**Lesson\_1\_Notes\_and\_Journal**

**2024.04.18**

* Cleaned up git again – Should be in step at start of lesson.
* Installed Fritzing on HP, ARM not yet supported. (Scroll Down for Notes)
* Syntax Notes
  + Comments

A screenshot of a video game

Description automatically generated

A black and red rectangle with white text

Description automatically generated

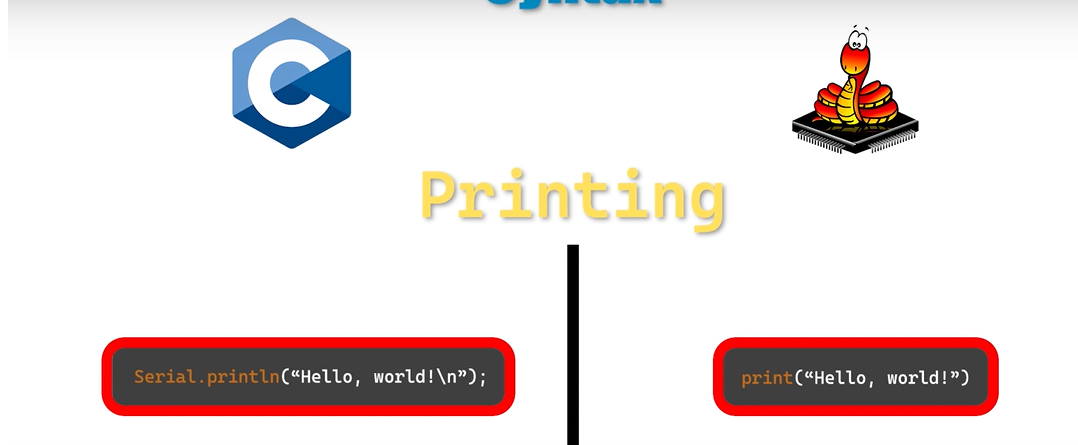
# Single – line comment – starts with #

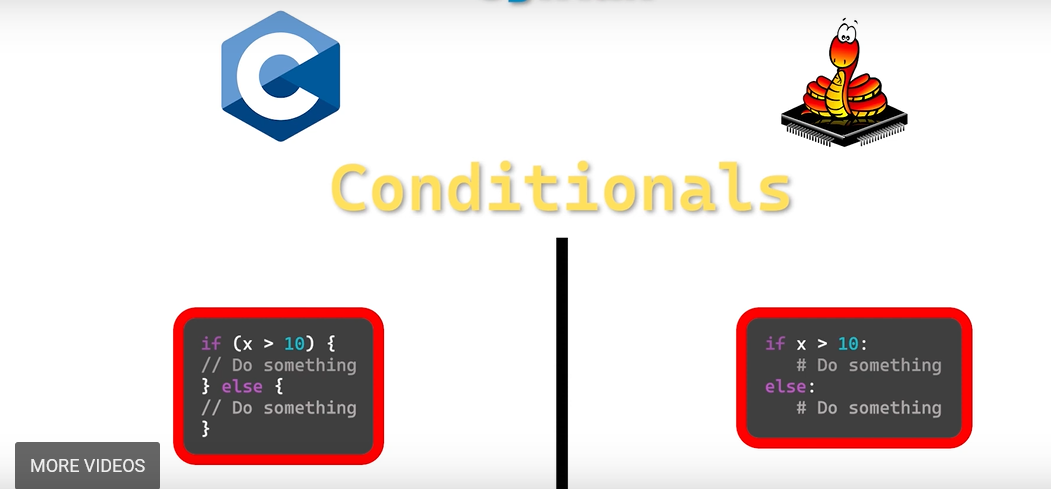
‘’’ Multi – line (Starts and ends with ‘’’)

Comment ‘’’

