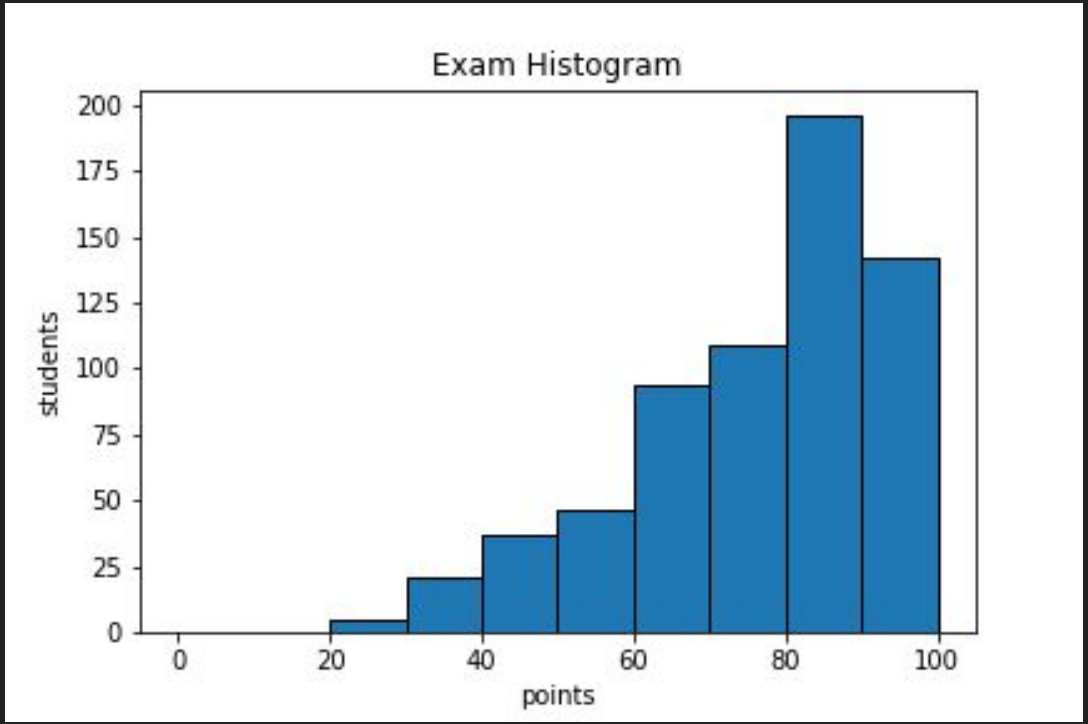
CSE231 - Lab 05

Reading Text Files, try, except

How’d Exam 1 go?

Exam 1 Stats (better than most years!)



File Reading, .txt

Files, when you think about it, are just huge heaps of organized data. They're a way to store massive amounts of information into a neatly organized box that we can move from place to place. If we have massive amounts of data in a file, it gets pretty cumbersome for a human to interpret, which is why we can instead have computers do all the work for us.

For right now, we're going to talk about reading .txt files in Python. Most programming languages have native support for reading .txt files because they're the simplest of all text files -- you can't add headers, footers, have fancy text, emojis, etc.. .txt files can only store characters that are stored within your computer's operating system, AKA the ASCII set.

open()

So how can we open one? Well, Python conveniently comes packaged with the open() function. open() parameters:

open(name[, mode])

name - a string of the name of the file. You only need to supply the name of the file if it’s in the same directory as your Python file (in Spyder), otherwise you have to supply a full path.

mode - a character string that denotes how you want to open the file.

open() Modes

“r” - Read-only mode. Starts at the beginning of the file

“w” - Write-only mode. Overrides the file if it exists, or creates a new text file to write in. Starts at the beginning of the file

“a” - Append mode. Creates file if it does not exist, and starts at the end of the file. Subsequent writes to the file will always append to the end of the file.

“r+” - Read & Write mode. Starts at the beginning of the file

“w+” - Same as “r+”, but overrides file if it exists, or creates a new one if it doesn’t

“a+” - Opens for reading and appending. Same extra properties as “a”.

Reading Lines / try-except / .seek()

The return from open() will give you a *file object.* When you iterate through it, each unpacking of the iteration will give you the corresponding line that was read.

When you iterate through a file object, you would have “read” through the file. When you try to read through more of the file, all you’ll get is the last line. You can reset the position of the “reader” using .seek() on it.

An example of setting the reader back to the beginning of the file, (line 0), would be fileObject.seek(0). You can, of course, skip ahead to a different line by supplying a different number. We’re gonna go through a lot of example code because it’s hard to explain this in a slide.

L5-1/2/3/4/5

Sidenote About Files

This is your first introduction to manipulation of your computer's file system. Which means that working with this stuff can get *dangerous.*

Loops and creating/deleting files is a dangerous combination of concepts. Be careful with the names of the files you're working with.

When you delete a file with a programming language, *Windows/MacOS will* ***not*** *prompt to make sure you truly mean to delete that file*. Additionally, *it will not go to your recycle bin, it will be gone forever.*

It only takes 3 lines of Python to create malware

Lab time