U1BP42MRS : please format your code, use backticks on both sides instead of quotes: ` ``` ` <@U3UR8LD18>
U3UR8LD18 : backtics says syntax error
U1BP42MRS : In slack, my friend
U3UR8LD18 : oh...:grimacing:
U3UR8LD18 : thanks
U1BP42MRS : The short of it is this: `os.system` does not capture the stdout from the subprocess - you need to use the `subprocess` module: <https://docs.python.org/3/library/subprocess.html#popen-constructor>
U5NMSURAQ : ```import subprocess

output = subprocess.check_output('df -h | fgrep udev', shell=True)
print(repr(output))
```

U219WLJNN : If I have an unknown # of properties that come from a JSON document that I want to attach to a Python class and then use that new class for every record moving forward, what's the best way to do that?  I was trying to grasp python class factories and was struggling a bit with it
U5NMSURAQ : wow, do you really want exactly that?
U5NMSURAQ : that's crazy!
U5NMSURAQ : ```import json

class A:
    def __init__(self, json_data):
        self.__dict__.update(json.loads(json_data))

text = '{"a": 7, "b": 10, "flag": true}'
a = A(text)
print(vars(a))
```

U3UR8LD18 : thanks
U5NMSURAQ : <@U219WLJNN> If it's crazy but works, it's not crazy :wink:
U5NMSURAQ : (but actually it IS)
U219WLJNN : Well, the properties come from json source 1, and the attributes come from json source 2
U219WLJNN : so I want to just build out my template from source 1 and then loop through the thousands of records in source 2 with the template as my container.
U219WLJNN : the structure of the json is goofy though so i have to load it and process it separately from the standard .loads(). good times.
U5PJK7JHE : so I have a question is there a slack channel for bokeh or flask?
U5PJK7JHE : I have a question on integrating those two
U0LSCQQNR : there is <#C0LN2AD7T|flask>
U5PJK7JHE : is this the place to ask?
U5PJK7JHE : <@U0LSCQQNR> Thank you :slightly_smiling_face:
U2BS4M1RV : I saw a YouTube video where someone used similar but for xml and made it where he could import classes defined in xml, it was pretty cool.
U2BS4M1RV : Sorry, that is to the above with json to class with variables.
U1ENC5QCQ : what the best way to check if the answer is yes and do something only of it yes
U1ENC5QCQ : SyntaxError: unexpected EOF while parsing
U1BP42MRS : <@U1ENC5QCQ> can you give more context? Are you trying to read from user input, or check a value?
U1ENC5QCQ : read from a user
U1BP42MRS : So what do you have now
U1ENC5QCQ : recent = int(raw_input("Provide a number: ") or num_len)
U1ENC5QCQ : trying to learn also shutil.copy
U1BP42MRS : Please be sure to use code formatting
U1ENC5QCQ : oh
U1BP42MRS : Break down your problem into small components, what is the current issue? "Provide a number" suggests to me that you are not looking for "yes"
U1ENC5QCQ : nope
U1ENC5QCQ : I am looking to see if some number was provided, if not use the full legth of the array
U1BP42MRS : "the array" is pretty vague, I think you need to expand on your issue a bit more
U1BP42MRS : <https://stackoverflow.com/help/how-to-ask>

U1ENC5QCQ : let me think about this more on my own
U1ENC5QCQ : thanks <@U1BP42MRS>
U5VCG8JHE : Sounds like you are trying to do an if statement
U1ENC5QCQ : I think I figured a better way
U1ENC5QCQ : brb
U1JBNTG4R : anyone worked with PyMC?
U2BS4M1RV : So, I broke our mysql cluster, how's everyone else's day going?
U2BS4M1RV : Apparently a query caused a table to lock and never unlock.
U1BP42MRS : We had that during a deploy once <@U2BS4M1RV> - someone's console had a lock on a table during a deploy and it got stuck.
U1BP42MRS : You may ask "why does someone have production access to have a console open" - but that's beside the point :joy:
U2BS4M1RV : With the data we are working with, it wouldn't surprise me if we ended up doing that.
U2BS4M1RV : We have weekly cron jobs that truncate then insert hundreds of thousands of records.
U2BS4M1RV : Unfortunately, much of the data we are working with isn't indexed when we get the data.
U1BP42MRS : Dang, that is a bit of data.
U2BS4M1RV : Would be nice to index it, but then those bulk inserts become insert or updates and would likely take a long time.
U1BP42MRS : Was the fix pretty difficult today?
U2BS4M1RV : Don't think he has it fixed yet. Thankfully, not my baby.
U1BP42MRS : Any reason to roll your own test framework? It looks like one big issue is that you don't handle exceptions in the runner, so cleanup won't be guaranteed
U5PJK7JHE : has anyone used pyflot? how is compared to bokeh?
U5WCDJ86B : Can anyone explain why this works?```
with open(path, 'rb') as file_:
    for chunk in iter(lambda: file_.read(4096), b""):
        hash_obj.update(chunk)
```
Specifically the `b""`?

U1BP42MRS : <@U5WCDJ86B> the docs for `iter` explain it pretty well (<https://docs.python.org/3/library/functions.html#iter>).
The second argument is a "sentinel" value, which will create the stop condition when the result of the callable (the lambda function) equals that value

U1BP42MRS : So each iteration it will check the result of the lambda against that value, if they're equal it raises `StopIteration`, a special built in exception to exit a loop.
U4EEBC4SJ : <@U1BP42MRS> I handle exceptions in the real code, they are included in the "do some stuff, execute the actual test" comment. In any event, I thought it made sense to execute the tests this way, and I have a Jenkins job that that does so by executing main.py (each test case being called from within the main file). Really I'm just curious if that goes against any conventions or industry practices because I want to correct any bad habits. I don't really have a reason for rolling my own framework other than to get better at coding (this is my first coding job).
U1BP42MRS : I'd say that, from experience, don't ever roll your own at work. That's free time stuff until you find a deficiency in community accepted frameworks (unittest or pytest, in the python world for the big couple). Every time I have rolled my own it has bitten me hard.
U1BP42MRS : <@U53E0JYLV> - the `illegal byte sequence` is from stderr of the subprocess, one of the pipes is getting something it can't handle. From the try local it's the `cut -c30`.
U1BP42MRS : If you know you don't want whatever is from that, you can just use `stderr=subprocess.PIPE` and it will be captured (and not printed to the python console). Since from the console you will get bytes back, you'll need to decode and probably strip them
U5WCDJ86B : <@U1BP42MRS> sure, but what is `b`? it's not an identifier?
U1BP42MRS : `b"` is a bytes object
U5WCDJ86B : ohh
```