**Challenge 5**

[www.pythonchallenge.com/pc/def/peak.html](http://www.pythonchallenge.com/pc/def/peak.html)

  
***pronounce it***

*(I thought this challenge was significantly harder than the previous ones. I had to cheat a couple of times.)*

The page title is ‘peak hell’. I tried saying it fast and came up with ‘piquel’ which I thought might be a word in another language? Got pickling – so, of course, pickle. This turns out to be a python thing (and now that I look at it – the image looks like a pickle). From python doc:

*The*[*pickle*](https://docs.python.org/3/library/pickle.html#module-pickle)*module implements binary protocols for serializing and de-serializing a Python object structure.*“Pickling”*is the process whereby a Python object hierarchy is converted into a byte stream, and*“unpickling”*is the inverse operation, whereby a byte stream (from a*[*binary file*](https://docs.python.org/3/glossary.html#term-binary-file)*or*[*bytes-like object*](https://docs.python.org/3/glossary.html#term-bytes-like-object)*) is converted back into an object hierarchy. Pickling (and unpickling) is alternatively known as “serialization”, “marshalling,”*[*1*](https://docs.python.org/3/library/pickle.html#id7)*or “flattening”; however, to avoid confusion, the terms used here are “pickling” and “unpickling”.*

What? Tried …pickle.html and go this message: yes! Pickle!

Had to let this one stew awhile, too. Then one of the solution sites pointed out the line <peakhell src="[banner.p](http://www.pythonchallenge.com/pc/def/banner.p)"/> in the source which I hadnt’t noticed. *NOTE TO SELF: always comb the page source for clues.*

Replacing peak.html with banner.p is a web page with weird stuff in it like this: (lp0

(lp1

(S' '

p2

I95

tp3 . . .

Weird stuff -- I’m learning means bytes. Using **requests.get(url).text** gives a string and **.encode(‘utf-8’)** turns that into bytes in some format. The default is ‘utf-8’ and there are a bunch of different formats that translate to different characters. The result is a string with **b** in front of it.

There was a pickle tutorial [Python Pickle Tutorial - DataCamp](https://www.datacamp.com/community/tutorials/pickle-python-tutorial) that I skimmed. The example was to take a dictionary, pickle and unpickle it. So, I played with it some. It takes a dictionary dog\_ages = {‘ozzy’: 3, ‘filou’: 8, ‘luna’: 5} and does pickle.dump to write it to a file using ‘wb’.

When I tried to just do the same with the url text, I just got back what I started with, which still isn’t anything helpful. Then remembered to turn the string into bytes: **requests.get(url).text.encode().** I still couldn’t get this to work but finally just did a plain ‘rb’ write to the file, instead of pickle.dump. Finally, got something different—a sort of dictionary:

[[(' ', 95)], [(' ', 14), ('#', 5), (' ', 70), ('#', 5), (' ', 1)], [(' ', 15), ('#', 4), (' ', 71), ('#', 4), (' ', 1)], [(' ', 15), ('#', 4), (' ', 71), ('#', 4), (' ', 1)], [(' ', 15), ('#', 4), (' ', 71), ('#', 4), (' ', 1)], [(' ', 15), ('#', 4), (' ', 71), ('#', 4), (' ', 1)], [(' ', 15), ('#', 4), (' ', 71), ('#', 4), (' ', 1)], [(' ', 15), ('#', 4), (' ', 71), ('#', 4), (' ', 1)], [(' ', 15), ('#', 4), (' ', 71), ('#', 4), (' ', 1)], [(' ', 6), ('#', 3), (' ', 6), ('#', 4), (' ', 3), ('#', 3), (' ', 9), ('#', 3), (' ', 7),. . .

The ‘aha’ is the file is named banner.p. In the olden days, banners where things like

---XXX--- . . .

--XX-XX—-

-XX---XX-

-XX---XX-

--XX-XX—-

---XXX---

This output looks like a list for each row: 95 blanks, 14 blanks + 5 ‘#’, etc.

Printing it leads to the next challenge.