Part A: Write a program to connect to port 80 on nigelward.com; request the index.html file, and report the value of bytes 1012 through 1028.

#!/usr/bin/env python3

from socket import \*

#part A

s = socket(AF\_INET, SOCK\_STREAM)

s.connect(("nigelward.com", 80))

s.sendall(("GET /index.html HTTP/1.1\r\n" +

"Host: nigelward.com\r\n" +

"Accept: text/html\r\n" +

"Connection: close\r\n\r\n").encode())

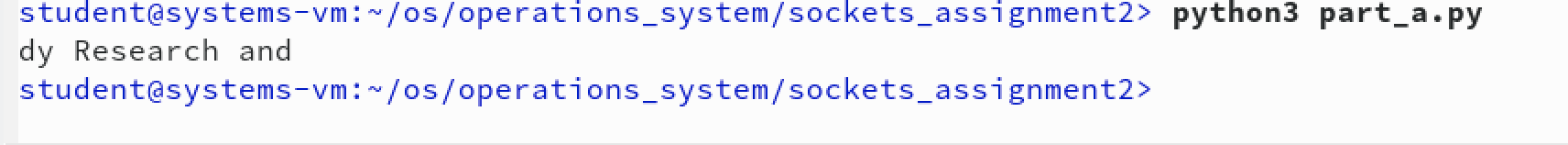
data = s.recv(1012).decode()

data = s.recv(16).decode()

print(data)

s.close()

Output:



Part B: Python is only willing to send bytes via sockets, so if I want

to send the number 256, for example, I have to do something on the

server side, like str(256) + '\n' and then on the client side

something like int(whatWasReceived). Some people prefer to do

(256).to\_bytes(2, byteorder=big) on the server side, and on the

client side do int.from\_bytes(received, byteorder=big).

1. Modify server-demo.py and your client to see if you can really send numbers this way.

Client Server

#!/usr/bin/env python3

from socket import \*

s = socket(AF\_INET, SOCK\_STREAM)

s.connect(("127.0.0.1", 7069))

while 1:

data = s.recv(10000)

data = int.from\_bytes(data, byteorder="big")

if data == 0:

break

print(data)

s.close()

#!/usr/bin/env python3

from socket import \*

s = socket(AF\_INET, SOCK\_STREAM)

s.bind(("127.0.0.1", 7069))

s.listen(5)

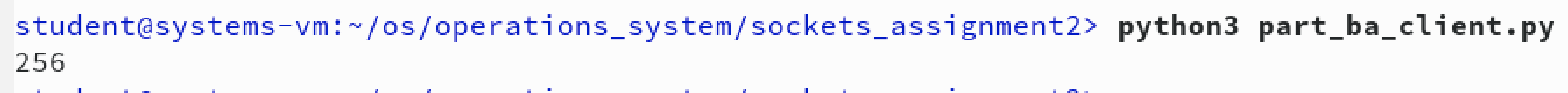
while True:

c,a = s.accept()

data = c.send((256).to\_bytes(2, byteorder = 'big'))

c.close()

Output:



1. Show what happens if one side instead uses byteorder=small.

Client

#!/usr/bin/env python3

from socket import \*

s = socket(AF\_INET, SOCK\_STREAM)

s.connect(("127.0.0.1", 7069))

while 1:

data = s.recv(10000)

data = int.from\_bytes(data, byteorder="big")

if data == 0:

break

print(data)

s.close()

Server

#!/usr/bin/env python3

from socket import \*

s = socket(AF\_INET, SOCK\_STREAM)

s.bind(("127.0.0.1", 7069))

s.listen(5)

while True:

c,a = s.accept()

data = c.send((256).to\_bytes(2, byteorder = 'little'))

#data = c.send((256).to\_bytes(2, byteorder = 'small'))

c.close()

Output:



Client

#!/usr/bin/env python3

from socket import \*

s = socket(AF\_INET, SOCK\_STREAM)

s.connect(("127.0.0.1", 7069))

while 1:

data = s.recv(10000)

data = int.from\_bytes(data, byteorder="little")

if data == 0:

break

print(data)

s.close()

