ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΚΡΥΠΤΟΓΡΑΦΙΑ

9ο ΕΞΑΜΗΝΟ

1η Σειρά Ασκήσεων

Ονοματεπώνυμο (Αριθμός Μητρώου):

Γεώργιος Γκοτζιάς (03119047)

7 Νοεμβρίου 2023

Άσκηση 1

Περιγραφή τρόπου επίλυσης

Αρχικά, μετράμε τη συχνότητα εμφάνισης κάθε γράμματος στο κρυπτοκείμενο. Έπειτα, αντικαθιστούμε κάθε γράμμα του κρυπτοκειμένου με το αντίστοιχο σε συχνότητα εμφάνισης, δηλαδή το πιο συχνό γράμμα του κειμένου αντικαθίσταται από το πιο συχνά εμφανιζόμενο γράμμα στην αγγλική γλώσσα, κ.ο.κ. Αυτό είναι το αρχικό κλειδί που υποθέτουμε για αποκρυπτογράφηση.

Το κείμενο που παίρνουμε εξακολουθεί να απέχει από κάποιο κείμενο στα αγγλικά, αλλά είναι πιο πιθανό κάποια γράμματα να έχουν πράγματι αντικατασταθεί σωστά. Προσπαθούμε, τώρα να γίνουν αντικαταστάσεις στο κλειδί που χρησιμοποιείται για την αποκρυπτογράφηση, ώστε να καταλήξουμε στο αρχικό κείμενο. Παρατηρούμε ότι στην αρχή εμφανίζεται η 'λέξη' RCE, η οποία δεν είναι πραγματική λέξη της αγγλικής, όμως είναι πολύ πιθανό το κείμενο αρχίζει με την λέξη THE, οπότε το αρχικό κλειδί τροποποιείται, ώστε το γράμμα που αντιστοιχίζοταν στο R να αντιστοιχίζεται στο T και αντιστρόφως, και αντίστοιχα για τα C,H.

Εξακολουθεί το κείμενο να απέχει από κανονικό κείμενο, όμως παρατηρούμε ότι εμφανίζονται συχνά οι 'λέξεις' ALE, HAFE, που είναι πολύ πιθανό να αντιστοιχούσαν στο αρχικό κείμενο στις λέξεις ARE, HAVE, οπότε κάνουμε τις κατάλληλες αλλαγές.

Ακόμη, παρατηρούμε τις λέξεις CACER, THEORW, οι οποίες πιθανότατα προέρχονται από τις λέξεις PAPER, THEORY στο αρχικό κείμενο.

Επαναλαμβάνουμε την ίδια διαδικασία βρίσκοντας λέξεις που μοιάζουν με γνωστές αγγλικές λέξεις και σε κάθε βήμα το κείμενο που προκύπτει αρχίζει να μοιάζει περισσότερο με κανονικό κείμενο στην αγγλική γλώσσα, οπότε και η εύρεση των λέξεων γίνεται ευκολότερη. Οι υπόλοιπες αλλαγές φαίνονται στον κώδικα.

Το κλειδί που χρησιμοποιήθηκε για την αποκρυπτογράφη είναι το ακόλουθο, δηλαδή το dictionary subs όπως προκύπτει από τον κώδικα:

{'S': 'K', 'E': 'J', 'M': 'Z', 'F': 'Q', 'T': 'X', 'Z': 'W', 'O': 'Y', 'J': 'G', 'X': 'V', 'Y': 'D', 'P': 'F', 'I': 'P', 'A': 'B', 'B': 'M', 'N': 'U', 'H': 'C', 'V': 'H', 'R': 'R', 'L': 'L', 'G': 'S', 'C': 'I', 'Q': 'O', 'D': 'N', 'K': 'T', 'W': 'A', 'U': 'E'}

Επομένως, το κλειδί που χρησιμοποιήθηκε για την κρυπτογράφηση είναι το αντίστροφο, δηλαδή το 'K' αντιστοιχίζεται στο 'S', το 'J' στο 'E' και ούτω καθεξής.

Έτσι, καταλήγουμε ότι το plain text είναι το εξής:

