Name: Robert McKinney

**Assignment 4**

I had to take 2 screenshots for part 2 – my screen resolution simply isn’t high enough to fit all 100 lines of decryption in the Shell without scrolling, which I couldn’t do while taking a screenshot. I split it at line 72 (lines 1-72) and took another screenshot for lines 72-100. I also decreased the font size for part 2’s copy/pasted code below so lines didn’t wrap.

**Part I – python code**

## Robert McKinney

## Week 4 Assignment 4 Part 1

def partOneA():

for i in range(10):

for j in range(10):

print(j, end = " ")

print("\n")

def partOneB():

for i in range(10):

for j in range(10):

print(i, end = " ")

print("\n")

def partTwo():

rangeValue = 0 ## Initialize to 0

for i in range(10):

rangeValue = rangeValue + 1 ## Increment by 1 each loop

for j in range(rangeValue):

print(j, end = " ")

print("\n")

def extraCredit():

rangeValue = 0 ## Initialize to 0

x = 10 ## Initialize to 10 per sample run

for i in range(9): ## Run loop 9 times to end on 54 per sample run

rangeValue = rangeValue + 1 ## Increment by 1 each loop

for j in range(rangeValue):

print(x, end = " ")

x = x + 1 ## Increment by 1 each loop

print("\n")

partOneA()

## Blank line for space between homework parts

print()

partOneB()

## Blank line for space between homework parts

print()

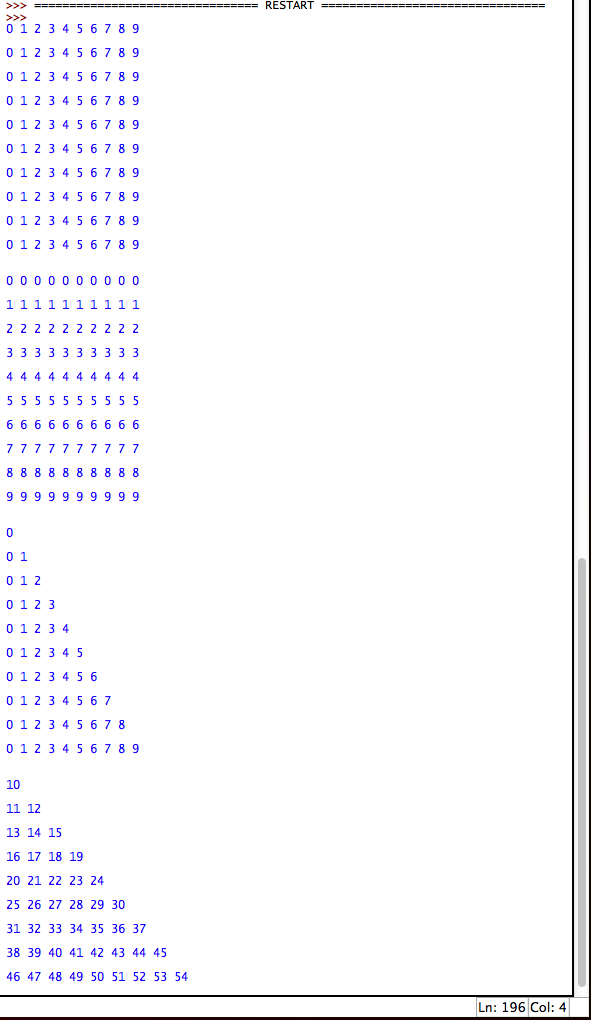
partTwo()

## Blank line for space between homework parts

print()

extraCredit()

**Part I – screen shot of sample run**

****

**Part II – python code**

## Robert McKinney

## Week 4 Assignment 4 Part 2

## ----------

## Encryption

## ----------

def encrypt\_character(original\_char, key):

"""Accept original character and key as parameters, return encrypted character"""

original\_char = ord(original\_char)

if (original\_char + key > 126):

return chr(original\_char + key + 32 - 127)

else:

return chr(original\_char + key)

def encrypt\_string(message, key):

"""Accept string and key as parameter, return encrypted message"""

result = ""

for original\_char in message:

temp\_char = ""

temp\_char = encrypt\_character(original\_char, key)

result = result + temp\_char

return result

def encrypt\_input\_output():

"""Accept regular message and key from user, print encrypted message"""

message = input("Enter a message to encrypt: ")

key = int(input("Enter a key as an integer from 1 to 100: "))

print("The encrypted message is: " + encrypt\_string(message, key))

## ----------

## Decryption

## ----------

def decrypt\_character(encrypted\_char, key):

"""Accept encrypted character and key as parameters, return decrypted character"""

encrypted\_char = ord(encrypted\_char)

if (encrypted\_char - key < 32):

return chr(encrypted\_char - key - 32 + 127)

else:

return chr(encrypted\_char - key)

def decrypt\_string(encrypted\_message, key):

"""Accept encrypted message and key as parameters, return decrypted message"""

result = ""

for encrypted\_char in encrypted\_message:

temp\_char = ""

temp\_char = decrypt\_character(encrypted\_char, key)

result = result + temp\_char

return result

def decrypt\_input\_output():

""" Accept encrypted message from user, print possible decryptions"""

encrypted\_message = input("Enter an encrypted message you would like to decrypt: ")

print("The following are the decrypted messages for keys 1 to 100: ")

## For loop, range 1 through 100, to cycle through all keys

for key in range(1, 101):

decrypted\_message = decrypt\_string(encrypted\_message, key)

print("Key:", key, "-> Decoded Message: ", decrypted\_message)

## -----------

## Run Program

## -----------

encrypt\_input\_output()

decrypt\_input\_output()

**Part II – screen shot of sample run**

