'''

Keni Mou

Kmou1@binghamton.edu

Assignment 9

Lab Section 52

CA Kyle Mille

'''

"""

Analysis

to encrypt or decrypt a cipher in a file

Output to file:

a file contained needed texts

Input from keyboard:

operator(str)

rotationKey(int)

fileName(str)

Tasks allocated to functions:

operationValidated: Check that requested operation is valid

rotationKeyValidated: Check that rotation key is of form <digits> or -<digits> or +<digits>

convertRotationKey: Convert rotation key to value usable for requested operation

keepInBounds: Perform string modulus operation to prevent processed character

processMessage: Encrypt or decrypt message using rotationKey

fileNameValidated: Checks that file exists and that extension is .txt

makeName:Generates output file name from input file name, operation requestedand rotation key

writeToFile: write text into a file

"""

import os.path

#Initialize constants ---------------------------------------------------------

# Mapping of valid operations to rotationKey factor

OPERATIONS = {'e':[1,"Encrypted"], 'd':[-1,"Decrypted"]}

# File processing modes

READ\_MODE = 'r'

WRITE\_MODE = 'w'

# Required file extension

FILE\_EXT = ".txt"

# Min and limit ordinals of printable ASCII

PRINTABLE\_ASCII\_MIN = 32

PRINTABLE\_ASCII\_LIMIT = 127

# Allowable rotation key prefixes

KEY\_PREFIXES = "-+"

# Functions ------------------------------------------------------------------

# Check that requested operation is valid

# param opStr (str) - operation requested

# return True when valid, False otherwise (bool)

def operationValidated(opStr):

return opStr in OPERATIONS

# Checks that file exists and that extension is .txt

# param name (str) - file name

# invoke isFile() from module os.path and endswith()

# return True when valid, False otherwise (bool)

def fileNameValidated(name):

return os.path.isfile(name) and name.endswith(FILE\_EXT)

# Generates output file name from input file name,

# operation requested and rotation key

# param fileName (str) - input file name

# param operation (str)

# param rotationKey (int)

# invoke str.split(), str.replace() and str.join()

# return output file name (str)

def makeName(fileName, operation, rotationKey):

nameList = fileName.split(".")

nameList[0] = nameList[0].replace(OPERATIONS['e'][1], "")

nameList[0] = nameList[0].replace(OPERATIONS['d'][1], "")

nameList[0] += (OPERATIONS[operation][1] + str(rotationKey))

return ".".join(nameList)

# Check that rotation key is of form <digits> or -<digits> or +<digits>

# param rotationKeyStr (str)

# invoke str.isdigit()

# returns: True when valid, False otherwise (bool)

def rotationKeyValidated(rotationKeyStr):

return str.isdigit(rotationKeyStr) or str.isdigit(rotationKeyStr[1:]) and \

rotationKeyStr[0] in KEY\_PREFIXES

# Convert rotation key to value usable for requested operation

# param op (str) - operation requested

# param rotationKeyStr (str)

# invoke int()

# return encryption or decryption rotation key (int)

def convertRotationKey(op, rotationKeyStr):

return OPERATIONS[op][0]\*int(rotationKeyStr)

# Perform string modulus operation to prevent processed character

# from going out of bounds

# param ordinal (int)

# returns adjusted ordinal of new character (int)

def keepInBounds(ordinal):

while ordinal>=PRINTABLE\_ASCII\_LIMIT:

ordinal=ordinal-PRINTABLE\_ASCII\_LIMIT+PRINTABLE\_ASCII\_MIN

while ordinal <PRINTABLE\_ASCII\_MIN:

ordinal=ordinal+PRINTABLE\_ASCII\_LIMIT-PRINTABLE\_ASCII\_MIN

return ordinal

# Encrypt or decrypt message using rotationKey

# param message (str)

# param rotationKey (int)

# invoke keepInBounds()

# return processedMessage (str)

def processMessage(message, rotationKey):

processedMessage=""

for alpha in message:

order=ord(alpha)+rotationKey

while order >= PRINTABLE\_ASCII\_LIMIT or order<PRINTABLE\_ASCII\_MIN:

order=keepInBounds(order)

alpha=chr(order)

processedMessage += alpha

return processedMessage

#write text into a file

#param theFile(file)

#param theText(list)