THE COMPUTABLE NUMBERS MAY BE DESCRIBED BRIEFLY AS THE REAL NUMBERS WHOSE EXPRESSIONS AS A DECIMAL ARE CALCULABLE BY FINITE MEANS. ALTHOUGH THE SUBJECT OF THIS PAPER IS OSTENSIBLY THE COMPUTABLE NUMBERS. IT IS ALMOST EQUALLY EASY TO DEFINE AND INVESTIGATE COMPUTABLE FUNCTIONS OF AN INTEGRAL VARIABLE OR A REAL OR COMPUTABLE VARIABLE, COMPUTABLE PREDICATES, AND SO FORTH. THE FUNDAMENTAL PROBLEMS INVOLVED ARE, HOWEVER, THE SAME IN EACH CASE, AND I HAVE CHOSEN THE COMPUTABLE NUMBERS FOR EXPLICIT TREATMENT AS INVOLVING THE LEAST CUMBROUS TECHNIQUE. I HOPE SHORTLY TO GIVE AN ACCOUNT OF THE RELATIONS OF THE COMPUTABLE NUMBERS, FUNCTIONS, AND SO FORTH TO ONE ANOTHER. THIS WILL INCLUDE A DEVELOPMENT OF THE THEORY OF FUNCTIONS OF A REAL VARIABLE EXPRESSED IN TERMS OF COMPUTABLE NUMBERS. ACCORDING TO MY DEFINITION, A NUMBER IS COMPUTABLE IF ITS DECIMAL CAN BE WRITTEN DOWN BY A MACHINE. I GIVE SOME ARGUMENTS WITH THE INTENTION OF SHOWING THAT THE COMPUTABLE NUMBERS INCLUDE ALL NUMBERS WHICH COULD NATURALLY BE REGARDED AS COMPUTABLE. IN PARTICULAR, I SHOW THAT CERTAIN LARGE CLASSES OF NUMBERS ARE COMPUTABLE. THEY INCLUDE, FOR INSTANCE, THE REAL PARTS OF ALL ALGEBRAIC NUMBERS, THE REAL PARTS OF THE ZEROS OF THE BESSEL FUNCTIONS, THE NUMBERS PI, E, ETC. THE COMPUTABLE NUMBERS DO NOT, HOWEVER, INCLUDE ALL DEFINABLE NUMBERS, AND AN EXAMPLE IS GIVEN OF A DEFINABLE NUMBER WHICH IS NOT COMPUTABLE. ALTHOUGH THE CLASS OF COMPUTABLE NUMBERS IS SO GREAT, AND IN MANY WAYS SIMILAR TO THE CLASS OF REAL NUMBERS, IT IS NEVERTHELESS ENUMERABLE. I EXAMINE CERTAIN ARGUMENTS WHICH WOULD SEEM TO PROVE THE CONTRARY. BY THE CORRECT APPLICATION OF ONE OF THESE ARGUMENTS, CONCLUSIONS ARE REACHED WHICH ARE SUPERFICIALLY SIMILAR TO THOSE OF GODEL. THESE RESULTS HAVE VALUABLE APPLICATIONS. IN PARTICULAR, IT IS SHOWN THAT THE HILBERTIAN ENTSCHEIDUNGSPROBLEM CAN HAVE NO SOLUTION.

