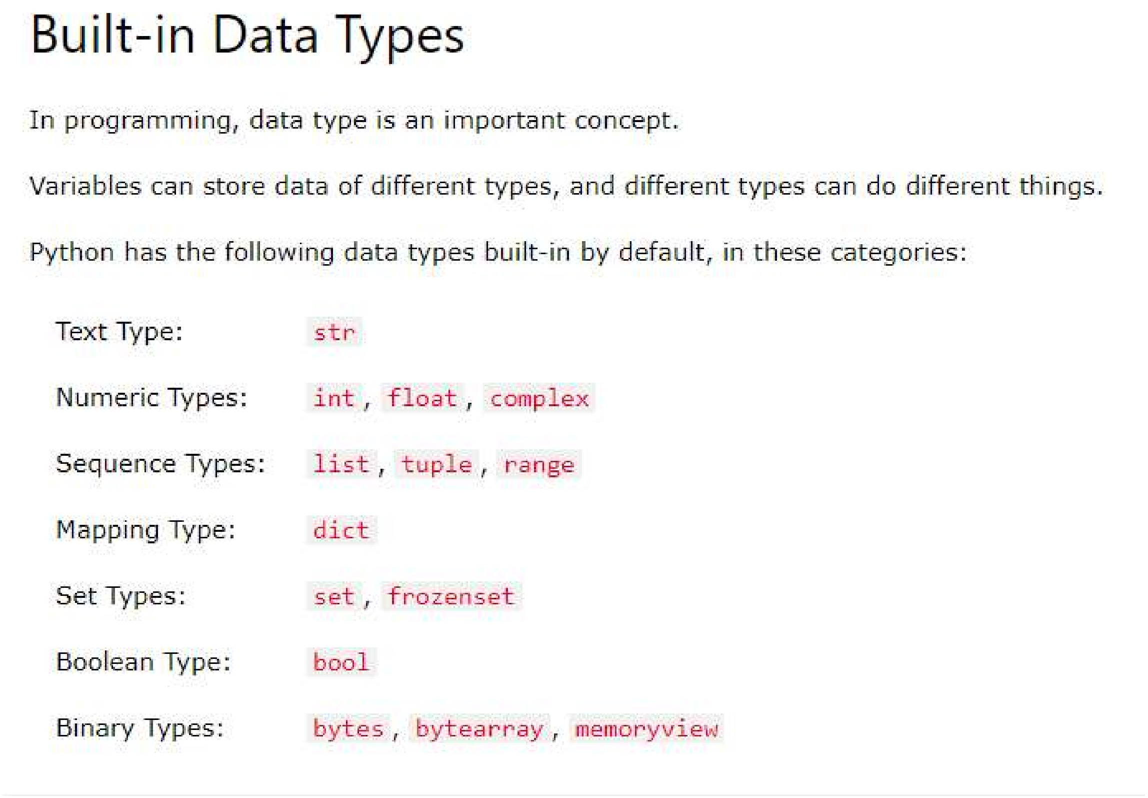
Syntax

- Declaration such as char, int, etc
- No ending of statement
- Structure such as if-else, switch-case, etc
- Functions
- Comment (#)

Reserve words

- Such as If, elif, int, float, etc

Data Type



Built-in Data Types

In programming, data type is an important concept.