#reutrn theFile(file)

def writeToFile(theFile,theText):

for i in theText[:-1]:

theFile.write(i[:-1])

theFile.write('\n')

theFile.write(theText[-1]) #to deal with the newline characters

return theFile

# Main -----------------------------------------------------------------------

# Gets plain text or cipher code, operation requested (encrypt or decrypt),

# and rotation key for Caesar cipher

# Generates cipher code or plain text

def main():

# Describe program

print("This program encrypts or decrypts entire files at a time using a Caesar cipher")

#input the file name

fileName=input("Please input your fileName, or press ENTER to quit.")

while fileName:

while not fileNameValidated(fileName):

fileName=input("That file name does not appear to be valid, please try again")

# get the operator

operator=input("Please enter either 'e' to encrypt or 'd' to decrypt")

while not operationValidated(operator):

operator=input("That operation does not appear to be valid, please try again")

#deal with the rotation key

rotationKeyStr=input("Please enter the rotation Key")

while not rotationKeyValidated(rotationKeyStr):

rotationKeyStr=input("Invalid input! Please try agian.")

rotationKey=convertRotationKey(operator, rotationKeyStr)

#The crazy exception handling...

try: # outer try for infile open

inFileObject = open(fileName, READ\_MODE)

try: # inner try for processing infile

contents = inFileObject.readlines()

convertedText=[]

#Convert

for i in contents:

convertedText+=[processMessage(i, rotationKey)]

#Write to the file

try: # "outer" try for outfile open

outFileObject = open(makeName(fileName,operator,rotationKey), WRITE\_MODE)

try: # inner try to outfile processing

#write to the file

writeToFile(outFileObject,convertedText)

except IOError as err: # inner exception handler for outfile processing

print("\nProblem writing data: \n" + str(err))

except ValueError as err: # inner exception handler for outfile processing

print("\nProblem writing data, wrong format or corrupted? \n" + str(err) + '\n')

except Exception as err: # inner exception handler for outfile processing

print("\nData cannot be written to file: \n" + str(err) + '\n')

finally:# will close file whether or not exception has been raised

outFileObject.close()

except IOError as err: # "outer" exception handler for outfile open

print("\nExecption raised during open of output file, no write performed: \n" + str(err) + '\n')

except Exception as err: # outer exception handler for outfile processing

print("\nData cannot be read: \n" + str(err) + '\n')

except IOError as err: # inner exception handler for infile processing

print("\nProblem reading data: \n" + str(err))

except ValueError as err: # inner exception handler for infile processing

print("\nProblem processing data, wrong format or corrupted? \n" + str(err) + '\n')

except Exception as err: # inner exception handler for infile processing

print("\nData cannot be read: \n" + str(err) + '\n')

finally:# will close file whether or not exception has been raised

inFileObject.close()

except FileNotFoundError as err: # outer exception handler for infile open

print("\nFile not found: deleted or in wrong folder? \n" + str(err) + '\n')

except IOError as err: # outer exception handler for infile open

print("\nException raised during open of input file, try a different file: \n" + str(err) + '\n')

except Exception as err: # outer exception handler for infile open

print("\nData cannot be read: \n" + str(err) + '\n')

# Continuation read

fileName=input("Please input your fileName, or press ENTER to quit.")

print("You ended this program")

main()

Encrypt 1

Dbqubjo!Njeojhiu!boe!uif!Tfdsfu!Trvbespo

Boe!opx!cpzt!boe!hjsmt-!ifsf!jt!xibu!zpv!hfu!xifo!zpv!kpjo!uif!Tfdsfu!Trvbespo;!!uijt

nfncfstijq!dbse!xjui!zpvs!pxo!tfdsfu!trvbespo!ovncfs-!uijt!pggjdjbm!Tfdsfu!Trvbespo!nbovbm-!

boe!uijt!dpncjobujpo!cbehf!boe!efdpefs/!!Mppl-!ifsf(t!ipx!uif!efdpefs!xpslt;!!

J(mm!hjwf!zpv!b!uxp!xpse!dmvf-!boe!uif!tfdpoe!xpse!jt!jo!pvs!tfdsfu!dpef/

Uif!gjstu!xpse!jt!#XBUDI#/!Opx-!ifsf!jt!uif!tfdsfu!dpef!xpse!.!xsjuf!epxo!uiftf!ovncfst;!!28.4.31"

Opx-!tfu!zpvs!tfdsfu!efdpefs!mjlf!uijt;!!gps!DPEF!B.4/!!Uifo!efdpef!uijt!jnqpsubou!dmvf!up!ofyu!

xffl(t!bewfouvsf/!!