Κώδικας

```
def swap(dict, c1, c2):
    for key, value in dict.items():
        if value == c1:
```

```python
            k1 = key
            break

    for key, value in dict.items():
        if value == c2:
            k2 = key
            break

    dict[k2] = c1
    dict[k1] = c2


text = """
KVU HQBINKWALU DNBAURG BWO AU YUGHRCAUY ARCUPLO WG KVU RUWL DNBAURG ZVQGU
UTIRUGGCQDG WG W YUHCBWL WRU HWLHNLWALU AO PCDCKU BUWDG. WLKVQNJV KVU GNAEUHK
QP KVCG IWIUR CG QGKUDGCALO KVU HQBINKWALU DNBAURG. CK CG WLBQGK UFNWLLO
UWGO KQ YUPCDU WDY CDXUGKCJWKU HQBINKWALU PNDHKCQDG QP WD CDKUJRWL XWRCWALU
QR W RUWL QR HQBINKWALU XWRCWALU, HQBINKWALU IRUYCHWKUG, WDY GQ PQRKV. KVU
PNDYWBUDKWL IRQALUBG CDXQLXUY WRU, VQZUXUR, KVU GWBU CD UWHV HWGU, WDY C VWXU
HVQGUD KVU HQBINKWALU DNBAURG PQR UTILCHCK KRUWKBUDK WG CDXQLXCDJ KVU LUWGK
HNBARQNG KUHVDCFNU. C VQIU GVQRKLO KQ JCXU WD WHHQNDK QP KVU RULWKCQDG QP KVU
HQBINKWALU DNBAURG, PNDHKCQDG, WDY GQ PQRKV KQ QDU WDQKVUR. KVCG ZCLL CDHLNYU W
YUXULQIBUDK QP KVU KVUQRO QP PNDHKCQDG QP W RUWL XWRCWALU UTIRUGGUY CD KURBG
QP HQBINKWALU DNBAURG. WHHQRYCDJ KQ BO YUPCDCKCQD, W DNBAUR CG HQBINKWALU
CP CKG YUHCBWL HWD AU ZRCKKUD YQZD AO W BWHVCDU. C JCXU GQBU WRJNBUDKG ZCKV
KVU CDKUDKCQD QP GVQZCDJ KVWK KVU HQBINKWALU DNBAURG CDHLNYU WLL DNBAURG
ZVCHV HQNLY DWKNRWLLO AU RUJWRYUY WG HQBINKWALU. CD IWRKCHNLWR, C GVQZ KVWK
HURKWCD LWRJU HLWGGUG QP DNBAURG WRU HQBINKWALU. KVUO CDHLNYU, PQR CDGKWDHU,
KVU RUWL IWRKG QP WLL WLJUARWCH DNBAURG, KVU RUWL IWRKG QP KVU MURQG QP KVU
AUGGUL PNDHKCQDG, KVU DNBAURG IC, U, UKH. KVU HQBINKWALU DNBAURG YQ DQK,
VQZUXUR, CDHLNYU WLL YUPCDWALU DNBAURG, WDY WD UTWBILU CG JCXUD QP W YUPCDWALU
DNBAUR ZVCHV CG DQK HQBINKWALU. WLKVQNJV KVU HLWGG QP HQBINKWALU DNBAURG
CG GQ JRUWK, WDY CD BWDO ZWOG GCBCLWR KQ KVU HLWGG QP RUWL DNBAURG, CK CG
DUXURKVULUGG UDNBURWALU. C UTWBCDU HURKWCD WRJNBUDKG ZVCHV ZQNLY GUUB KQ IRQXU
KVU HQDKRWRO. AO KVU HQRRUHK WIILCHWKCQD QP QDU QP KVUGU WRJNBUDKG, HQDHLNGCQDG
WRU RUWHVUY ZVCHV WRU GNIURPCHCWLLO GCBCLWR KQ KVQGU QP JQYUL. KVUGU RUGNLKG
VWXU XWLNWALU WIILCHWKCQDG. CD IWRKCHNLWR, CK CG GVQZD KVWK KVU VCLAURKCWD
UDKGHVUCYNDJGIRQALUB HWD VWXU DQ GQLNKCQD.
"""

freq = {}
for i in "QWERTYUIOPASDFGHJKLZXCVBNM":
    freq[i] = 0

for c in text:
    if c == '\n' or c == ' ' or c == '.' or c ==',':
        continue
    freq[c] += 1

eng_freq = {
    'E' : 0.111607,
    'A' : 0.084966,
    'R' : 0.075809,
    'I' : 0.075448,
    'O' : 0.071635,
    'T' : 0.069509,
    'N' : 0.066544,
    'S' : 0.057531,
    'L' : 0.054893,
    'C' : 0.045338,
    'U' : 0.036308,
    'D' : 0.033844,
    'P' : 0.031671,
    'M' : 0.030129,
    'H' : 0.030034,
    'G' : 0.024705,
    'B' : 0.020720,
    'F' : 0.018121,
    'Y' : 0.017779,
    'W' : 0.012899,
    'K' : 0.011016,
    'V' : 0.010074,
    'X' : 0.002902,
```

```
    'Z' : 0.002722,
    'J' : 0.001965,
    'Q' : 0.001962
}

sorted_letters = sorted(freq, key=lambda k: freq[k])

sorted_eng_letters = sorted(eng_freq, key=lambda k: eng_freq[k])

subs = {}
for i in range(26):
    subs[sorted_letters[i]] = sorted_eng_letters[i]

decrypted = ""
for c in text:
    if c == '\n' or c == ' ' or c == '.' or c ==',':
        decrypted = decrypted + c
    else:
        decrypted = decrypted + subs[c]



#First word is RCE, probably the decrypted word is THE

swap(subs, 'T', 'R')

swap(subs, 'H', 'C')

#HAFE -> HAVE
swap(subs, 'V', 'F')

#ALE -> ARE
swap(subs, 'R', 'L')

#CACER -> PAPER
swap(subs, 'C', 'P')

#THEORW -> THEORY
swap(subs, 'W', 'Y')

#REAS->REAL
swap(subs, 'S', 'L')

#PROMLECN -> PROBLEMS
swap(subs, 'M', 'B')

swap(subs, 'C', 'M')

swap(subs, 'N', 'S')

#HOKEVER -> HOWEVER
swap(subs, 'K', 'W')

#EAUH UASE -> EACH CASE
swap(subs, 'U', 'C')

#COMPDTABLE -> COMPUTABLE
swap(subs, 'D', 'U')

#IUMBERS -> NUMBERS
swap(subs, 'I', 'N')

#BRIEGLY -> BRIEFLY
swap(subs, 'G', 'F')

#EGPRESSIONS -> EXPRESSIONS
swap(subs, 'G', 'X')

#ALTHOUKH -> ALTHOUGH
swap(subs, 'K', 'G')

#EKUALLY -> EQUALLY
swap(subs, 'K', 'Q')
```

```python

decrypted = ""
for c in text:
    if c == '\n' or c == ' ' or c == '.' or c ==',':
        decrypted = decrypted + c
    else:
        decrypted = decrypted + subs[c]

print(decrypted)
```

Άσκηση 2

1. Έστω ότι επιλέγουμε ότι $m_1 = 0$, τότε $c_1 = a0 + b \Rightarrow b = c_1$. Ακόμη, επιλέγουμε $m_2 = 1$, οπότε $c_2 = a1 + b \Rightarrow a = c_2 - c_1$.
2. Αν γίνει εξαντλητική αναζήτηση για τα $k_1 = (a_1, b_1), k_2 = (a_2, b_2)$, τότε πρέπει να εξεταστούν $25^2 \cdot 26^2$ τιμές (αν θεωρήσουμε $a \neq 0$ και εξετάσουμε την τετριμένη περίπτωση $a = 1, b = 0$), δηλαδή το τετράγωνο των περιπτώσεων της προηγούμενης περίπτωσης. Όμως, στην πραγματικότητα δεν χρειάζεται να γνωρίζουμε τους τέσσερις αυτούς αριθμούς, αφού $Enc(k, m) = Enc(k_2, Enc(k_1, m)) = Enc(k_2, a_1m + b_1) = a_2(a_1m + b_1) + b_2 = (a_2a_1)m + (a_2b_1 + b_2)$. Επομένως, αρκεί να βρούμε τις τιμές που παίρνουν τα a_2a_1 και $a_2b_1 + b_2$, οπότε το σύστημα δεν γίνεται πιο ασφαλές από το προηγούμενο.

