Converting from windows format to unix like. Remove end of line in unix??? Error with shebang

:e ++ff=dos

:w ++ff=unix

:e!

Socket.recv will hold data in a buffer and wait until it is next read. TCP will send the data and the ACKs will go through. The data is waiting to be read.

To reuse a socket:

Create socket

Setsockopt REUSEADDR

Bind

Listen

Accept

All in while loop

Closing a socket stops listening

Ord converts a Unicode number to a character

Chr converts a character to unicode

MEGA COURSE:

-jupyter is for half interpreter and half shell. Good for data analytics

-pandas for data analysis and chart/graphing --- reads jsons, csv, and others

-heroku to deploy web app

-git for change management and uploading web app to cloud

-flask for backend development

-Numpy for number arrays and pandas

-cv2 for image manipulation and video capture

-tkiner for GUI

-sqlite for local DB and postgreSQL for remote server DB

-folium for web maps

-bokeh for graph plotting data analytics

-beautiful soup for web scraping

-pandas data\_reader for scraping stock market data easily

-SQLAlchemy for flask SQL databases

Help(object) #looks up help for object type

Jupyter for half interpreter / half able to run

Glob.glob is like os.listdir but for wildcards

Make python an executable with pyinstaller? Lookup when needed

Cv2 for images

Lambda x: x.<something> --- temp store values in X and execute with value X

Variables string and integers and floats

%d to insert integers into strings

Lists – to organize same object types, can mix strings with integers

Dictionaries – can create items with attributes

Floating point needs to be 3.0 with the ‘.’

Float(3)/2

Int() can convert to integer

\*\* = power 2\*\*3

Abs = absolute value

To convert to str us STR()

%d integer, %f is floats

\n new line

Dictionary

>>> sam["weapon"] = 'chainsaw'

>>> sam["health"] = 10

>>> sam

{'weapon': 'chainsaw', 'health': 10}

>>> sam["weapon"]

'chainsaw'

>>> sam['health']

While loop

while (x<=10):

x+=1

print x

BREAK

while (True):

x+=1

y+=2

if(x+y>10):

break

x = [1,2,7]

>>> for i in x:

print i

FOR

>>> for i in range(30):

print i

for i in range(10,30,2):

print i

for i in range(30):

if not(i%3):

continue

print i

TRY

>>> try:

x=5+'ham'

except:

print 'doh' --- or pass

finally:

print “last error”

dir() – VIEWS ALL FUNCTIONS VARIABLES

>>> def suck():

'''

i documented something

'''

#comment out

Pass

print suck.\_\_doc\_\_

i documented something

import PH

for i in range(10):

PH.printHam()

raw\_input("END")

dict.get returns None which can be used for testing?

Testing if in dictionary….then deleting

**while** True:  
  
 a = raw\_input(**'what would you like to delete? '**)  
  
 **if** a **in** DIC:  
 **del** DIC[a]  
 **break  
 else**:  
 **pass  
  
print** DIC

SOCKET

-server – socket, bind, listen, accept

Sys.argv is how importing argument in python works sys.srgv[0] is the script name. uses the import sys library. Sys.argv[1] is first argument

Timeout can use socket.create\_connection? Or needs to have settimeout option set

Try:

Some code

Except:

Print an error

Else:

Runs if try does not raise exception

Finally:

Run the code even if exception

Can also raise an exception manually through ‘raise’

If \_\_name\_\_ == “\_\_main\_\_”: is used so that if two.py gets imported into one.py, the code in two.py will not execute because name will equal to two.py instead of \_\_main\_\_. \_\_main\_\_ is only set when the code executes on the python script it started from.

If not <var> means if the value is None or false or 0 then it is true.

If <var> means if the value is something or true then it is true.

In a list, it does not have to be list[2] it can be list[<variable of the name/obj>]

PERFECT LOOP

while True:

try:

x=raw\_input('>>')

if x=='5':

break

except:

print'X is not 5'

while True:

x=raw\_input('Type:')

if x=='5':

break

int(raw\_input('Please input the 1st number: '))

text is only updated when in mainloop, not a function or class….

UNICODE UTF8 ISSUE fixed

FILE = io.open('ZZ TEXT', 'a+', encoding='utf8')  
*print* 'write time'  
FILE.write(unicode(result+'\n'))  
FILE.read()

Python -m SimpleHTTPServer

String.replace(old, new, count)

**Default argument** in functions used if needed to input a number that can be changed

Def (x, y=70) y is a default argument that must go after non-default args

String index:

C = string

C[1] would equal t

Emails = [me@gmail, you@gmail.com]

For item in emails:

For ‘gmail’ in item:

Print(‘item’)

File r+ sets pointer to start of file. a+ starts pointer at end of file.

File.seek(0) – place pointer at first line

With statement used in file to auto close file

Pip install <package>

Os.\_\_doc\_\_ - string text doc, bulky

Help(os) – help file for list of methods and information about script

Dir(os) – list of methods

Os.\_\_doc\_\_

Strftime change date time to a string of choosing

Pandas

import pandas

df1=pandas.DataFrame([2,4,6,8],[10,20,30])

df1=pandas.DataFrame([[2,4,6,8],[10,20,30]])

df1

df1=pandas.DataFrame([[2,4,6,8],[10,20,30]],columns=["price','age','value'])

df1=pandas.DataFrame([[2,4,6,8],[10,20,30]],columns=['price','age','value'])

df1=pandas.DataFrame([[2,4,6],[10,20,30]],columns=['price','age','value'])

df1

df1=pandas.DataFrame([[2,4,6],[10,20,30]],columns=['price','age','value'],index=['first','second'])

df1

df1.mean()

df1.price

df1.age

history