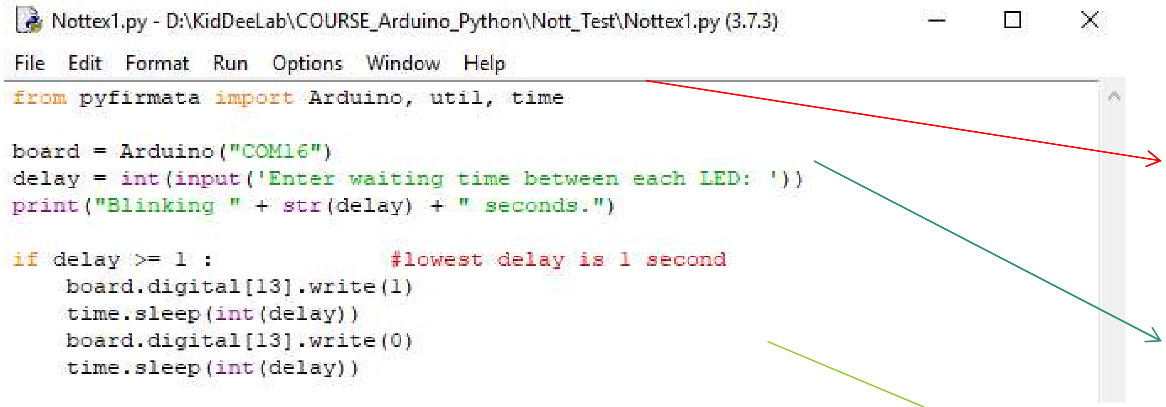
Variables can store data of different types, and different types can do different things.

Python has the following data types built-in by default, in these categories:

Text Type:	`str`
Numeric Types:	`int`, `float`, `complex`
Sequence Types:	`list`, `tuple`, `range`
Mapping Type:	`dict`
Set Types:	`set`, `frozenset`
Boolean Type:	`bool`
Binary Types:	`bytes`, `bytearray`, `memoryview`

- Value (digital, analog)
- Text

Format


Nottex1.py - D:\KidDeeLab\COURSE_Arduino_Python\Nott_Test\Nottex1.py (3.7.3)
File Edit Format Run Options Window Help

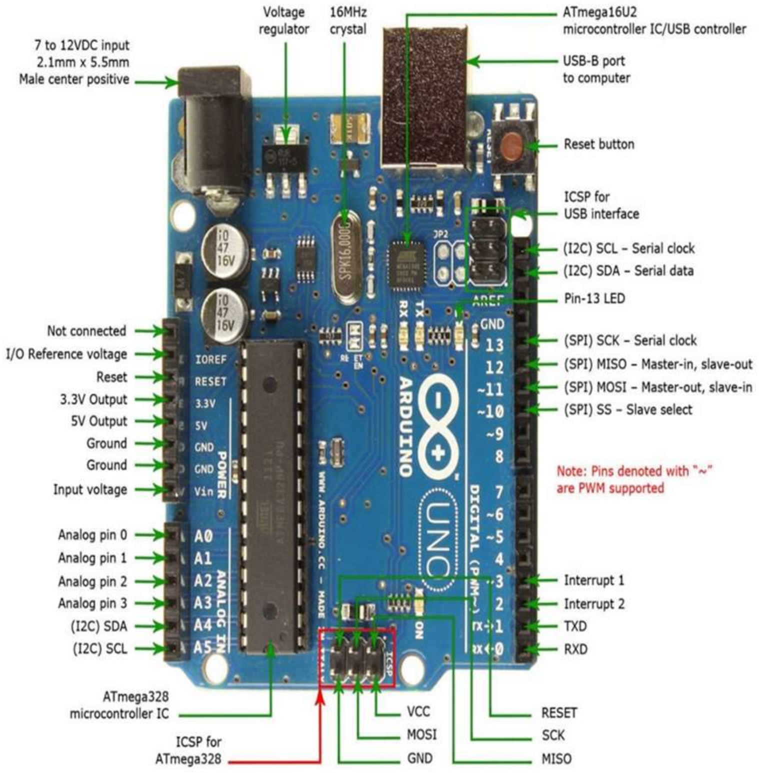
```
from pyfirmata import Arduino, util, time

board = Arduino("COM16")
delay = int(input('Enter waiting time between each LED: '))
print("Blinking " + str(delay) + " seconds.")

if delay >= 1 :                #lowest delay is 1 second
    board.digital[13].write(1)
    time.sleep(int(delay))
    board.digital[13].write(0)
    time.sleep(int(delay))
```