Άσκηση 3

Αρχικά, υπολογίζεται το μήκος κλείδιου. Προς αυτό, δοκιμάζονται στη σειρά μήκη κλειδιού από το 1 και αυξάνοντας, υπολογίζοντας τους δείκτες σύμπτωσης για κάθε στήλη που προκύπτει για το αντίστοιχο μήκος, και εντέλει επιλέγεται ως μήκος το πρώτο για το οποίο ο δείκτης σύμπτωσης κάθε στήλης είναι τουλάχιστον 0.05.
Στη συνέχεια, για την εύρεση του κλειδιού υπολογίζεται η εντροπία της στήλης χρησιμοποιώντας τις συχνότητες εμφάνισης των γραμμάτων στην αγγλική γλώσσα. Επιλέγεται ως αντίστοιχο κλειδί αυτό το οποίο δίνει την χαμηλότερη εντροπία.
Επειδή με τον παραπάνω τρόπο για το συγκεκριμένο κείμενο βρίσκεται η λύση δεν υπολογίζονται 10 κείμενα όπως αναφέρει η εκφώνηση. Θα μπορούσε το πρόγραμμα που χρησιμοποιήθηκε να εξετάζει για το κλειδί και τη δεύτερη μικρότερη εντροπία ή και την τρίτη και από τους συνδυασμούς που θα έβρισκε να επέλεγε ως πιθανότερους τους 10 με τους μεγαλύτερους δείκτες σύμπτωσης.
Με τον κώδικα που ακολουθεί βρίσκεται το κλειδί:
XTQWYJCLHM
Το PLAINTEXT:
WE STAND TODAY ON THE BRINK OF A REVOLUTION IN CRYPTOGRAPHY. THE DEVELOPMENT OF CHEAP DIGITAL HARDWARE HAS FREED IT FROM THE DESIGN LIMITATIONS OF MECHANICAL COMPUTING AND BROUGHT THE COST OF HIGH GRADE CRYPTOGRAPHIC DEVICES DOWN TO WHERE THEY CAN BE USED IN SUCH COMMERCIAL APPLICATIONS AS REMOTE CASH DISPENSERS AND COMPUTER TERMINALS. IN TURN, SUCH APPLICATIONS CREATE A NEED FOR NEW TYPES OF CRYPTOGRAPHIC SYSTEMS WHICH MINIMIZE THE NECESSITY OF SECURE KEY DISTRIBUTION CHANNELS AND SUPPLY THE EQUIVALENT OF A WRITTEN SIGNATURE. AT THE SAME TIME, THEORETICAL DEVELOPMENTS IN INFORMATION THEORY AND COMPUTER SCIENCE SHOW PROMISE OF PROVIDING PROVABLY SECURE CRYPTOSYSTEMS, CHANGING THIS ANCIENT ART INTO A SCIENCE. THE DEVELOPMENT OF COMPUTER CONTROLLED COMMUNICATION NETWORKS PROMISES EFFORTLESS AND INEXPENSIVE CONTACT BETWEEN PEOPLE OR COMPUTERS ON OPPOSITE SIDES OF THE WORLD, REPLACING MOST MAIL AND MANY EXCURSIONS WITH TELECOMMUNICATIONS. FOR MANY APPLICATIONS THESE CONTACTS MUST BE MADE SECURE AGAINST BOTH EAVESDROPPING AND THE INJECTION OF ILLEGITIMATE MESSAGES. AT PRESENT, HOWEVER, THE SOLUTION OF SECURITY PROBLEMS LAGS WELL BEHIND OTHER AREAS OF COMMUNICATIONS TECHNOLOGY. CONTEMPORARY CRYPTOGRAPHY IS UNABLE TO MEET THE REQUIREMENTS, IN THAT ITS USE WOULD IMPOSE SUCH SEVERE INCONVENIENCES ON THE SYSTEM USERS, AS TO ELIMINATE MANY OF THE BENEFITS OF TELEPROCESSING. THE BEST KNOWN CRYPTOGRAPHIC PROBLEM IS THAT OF PRIVACY: PREVENTING THE UNAUTHORIZED EXTRACTION OF INFORMATION FROM COMMUNICATIONS OVER AN INSECURE CHANNEL. IN ORDER TO USE CRYPTOGRAPHY TO ENSURE PRIVACY, HOWEVER, IT IS CURRENTLY NECESSARY FOR THE COMMUNICATING PARTIES TO SHARE A KEY WHICH

IS KNOWN TO NO ONE ELSE. THIS IS DONE BY SENDING THE KEY IN ADVANCE OVER SOME SECURE CHANNEL SUCH AS PRIVATE COURIER OR REGISTERED MAIL. A PRIVATE CONVERSATION BETWEEN TWO PEOPLE WITH NO PRIOR ACQUAINTANCE IS A COMMON OCCURRENCE IN BUSINESS, HOWEVER, AND IT IS UNREALISTIC TO EXPECT INITIAL BUSINESS CONTACTS TO BE POSTPONED LONG ENOUGH FOR KEYS TO BE TRANSMITTED BY SOME PHYSICAL MEANS. THE COST AND DELAY IMPOSED BY THIS KEY DISTRIBUTION PROBLEM IS A MAJOR BARRIER TO THE TRANSFER OF BUSINESS COMMUNICATIONS TO LARGE TELEPROCESSING NETWORKS.