A screenshot of a computer

Description automatically generated





A screenshot of a computer

Description automatically generated

A screenshot of a computer game

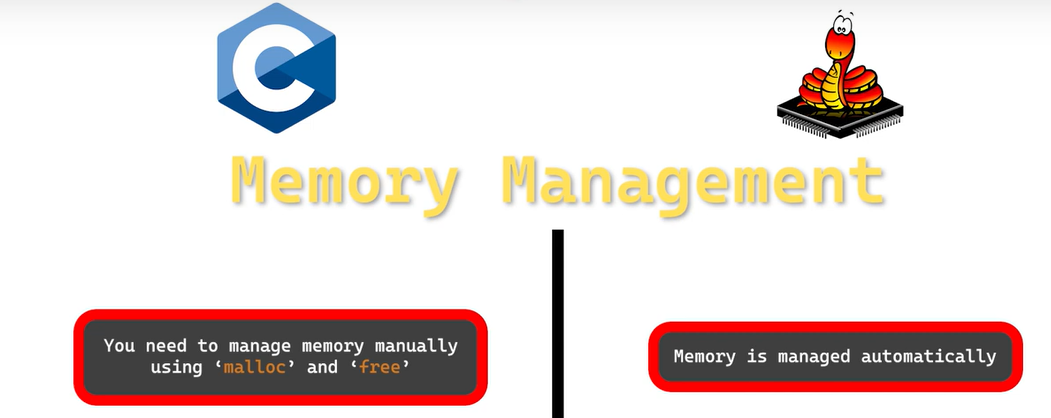
Description automatically generated

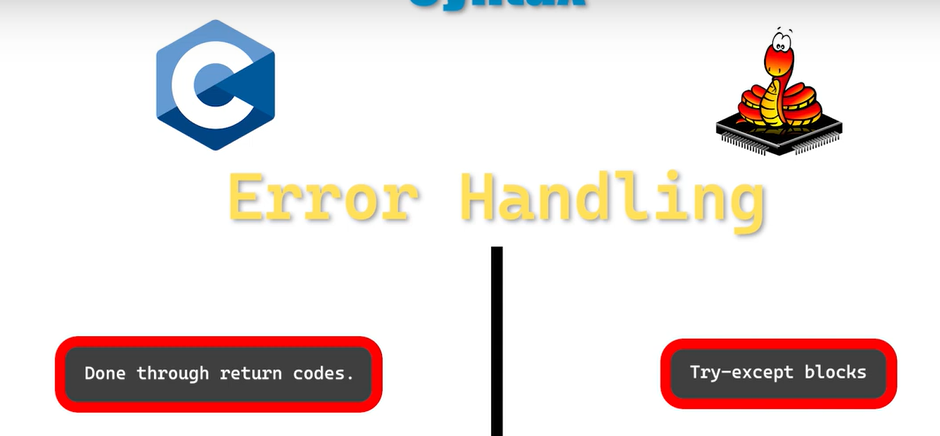
A screenshot of a computer

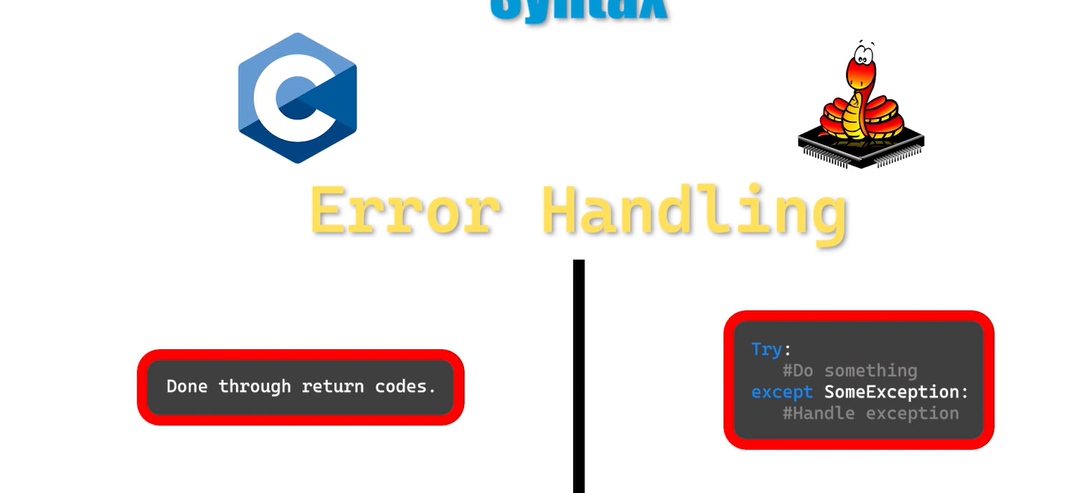
Description automatically generated

A screenshot of a computer

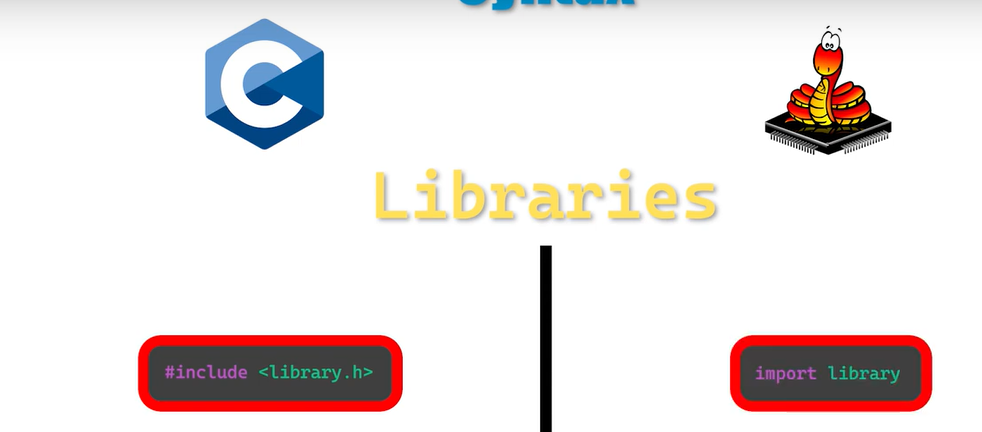
Description automatically generated







Python is cleaner and easier, but C and C++ have much better community and library support, especially for complex situations and parts.



* Compiling
  + C Programs in advance
  + Python / Micro-Python compile as needed (Scripting Language)
* Next 2
* Next 3

**FAIL – No lights**

* Retried Blink and Fast\_Blink – Both Succeeded
* Reversed Red LED – FAIL
* Swapped to 1K Ω on Red LED – Fail