Server

#!/usr/bin/env python3

from socket import \*

s = socket(AF\_INET, SOCK\_STREAM)

s.bind(("127.0.0.1", 7069))

s.listen(5)

while True:

c,a = s.accept()

data = c.send((256).to\_bytes(2, byteorder = 'big'))

c.close()

Output:



1. Read the python sockets page description of htons() and ntons() and experiment with them to determine whether the machine you are using is big-endian or little-endian.

import socket

if socket.ntohs(256) == 1:

print("Little endian because socket.ntohs(256) =", socket.ntohs(256))

else:

print("big endian because socket.ntohs(256) =", socket.ntohs(256))

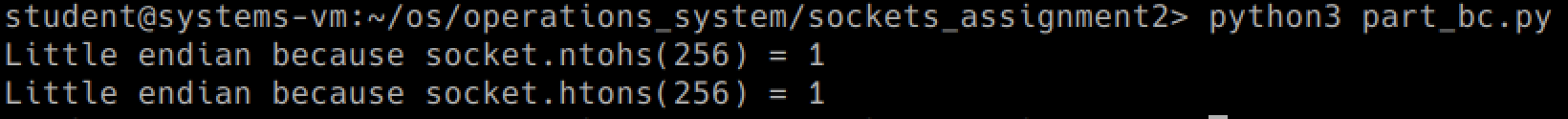
if socket.htons(256) == 1:

print("Little endian because socket.htons(256) =", socket.htons(256))

else:

print("big endian because socket.htons(256) =", socket.htons(256))

Output:



As the output shows on part b->a, on both sides are big and the output is 256 which is the data sent to client side.

In part b -> b, we can see that when one side is different from the other side (eg. little on client & big on server, or big on client & little on server), the output is 1.

As you see on part b-> c, it is a little endian because of the output from ntohs and htons.

Part C: Reimplement your chatbot from Assignment 1 as a client-server

system. The front end (the client) should interact with the user, the

back end (the server) should handle the response logic, and the two

should communicate via sockets, using protocol of your own design.

Client

#!/usr/bin/env python3

from socket import \*

s = socket(AF\_INET, SOCK\_STREAM)

s.connect(("127.0.0.1", 7069))

while 1:

data = s.recv(10000)

print(data.decode())

message = input()

s.send(message.encode())

if len(data) == 0:

break

s.close()

Server

#!/usr/bin/env python3

from socket import \*

s = socket(AF\_INET, SOCK\_STREAM)

s.bind(("127.0.0.1", 7069))

s.listen(5)

c,a = s.accept()

counter = 0

while True:

if counter == 0:

c.send("Hello, welcome to chatbot program.".encode())

c.send("Enter only either \"yes\" or \"no\".".encode())

counter += 1

elif counter == 1:

if data == "yes":

c.send("import file but currently not available.".encode())

c.send("System:\tHello, are you male or female?".encode())

counter += 1

elif data == "no":

c.send("Enter input; but currently not available.".encode())

c.send("System:\tHello, are you male or female?".encode())

counter += 1

else:

c.send("Please enter in the correct format ... \n".encode())

elif counter == 2:

if data == "female":

c.send("How excellent! Are you a CS major?".encode())

elif data == "male":

c.send("Me too. Are you CS major?".encode())

else:

c.send("Great! Anyways, are you CS major?".encode())

counter += 1

elif counter == 3:

if data == "no":

c.send("Too bad. Anyway, what's an animal you like, and two you don't?".encode())

elif data == "yes":

c.send("Excellent, I am too. What's an animal you don't like, and two you don't?".encode())

else:

c.send("Cool! By the way, what's an animal you like, and two you don't?".encode())

counter += 1

elif counter == 4:

data1 = data.split(',')

msg = "%s awesome, but i hate %s too. Bye for now." % (data1[0].strip(), data1[-1].strip())

c.send(msg.encode())

counter += 1

else:

c.send(''.encode())

data = c.recv(1000).decode()

c.close()

Output

Text

Description automatically generated