- Import data

- Configure

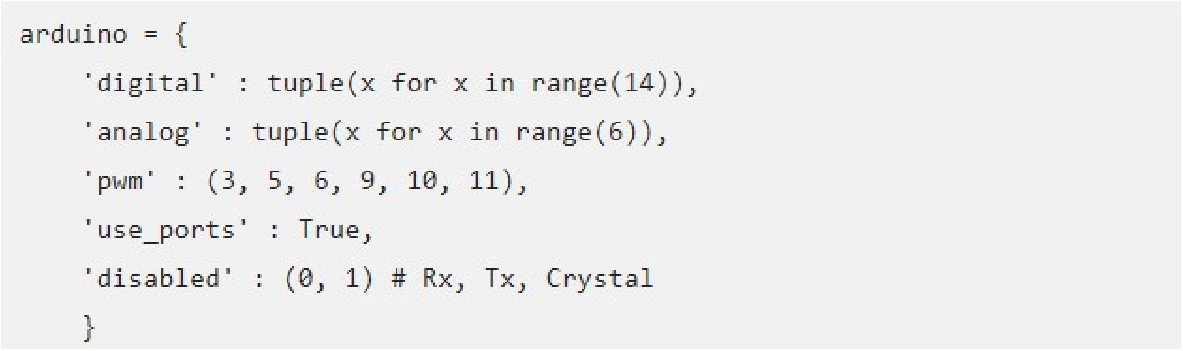
- Code

pyFirmata



Arduino : Arduino UNO

- Any board layout in *pyFirmata* is defined as a dictionary object. The following is a sample of the dictionary object for the Arduino board:



```
arduino = {
    'digital' : tuple(x for x in range(14)),
    'analog' : tuple(x for x in range(6)),
    'pwm' : (3, 5, 6, 9, 10, 11),
    'use_ports' : True,
    'disabled' : (0, 1) # Rx, Tx, Crystal
    }
```

pyfirmata

```
from pyfirmata import Arduino, time
```

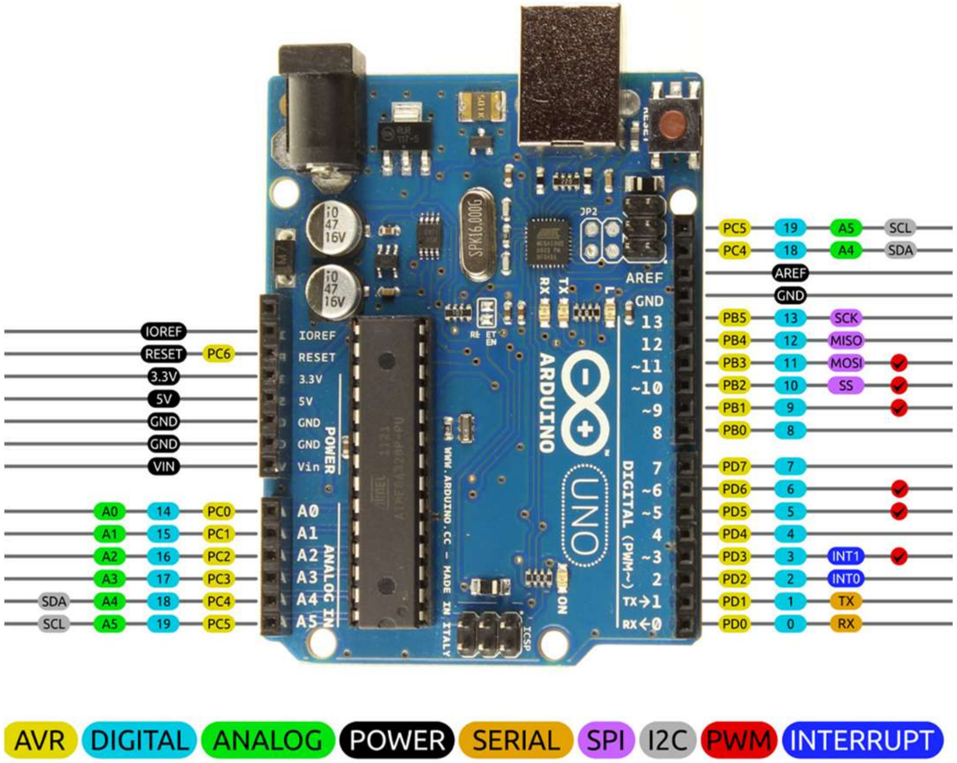
Arduino	: Arduino UNO
time	: sleep time

```
from pyfirmata import OUTPUT
```