Και ο δείκτης σύμπτωσης υπολογίζεται ως 0.06583705908983145.

Χρησιμοποιήθηκε ο ακόλουθος κώδικας python:

```python
    import re
import math

def freq(str):
    freq = {}
    for i in "QWERTYUIOPASDFGHJKLZXCVBNM":
        freq[i] = 0
    for c in str:
        if ord(c) < ord('A') or ord(c) > ord('Z'):
            continue
        freq[c] += 1
    return freq


text ="""
ZL CXCEB IHRDF YR VYC QKWQR YJ C ICKHZXASSP ZL RKMSAYKTRNWR. HKL NIXVJDIAHUD SH
TFTTD GPQMVRJ WTFGDKVG YYH YFHLN MV WPDF HKL NIUZEC EWPPDEVZMCL CI TOGJRLXVOO
JYQRLRXGU DUN FTFSVAH WOO GQJR DY VLNR KTRBT VFBWDSIIYEAWF KOZKTCH WCZU
DS YYCGX HKLI GCE ZT NGHK SR ULAW VCPTOVEZYA TDSSSGCKGDGG DZ BIOFRT VOVO
NMUGCCLSUZ KRF TMBIIWLB XGIKXGOOZ. SR VLPC, LIFO KTRCGRTHLVXW EICPMS D UOIF
WMG GSZ AITGJ MU VFBWDSIIYEAWF ZIWVVKH PVLJR QKEGBBNH ARI PVATLGLAI SH JCRNFH
ROC FZQIKWEBDMQE AWTBQLVW CEB HNDSSI XJV CFNWYHVIPK MU T KUPDXGE QXZBDAEVG. RR
IAS VHWI VZKT, MVHVBIVZAPE RHCOPQGKTGHV PX MPWMGFOWPYR VYCDKM DUN GQDNJMSU
ZMMGEAT LVRD ZVQDGHX CI WBSXZBXGU SYYZCSJN LSFBBI EIWEMCVFCXGDQ, RAOQNSRI
KFXL OQJSIPK YGM WQAY E UTGTGQH. ARI FVTTECSTORV FD RHASBDIT TMCMFRSVIF
TMBFIQPMEVZMC GSWDYVMJ NGHALZOW GWDDKHOLCW CEB XGSAWORUZTT VCQAKGV SCIPSHU
ZIQGJT HF FVWTWKCGL CQ VZTQJGIX GLKOW QW RWX KRYVH, TVNATQLUQ QQJR BTWO
HXH ORLN XLFBBWKFLH PWWO DINVADFAXUSGCKGDGG. IVB QCEW PIDOPMEVZMCL HKLCI
EFLITQWZ WYUK ZT FOGL CIELPT TUDPXWV SMIA SDCOWFIMEIWQN KRF KFT BBMLMXKFL DY
WOSOKKKGBTHH TOWURETL. OW WBIUVLI, ACZLFIT, KFT LCOBDMQE MU LSFBBMVP NGHPOLWW
NREH PSOS LIJZLS HHKLB ETVYH HT FVWQWEGRTHLVXW VVAWGCOVQC. EFLIXASVBETP
AGRDWVQVCGFN BG XUKFNV RD FSHA DLG ICFNWULWIPKQ, XG HKHD MVJ SHX KRBVH
KDNDLS VBML UVTTKS LUMSPMCCBSQJOW QE RWX GBZDIO LQTKG, DZ DS GCGBBBDAO QCEW
DY HKL LIPVDXMG RM DINVNGHQHZCMPX. RWX PHZD OPFUC VFBWDSIIYEAWF WBSDCCB BG
WOKX QW NGBJDJI: TTVTTGHLUQ XJV SCTIWOYVKQCS XLWYKGVZMC HT LUPSTDYIBCQ MBSO
TMBFIQPMEVZMCL CYLB EP ZLHXQXYO GJRLCXZ. LU YVFVP IH IVL MVAGRDZFDWRC VF CCLIUL
ZVKMYRR, VRDOZGI, GI BG FBBVGERAR BHJOWURPN YCU ARI EFKBNBLJKXKEE ETFWPOW VF
QWTFH H UIA NFXVV LZ URQNL IH BR VXI GCQT. MVLZ SW FFLT UM VLXHKEE IAS NLI MP
RBKTBFL YZGI QDFS VLMYTV AWTBQLV WWTF PL DUPFEVV ADNFLLB ST ICVBGWLBIF DYXE.
O SYSZCKC RHBYLBWCKGDG PHAGIGE RLH DHVZPG NGIA BR WBMQI YRJIDPXXCEAT BG D
JYQOFL DVQXYBIPTC XG PXZSRGJQ, WHKHCOV, CEB XM WV BXVGRJXLHLJ DS GONTVH LUSXKRJ
QNGLUOWU TMCMOFAC XQ SC EHGWWYRGU JDGU HUYYIY DDK YHFC XQ SC IKOQZWMVKCS
UM VVWI RYWHBQDS WICEQ. IAS FVCX CEB SXZDF SQRFQTW PB ARMU BCN WWVABMDLRXHB
SYYFNVK XL O PHTST SYGKWHY DS VYC IKOQZPIT FD QNGLUOWU TMBFIQPMEVZMCL HR SKVIV
RTESSYYGGJQXGU QLDAQIIH.
"""

letters = re.sub('[^a-zA-Z]', '', text)


vig_len = 0
ic = 0

for r in range(1, 101):
    for i in range(r):
        col = letters[i::r]
        fr = freq(col)
        ic = 0
        for c in "QWERTYUIOPASDFGHJKLZXCVBNM":
            ic += fr[c] * (fr[c] - 1)
        ic /=  len(col) * (len(col) - 1)
```

```
        if ic < 0.05:
            break
    if ic >= 0.05:
        vig_len = r
        break

eng_freq = {
    'E' : 0.111607,
    'A' : 0.084966,
    'R' : 0.075809,
    'I' : 0.075448,
    'O' : 0.071635,
    'T' : 0.069509,
    'N' : 0.066544,
    'S' : 0.057531,
    'L' : 0.054893,
    'C' : 0.045338,
    'U' : 0.036308,
    'D' : 0.033844,
    'P' : 0.031671,
    'M' : 0.030129,
    'H' : 0.030034,
    'G' : 0.024705,
    'B' : 0.020720,
    'F' : 0.018121,
    'Y' : 0.017779,
    'W' : 0.012899,
    'K' : 0.011016,
    'V' : 0.010074,
    'X' : 0.002902,
    'Z' : 0.002722,
    'J' : 0.001965,
    'Q' : 0.001962
}


key = []
for i in range(r):
    col = letters[i::r]

    fr = freq(col)
    h = []
    for j in range(26):
        entr = 0
        for k in range(26):
            entr -= fr[chr(k + ord('A'))] * eng_freq[chr((j+k)%26 +ord('A') )]
        h.append(entr)
    min = 0
    c = 0
    for i, ent in enumerate(h):
        if ent < min:
            min = ent
            c = i
    key.append(c)
    print(chr(ord('A') + c), end="")

print()

decrypted = ""
k = 0
for t in text:
    if ord(t) < ord('A') or ord(t) > ord('Z'):

        decrypted += t
        continue
    c = ord(t) + key[k%r]
    if c > ord('Z'):
        c -= 26

    decrypted += chr(c)
    k +=1

