Requirements:

* Create module for just handling all of ipx communication,
* Needs to open/close ports and handle all potential errors
* Vision is that its as easy as using IPX.get\_uids when coding main script
* Just code to focus on sending the commands and receiving the right responses.
* Need to get the hang of serial communication
* Automate com port detection, don’t leave selection to person using the software

Tasks:

* Mess around with serial communication and properly understand it
* Then finish coding IPX communication module

21/10/2025:

Tasks:

* Add response error checking for everything, thing is the response of things like get\_raw are a string of just numbers, same with list\_uids, so cannot check if the same thing is in there, perhaps with uid, can check with 102, but this is a janky method and will cause error if changed to a different number
* Verify that execute and verify function is working as expected, is currently returning none
* Make sure execute and verify always returns a tuple even if something fails
* Figure out how to manipulate responses in a separate file, then transfer logic to the main file

22/10/25:

* Gemini suggesting to add both a multiline reader, and a single line reader  
  Unsure as not entirely sure how IPXs respond, is it always one line?
* Fully working version of the code without the verification function, works very effectively
* ATM have to use with as for the \_\_enter\_\_ and \_\_exit\_\_ dunder methods, not sure if this is the most effective way of using the code, I would like a version where the end user wouldn’t have to utilise the with as function, and just write say ipx = IPXserialcommunicatr (etc etc). However, gemini thinks this may be fairly complicated to implement, going to leave as is for now

Next steps:

* Explore implementation of with as statements internally:
* Test if the code fully works with dan
* Add logging/.printing so each uid prints as its received
* Structured response from command to command -> higher level, return byte array or string?
* Add error catching for example if a janky loads of bytes are received, catch this, small error catching needs to be added
* Firstly focus on understanding how the ipxs communicate with the terminal across all command
* Additional argument in function for returning different things – higher priority, focus on getting hang of manipulating the data received

Tasks for tomorrow (23/10/2025):

Above task pretty much completed

* Make readme.md
* Add automatic switch for turning ipx on and off -> maybe at higher level

With block calls serials enter and exit methods, which automatically handle the opening and closing of the serial ports

What I wan