**Next Troubleshooting**

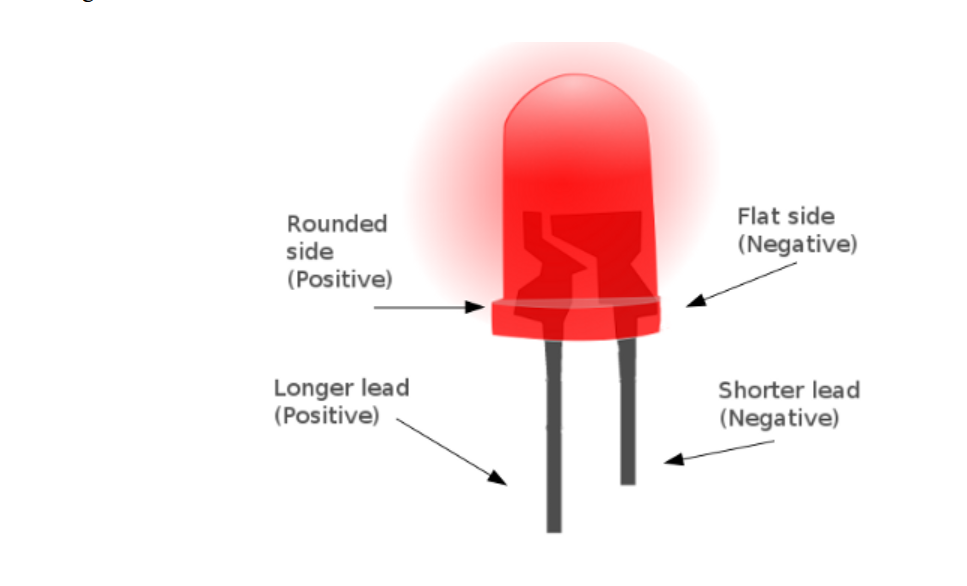
* Confirm Seating or Pi – I very well may not have seated it well.
* Confirm circuit
  + Paths
  + 220??? Resistors
  + Orientation on LEDs
* Get on forum / help

**2024.04.18**

* Confirm Seating or Pi – it needed a firm push
* Confirm circuit – Had to review diodes – Polarity
* Resistor colors are hard to see and match with breadboard
* Had to revise pin names to match the wiring diagram
* Then it ran once
* Had to create a loop
  + Used a loop counter and then went to an increment of zero to make the loop infinite
  + There is probably a better way to do this
* I am not sure if I should be pissed at them or appreciate the troubleshooting / confidence building.
* Practice with Fritzig and possibly KiCad
* Look for better symbols to match up with examples

**2024.04.21**

* I couldn’t find or edit LED to what I wanted, funky As Built LIVES!
* As Built File Created