```

```
print(decrypted)
fr = freq(decrypted)
ic = 0
n = 0
for c in "QWERTYUIOPASDFGHJKLZXCVBNM":
    ic += fr[c] * (fr[c] - 1)
    n += fr[c]
ic /=  n * (n - 1)

print(ic)
```

Άσκηση 4

1. Σε ένα σύστημα που διαθέτει τέλεια μυστικότητα δεν είναι αναγκαίο κάθε να επιλέγεται με την ίδια πιθανότητα. Ως αντιπαράδειγμα θεωρούμε $M = \{0,1\}, C = \{A,B\}, K = \{K_1,K_2,K_3,K_4\}$, όπου $Pr[K_1] = \frac{1}{4}, Pr[K_2] = \frac{1}{3}, Pr[K_3] = \frac{1}{4}, Pr[K_4] = \frac{1}{6}$, και $K_1 = \{0 \to A, 1 \to B\}, K_2 = \{0 \to B, 1 \to A\}, K_3 = \{0 \to A, 1 \to B\}, K_4 = \{0 \to B, 1 \to A\}$. Το παραπάνω σύστημα είναι τέλια μυστικό, εφόσον $Pr[M = 0] = Pr[M = 1]$, όμως δεν έχουν όλα τα κλειδιά την ίδια πιθανότητα.
Αν οι χώροι είναι ισοπληθικοί είναι αναγκαίο κάθε κλειδί να επιλέγεται με την ίδια πιθανότητα. Αρκεί να δειχθεί ότι αν διατίθεται τέλεια μυστικότητα, τότε κάθε κλειδί επιλέγεται με την ίδια πιθανότητα.
Εφόσον διατίθεται τέλεια μυστικότητα, τότε $\forall x_1, x_2 \in M, y \in C, Pr[C = y|M = x_1] = Pr[C = y|M = x_2] \Leftrightarrow Pr[C = y, M = x_1] = Pr[C = y, M = x_2]$, για ισοπίθανα μηνύματα. Όμως, όταν οι χώροι είναι ισοπληθικοί υπάρχει ακριβώς ένα κλειδί K_i και ένα κλειδί K_j τέτοια, ώστε $Enc_{k_i}(x_1) = y$ και $Enc_{k_j}(x_2) = y$, άρα $Pr[C = y, M = x_1] = Pr[K_i], Pr[C = y, M = x_2] = Pr[K_J]$, ενώ πρέπει να ισχύει ότι $K_i \neq K_j$, για να είναι δυνατή η αποκρυπτογράφηση. Έτσι, για να ισχύει η συνθήκη τέλειας μυστικότητας για κάθε $x_1, x_2 \in M, y \in C$ πρέπει όλα τα κλειδιά να είναι ισοπίθανα.
2.
i. Για κάθε $x \in M, y \in C$ έχουμε ότι