Jg!zpv!epo(u!ibwf!zpvs!efdpefs!cbehf-!ifsf(t!ipx!zpv!dbo!hfu!pof!gps!zpvs!wfsz!pxo/!!Gjstu-!hfu!b!kbs!

pg!uif!pggjdjbm!tfdsfu!trvbespo!esjol;!!efmjdjpvt!dipdpmbuf.gmbwpsfe!Pwbmujof/!!Uif!gppe!esjol!gps!spdlfu

qpxfs"!!Uifo!dvu!pvu!uif!xby!qbqfs!ejtl!uibu!dpwfst!uif!Pwbmujof!kbs-!boe!tfoe!uibu!ejtl!xjui!zpvs!obnf-!

zpvs!beesftt!up!nf-!Dbqubjo!Njeojhiu!Cpy!Q!Dijdbhp!88-!Jmmjopjt/!!Uibu(t!bmm-!tfoe!op!npofz-!ju(t!gsff!up!

fwfsz!cpz!boe!hjsm!xip!kpjo!uif!tfdsfu!trvbespo/!!Boe!xifo!zpv!sfdfjwf!zpvs!tfdsfu!efdpefs!cbehf-!nfncfstijq

dbse!xjui!zpvs!wfsz!pxo!tfdsfu!trvbespo!ovncfs-!boe!23!qbhf!nbovbm!zpv!xjmm!cf!b!gvmm.gmfehfe!nfncfs/!!Sfnfncfs-

hfu!zpvs!Pwbmujof-!uif!gppe!esjol!gps!spdlfu!qpxfs"!!Ipu!ps!dpme-!ju(t!hpu!xibu!ju!ublft!gps!zpv!up!cf!b!mfbefs!

jo!zpvs!hboh"!!Tp!fwfsz!ebz-!esjol!jotubou!Pwbmujof"

Encrypt -1

B`os`hm~Lhcmhfgs~`mc~sgd~Rdbqds~Rpt`cqnm

@mc~mnv~anxr~`mc~fhqkr+~gdqd~hr~vg`s~xnt~fds~vgdm~xnt~inhm~sgd~Rdbqds~Rpt`cqnm9~~sghr

ldladqrgho~b`qc~vhsg~xntq~nvm~rdbqds~rpt`cqnm~mtladq+~sghr~neehbh`k~Rdbqds~Rpt`cqnm~l`mt`k+~

`mc~sghr~bnlahm`shnm~a`cfd~`mc~cdbncdq-~~Knnj+~gdqd&r~gnv~sgd~cdbncdq~vnqjr9~~

H&kk~fhud~xnt~`~svn~vnqc~bktd+~`mc~sgd~rdbnmc~vnqc~hr~hm~ntq~rdbqds~bncd-

Sgd~ehqrs~vnqc~hr~!V@SBG!-~Mnv+~gdqd~hr~sgd~rdbqds~bncd~vnqc~,~vqhsd~cnvm~sgdrd~mtladqr9~~06,2,1/

Mnv+~rds~xntq~rdbqds~cdbncdq~khjd~sghr9~~enq~BNCD~@,2-~~Sgdm~cdbncd~sghr~hlonqs`ms~bktd~sn~mdws~

vddj&r~`cudmstqd-~~

He~xnt~cnm&s~g`ud~xntq~cdbncdq~a`cfd+~gdqd&r~gnv~xnt~b`m~fds~nmd~enq~xntq~udqx~nvm-~~Ehqrs+~fds~`~i`q~

ne~sgd~neehbh`k~rdbqds~rpt`cqnm~cqhmj9~~cdkhbhntr~bgnbnk`sd,ek`unqdc~Nu`kshmd-~~Sgd~ennc~cqhmj~enq~qnbjds

onvdq ~~Sgdm~bts~nts~sgd~v`w~o`odq~chrj~sg`s~bnudqr~sgd~Nu`kshmd~i`q+~`mc~rdmc~sg`s~chrj~vhsg~xntq~m`ld+~

xntq~`ccqdrr~sn~ld+~B`os`hm~Lhcmhfgs~Anw~O~Bghb`fn~66+~Hkkhmnhr-~~Sg`s&r~`kk+~rdmc~mn~lnmdx+~hs&r~eqdd~sn~

dudqx~anx~`mc~fhqk~vgn~inhm~sgd~rdbqds~rpt`cqnm-~~@mc~vgdm~xnt~qdbdhud~xntq~rdbqds~cdbncdq~a`cfd+~ldladqrgho