**Longer lead is positive, bent lead is longer!**

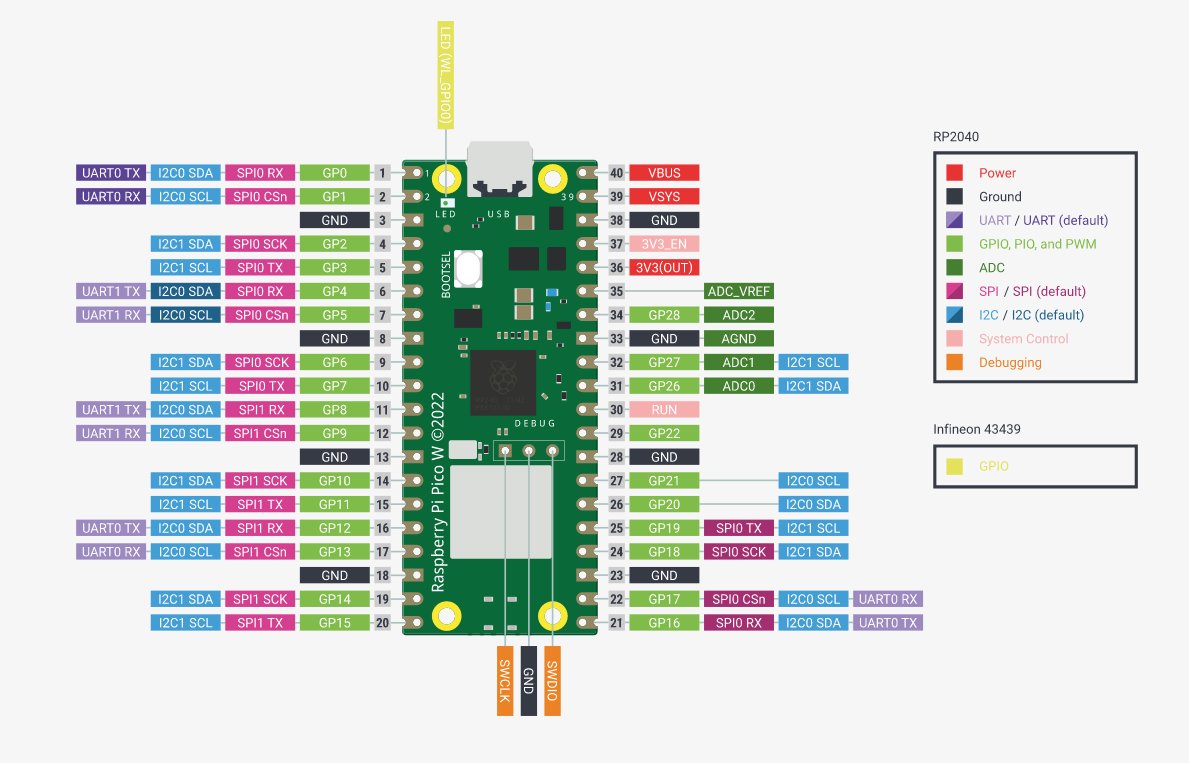
A drawing of a wire

Description automatically generated

A yellow light on a transparent background

Description automatically generated A diagram of anode and cathode

Description automatically generated



A screenshot of a computer program

Description automatically generated

**Installing Fritzing**

**Windows**

We tested Fritzing 1.0.2 on Windows 10 and Windows 11. To install Fritzing, run the downloaded installer file and follow the instructions. It may ask for admin rights ("UAC") during the installation. Please confirm this request by clicking "yes". This will allow the installation of the Visual C++ Redistributable from Microsoft.   
  
For 32 bit systems, as well as for Windows 8.1 and before, Fritzing 0.9.10 is the latest version.

**Mac**

We tested Fritzing 1.0.2 on Mac OS X Big Sur, Monterey and Ventura and Sonoma. To install Fritzing on your Mac, open the downloaded \*.dmg file and move Fritzing to your applications folder. You can then launch Fritzing from the applications folder.   
  
For macOS Catalina, Fritzing 1.0.1 is the latest version. For macOS High Sierra, Fritzing 0.9.10 is the latest version. For macOS Sierra and El Capitan, Fritzing 0.9.3 is the latest version.

**Linux**

We created an AppImage that runs on most Linux systems. We tested it on Ubuntu 20.04 and 22.04. It should run on any system with glibc >= 2.31 (about 2020 and later). ARM based systems are not yet supported.  
  
We recommend storing the AppImage in a Folder like $HOME/bin or $HOME/appimages. To run the AppImage, add the executable permission to the file and start it.  
On Ubuntu 22.04, the libfuse2 library needs to be installed to support AppImages: apt install libfuse2