$$Pr[M = x|C = y] = Pr[M = x] \Leftrightarrow$$

$$\frac{Pr[M = x, C = y]}{Pr[C = y]} = Pr[M = x] \Leftrightarrow$$

$$\frac{Pr[M = x, C = y]}{Pr[M = x]} = Pr[C = y] \Leftrightarrow$$

$$Pr[C = y|M = x] = Pr[C = y]$$

Η παραπάνω απόδειξη ισχύει για $Pr[C = y] \neq 0, Pr[M = x] \neq 0$, όμως η περίπτωση που κάποια πιθανότητα ισούται με μηδέν είναι τετριμένη, οπότε ισχύει για κάθε τιμή που μπορεί να πάρουν οι πιθανότητες.
ii. <u>Ευθύ</u>
Από το πρώτο ερώτημα έχουμε $\forall x \in M, y \in C$:

$$Pr[M = x|C = y] = Pr[M = x] \Leftrightarrow$$

$$Pr[C = y|M = x] = P[rC = y]$$

Οπότε $\forall x_1, x_2 \in M, y \in C$ ισχύει ότι:

$$Pr[C = y|M = x_1] = Pr[C = y]$$

$$Pr[C = y|M = x_2] = Pr[C = y]$$

Επομένως,

$$Pr[C = y|M = x_1] = Pr[C = y|M = x_2]$$

<u>Αντίστροφο</u>
Αν $\forall x_1, x_2 \in M, y \in C$ ισχυεί ότι $Pr[C = y|M = x_1] = Pr[C = y|M = x_2]$, τότε $\forall x \in M, y \in C$ είναι $Pr[C = y|M = x] = k$, για κάποια σταθερά $k \in [0,1]$. Οπότε αν $M = \{x_1, x_2, ..., x_n\}$ έχουμε ότι:

$$Pr[C = y|M = x_1] = k$$

$$Pr[C = y|M = x_2] = k$$
$$...$$
$$Pr[C = y|M = x_n] = k$$

Ή ισοδύναμα:

$$Pr[C = y, M = x_1] = kPr[M = x_1]$$
$$Pr[C = y, M = x_2] = kPr[M = x_2]$$
$$...$$
$$Pr[C = y, M = x_n] = kPr[M = x_n]$$

Αθροίζοντας όλες τις παραπάνω σχέσεις παίρνουμε ότι:

$$Pr[C = y, M = x_1]+Pr[C = y, M = x_2]+...+Pr[C = y, M = x_n] = k(Pr[M = x_1]+Pr[M = x_2]+...+Pr[M = x_n]) \Leftrightarrow$$
$$Pr[C = y] = k$$

Επομένως, έχουμε ότι:

$$Pr[C = y|M = x_n] = Pr[C = y]$$

Η οποία είναι ισοδύναμη με την συνθήκη τέλειας μυστικότητας του Shannon σύμφωνα με το ερώτημα 1.

Άσκηση 5

Αν αφαιρέσουμε ένα κλειδί,τότε παραβιάζεται η συνθήκη τέλειας μυστικότητας του Shannon, αφού $|M| = |C| > |K|$, ενώ είναι αναγκαία συνθήκη για μυστικότητα να ισχύει $|M| \leq |C| \leq |K|$, όμως έχουμε ότι $|M| = |C| = 2^{\lambda}$, αφού μπορούμε να έχουμε κάθε πιθανή συμβολοσειρά, ενώ με κλειδί που αποτελείται μόνο από 1 συμπεραίνουμε άμεσα ότι κάθε πιθανή συμβολοσειρά μπορεί να είναι cipher. Όμως, το πλήθος των κλειδιών είναι $2^{\lambda} - 1$, αφού είναι όλες οι πιθανές συμβολοσειρές εκτός από αυτή που αποτελείται μόνο από 0.

Άσκηση 6

1. Ορίζουμε τη συνάρτηση:

$$Dec(k, c) = (k^{-1}c) \mod p$$

Όπου γνωρίζουμε ότι υπάρχει $k^{-1} \in \mathbb{Z}_p^*$ αφού p πρώτος.
2. Έστω ότι $c = (km) \mod p$, τότε η συνάρτηση αποκρυπτογράφησης μας δίνει ότι:

$$Dec(k, c) = k^{-1}(km) = (k^{-1}k)m = m \mod p$$

Δηλαδή παίρνουμε το αρχικό μήνυμα.
3. Εφόσον $\forall k_i \in \mathbb{Z}_p^*, Pr[k = k_i] = \frac{1}{p-1}$, τότε το σύστημα είναι τέλεια ασφαλές, αφού για p πρώτο για κάθε κείμενο x και cipher c θα υπάρχει ακριβώς ένα k, τέτοιο ώστε $Enc(k, x) = c$, και αφού κάθε k είναι ισοπίθανο θα ισχύει η συνθήκη τέλειας μυστικότητας. Το παραπάνω προκύπτει από το γεγονός ότι η $\mathbb{Z}_p^*$ είναι ομάδα.