b`qc~vhsg~xntq~udqx~nvm~rdbqds~rpt`cqnm~mtladq+~`mc~01~o`fd~l`mt`k~xnt~vhkk~ad~`~etkk,ekdcfdc~ldladq-~~Qdldladq+

fds~xntq~Nu`kshmd+~sgd~ennc~cqhmj~enq~qnbjds~onvdq ~~Gns~nq~bnkc+~hs&r~fns~vg`s~hs~s`jdr~enq~xnt~sn~ad~`~kd`cdq~

hm~xntq~f`mf ~~Rn~dudqx~c`x+~cqhmj~hmrs`ms~Nu`kshmd

Encrypt 127

c"15"\*/@m\*%/\*()5@"/%@5)&@s&$3&5@s26"%30/

a/%@/08@#0:4@"/%@(\*3-4L@)&3&@\*4@8)"5@:06@(&5@8)&/@:06@+0\*/@5)&@s&$3&5@s26"%30/Z@@5)\*4

.&.#&34)\*1@$"3%@8\*5)@:063@08/@4&$3&5@426"%30/@/6.#&3L@5)\*4@0''\*$\*"-@s&$3&5@s26"%30/@."/6"-L@

"/%@5)\*4@$0.#\*/"5\*0/@#"%(&@"/%@%&$0%&3N@@l00,L@)&3&G4@)08@5)&@%&$0%&3@803,4Z@@

iG--@(\*7&@:06@"@580@803%@$-6&L@"/%@5)&@4&$0/%@803%@\*4@\*/@063@4&$3&5@$0%&N

t)&@'\*345@803%@\*4@BwatchBN@n08L@)&3&@\*4@5)&@4&$3&5@$0%&@803%@M@83\*5&@%08/@5)&4&@/6.#&34Z@@QWMSMRPA

n08L@4&5@:063@4&$3&5@%&$0%&3@-\*,&@5)\*4Z@@'03@code@aMSN@@t)&/@%&$0%&@5)\*4@\*.1035"/5@$-6&@50@/&95@

8&&,G4@"%7&/563&N@@

i'@:06@%0/G5@)"7&@:063@%&$0%&3@#"%(&L@)&3&G4@)08@:06@$"/@(&5@0/&@'03@:063@7&3:@08/N@@f\*345L@(&5@"@+"3@

0'@5)&@0''\*$\*"-@4&$3&5@426"%30/@%3\*/,Z@@%&-\*$\*064@$)0$0-"5&M'-"703&%@o7"-5\*/&N@@t)&@'00%@%3\*/,@'03@30$,&5

108&3A@@t)&/@$65@065@5)&@8"9@1"1&3@%\*4,@5)"5@$07&34@5)&@o7"-5\*/&@+"3L@"/%@4&/%@5)"5@%\*4,@8\*5)@:063@/".&L@

:063@"%%3&44@50@.&L@c"15"\*/@m\*%/\*()5@b09@p@c)\*$"(0@WWL@i--\*/0\*4N@@t)"5G4@"--L@4&/%@/0@.0/&:L@\*5G4@'3&&@50@

&7&3:@#0:@"/%@(\*3-@8)0@+0\*/@5)&@4&$3&5@426"%30/N@@a/%@8)&/@:06@3&$&\*7&@:063@4&$3&5@%&$0%&3@#"%(&L@.&.#&34)\*1

$"3%@8\*5)@:063@7&3:@08/@4&$3&5@426"%30/@/6.#&3L@"/%@QR@1"(&@."/6"-@:06@8\*--@#&@"@'6--M'-&%(&%@.&.#&3N@@r&.&.#&3L

(&5@:063@o7"-5\*/&L@5)&@'00%@%3\*/,@'03@30$,&5@108&3A@@h05@03@$0-%L@\*5G4@(05@8)"5@\*5@5",&4@'03@:06@50@#&@"@-&"%&3@