OUTPUT	: assigning Pins

```
board = Arduino(“communication port”)
```

Arduino Uno R3 Pinout



Digital Output

Function:

board.digital[Port].mode = OUTPUT

board.digital[Port].write(Action)

- Port : pin number

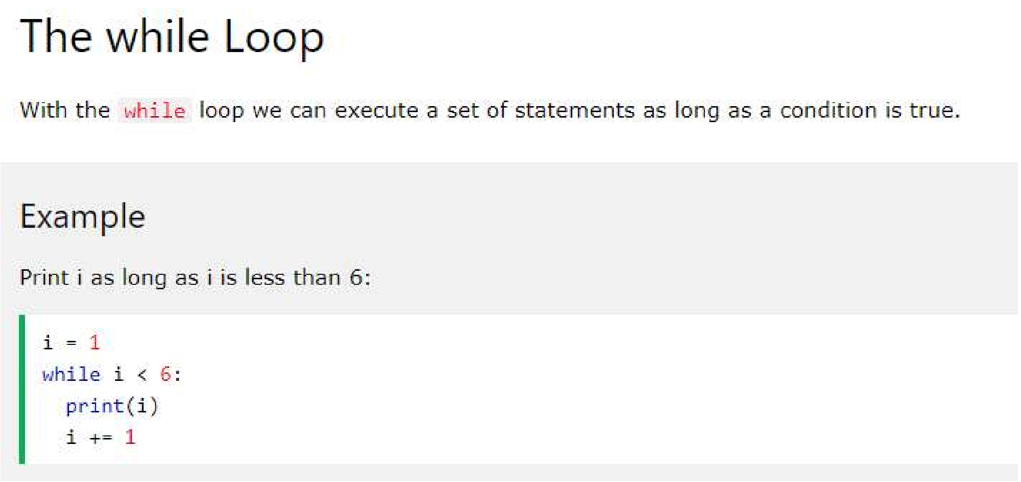
- Action : 0/1

Delay function

Python:

```
Import time
time.sleep(delay in s)
```

while



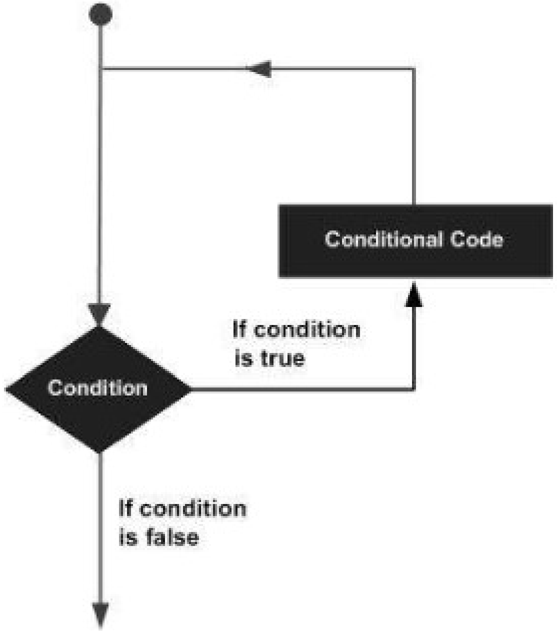
The while Loop

With the `while` loop we can execute a set of statements as long as a condition is true.

Example

Print i as long as i is less than 6:

```
i = 1
while i < 6:
  print(i)
  i += 1
```



Example 10.1

With **Active High** Connection, make LED with Pin12 ON for 1s and OFF for 1s, and repeat for 5 times

- Connect output as active high
- Use Pin 12 as output port
- Write Pin to be ON
- Delay 1s
- Write Pin to be OFF
- Delay 1s
- Repeat for 5 times using “while”

Example 10.2

With **Active one active High and one active Low** Connection, make 2 LEDs on/off with following sequences for 5 times.

- Choose ports 11(AL) and 12(AH) as output