\*/@:063@("/(A@@s0@&7&3:@%":L@%3\*/,@\*/45"/5@o7"-5\*/&A

Encrypt -127

#APTAIN\_-IDNIGHT\_AND\_THE\_3ECRET\_3QUADRON

!ND\_NOW\_BOYS\_AND\_GIRLSk\_HERE\_IS\_WHAT\_YOU\_GET\_WHEN\_YOU\_JOIN\_THE\_3ECRET\_3QUADRONy\_\_THIS

MEMBERSHIP\_CARD\_WITH\_YOUR\_OWN\_SECRET\_SQUADRON\_NUMBERk\_THIS\_OFFICIAL\_3ECRET\_3QUADRON\_MANUALk\_

AND\_THIS\_COMBINATION\_BADGE\_AND\_DECODERm\_\_,OOKk\_HEREfS\_HOW\_THE\_DECODER\_WORKSy\_\_

)fLL\_GIVE\_YOU\_A\_TWO\_WORD\_CLUEk\_AND\_THE\_SECOND\_WORD\_IS\_IN\_OUR\_SECRET\_CODEm

4HE\_FIRST\_WORD\_IS\_a7!4#(am\_.OWk\_HERE\_IS\_THE\_SECRET\_CODE\_WORD\_l\_WRITE\_DOWN\_THESE\_NUMBERSy\_\_pvlrlqo`

.OWk\_SET\_YOUR\_SECRET\_DECODER\_LIKE\_THISy\_\_FOR\_#/$%\_!lrm\_\_4HEN\_DECODE\_THIS\_IMPORTANT\_CLUE\_TO\_NEXT\_

WEEKfS\_ADVENTUREm\_\_

)F\_YOU\_DONfT\_HAVE\_YOUR\_DECODER\_BADGEk\_HEREfS\_HOW\_YOU\_CAN\_GET\_ONE\_FOR\_YOUR\_VERY\_OWNm\_\_&IRSTk\_GET\_A\_JAR\_

OF\_THE\_OFFICIAL\_SECRET\_SQUADRON\_DRINKy\_\_DELICIOUS\_CHOCOLATElFLAVORED\_/VALTINEm\_\_4HE\_FOOD\_DRINK\_FOR\_ROCKET

POWER`\_\_4HEN\_CUT\_OUT\_THE\_WAX\_PAPER\_DISK\_THAT\_COVERS\_THE\_/VALTINE\_JARk\_AND\_SEND\_THAT\_DISK\_WITH\_YOUR\_NAMEk\_

YOUR\_ADDRESS\_TO\_MEk\_#APTAIN\_-IDNIGHT\_"OX\_0\_#HICAGO\_vvk\_)LLINOISm\_\_4HATfS\_ALLk\_SEND\_NO\_MONEYk\_ITfS\_FREE\_TO\_

EVERY\_BOY\_AND\_GIRL\_WHO\_JOIN\_THE\_SECRET\_SQUADRONm\_\_!ND\_WHEN\_YOU\_RECEIVE\_YOUR\_SECRET\_DECODER\_BADGEk\_MEMBERSHIP

CARD\_WITH\_YOUR\_VERY\_OWN\_SECRET\_SQUADRON\_NUMBERk\_AND\_pq\_PAGE\_MANUAL\_YOU\_WILL\_BE\_A\_FULLlFLEDGED\_MEMBERm\_\_2EMEMBERk

GET\_YOUR\_/VALTINEk\_THE\_FOOD\_DRINK\_FOR\_ROCKET\_POWER`\_\_(OT\_OR\_COLDk\_ITfS\_GOT\_WHAT\_IT\_TAKES\_FOR\_YOU\_TO\_BE\_A\_LEADER\_

IN\_YOUR\_GANG`\_\_3O\_EVERY\_DAYk\_DRINK\_INSTANT\_/VALTINE`

Encrypt 570 and -570

Captain Midnight and the Secret Squadron

And now boys and girls, here is what you get when you join the Secret Squadron: this

membership card with your own secret squadron number, this official Secret Squadron manual,

and this combination badge and decoder. Look, here's how the decoder works:

I'll give you a two word clue, and the second word is in our secret code.

The first word is "WATCH". Now, here is the secret code word - write down these numbers: 17-3-20!

Now, set your secret decoder like this: for CODE A-3. Then decode this important clue to next

week's adventure.

If you don't have your decoder badge, here's how you can get one for your very own. First, get a jar

of the official secret squadron drink: delicious chocolate-flavored Ovaltine. The food drink for rocket

power! Then cut out the wax paper disk that covers the Ovaltine jar, and send that disk with your name,

your address to me, Captain Midnight Box P Chicago 77, Illinois. That's all, send no money, it's free to

every boy and girl who join the secret squadron. And when you receive your secret decoder badge, membership

card with your very own secret squadron number, and 12 page manual you will be a full-fledged member. Remember,

get your Ovaltine, the food drink for rocket power! Hot or cold, it's got what it takes for you to be a leader

in your gang! So every day, drink instant Ovaltine!