Άσκηση 7

1. Έστω ότι n δεν είναι πρώτος, τότε υπάρχουν $p, q \in \{2, 3, ..., n-1\}$ τέτοια, ώστε $n = p \cdot q$, οπότε
$2^n - 1 = 2^{pq} - 1 = (2^p)^q - 1^q = (2^p - 1) \cdot (2^{(q-1)p} + 2^{(q-2)p} + ... + 2^p + 1)$
Άτοπο, αφού $2^n - 1$ πρώτος, άρα δεν παραγοντοποιείται.
Επομένως, n πρώτος.

2.

i. Από μικρό θεώρημα Fermat έχουμε ότι $2^{p-1} \equiv 1(\mod p)$, δηλαδή για κάποιο $k \in mathbbZ$ είναι:
$2^{p-1} = kp + 1 \Leftrightarrow 2^p = 2kp + 2 \Leftrightarrow 2^p - 1 = 2kp + 1 \Leftrightarrow M_p \equiv 1(\mod p)$.

ii. Έχουμε ότι $gcd(2, 2^p - 1) = 1$, οπότε από θεώρημα Euler παίρνουμε ότι $2^{\phi(2^p-1)} \equiv 1(\mod 2^p - 1)$.
Επίσης, ισχύει ότι $2^p \equiv 1(\mod 2^p - 1)$.
Έστω ότι p δεν διαιρεί το $\phi(2^p - 1)$, οπότε υπάρχουν $k \in \mathbb{Z}, r \in \{1, 2, ..., p-1\}$ τέτοια, ώστε $\phi(2^p - 1) = kp + r$.
Έχουμε $2^{\phi(2^p-1)} \equiv 2^{pk+r} \equiv (2^p)^q \cdot 2^r \equiv 2^r \equiv 1(\mod 2^p - 1)$ το οποίο είναι άτοπο, οπότε θα είναι $r = 0$, δηλαδή το p διαιρεί το $\phi(2^p - 1)$.

Άσκηση 8

Επειδή p, q διαφορετικοί πρώτοι ο q δεν διαρεί τον p, άρα από μικρό θεώρημα Fermat παίρνουμε ότι:

$$p^{q-1} \equiv 1 \mod q$$

Ακόμη, ισχύει ότι:

$$q^{p-1} \equiv 0 \mod q$$

Άρα, έχουμε ότι:

$$p^{q-1} + q^{p-1} \equiv 1 \mod q$$

Ομοίως, από μικρό θεώρημα Fermat, αφού ο p δεν διαιρεί τον q παίρνουμε ότι:

$$q^{p-1} \equiv 1 \mod p$$

Επίσης, ισχύει:

$$p^{q-1} \equiv 0 \mod p$$

Οπότε, προκύπτει:

$$p^{q-1} + q^{p-1} \equiv 1 \mod p$$

Επομένως, από την ειδική περίπτωση του Κινέζικου Θεωρήματος Υπολοίπων έχουμε:

$$p^{q-1} + q^{p-1} \equiv 1 \mod pq$$

Άσκηση 9

ΕΦόσον $p > 2$ πρώτος αριθμός, ισχύει ότι $\mathbb{Z}_p^* = \{1, 2, ..., p-1\}$, οπότε για κάθε αριθμό υπάρχει αριθμός που αθροίζουν στο 0 ως εξής:

$$1 + (p-1) \equiv 0 \mod p$$

$$2 + (p-2) \equiv 0 \mod p$$

$$...$$

$$\frac{p-1}{2} + \left(\frac{p-1}{2} + 1\right) \equiv 0 \mod p$$

Όπου $\frac{p-1}{2}$ είναι ακέραιος αφού κάθε p πρώτος διάφορος του 2 είναι περιττός. Αθροίζοντας όλες τις παραπάνω σχέσεις παίρνουμε ότι:

$$1 + 2 + ... + (p-1) \equiv 0 \mod p$$

$$\sum_{\beta \in \mathbb{Z}_p^*} \beta \equiv 0 \mod p$$

Η σχέση $\sum_{\beta \in \mathbb{Z}_p^*} \beta = \sum_{\beta \in \mathbb{Z}_p^*} \beta^{-1}$ προκύπτει άμεσα, αφού για κάθε $\beta \in \mathbb{Z}_p^*$, ισχύει ότι $\beta^{-1} \in \mathbb{Z}_p^*$, άρα τα δύο αθροίσματα είναι αντιμεταθέσεις των όρων.

Άσκηση 10

1. <u>Ευθύ</u>
Αν n πρώτος, τότε $gcd(n, i) = 1, \forall i \in \{1, 2, ..., m\}$, οπότε $\sum_{j=1}^{m} gcd(n, j) = \sum_{j=1}^{m} 1 = m$.
<u>Αντίστροφο</u>
Έστω ότι $\sum_{j=1}^{m} gcd(n, j) = m$ και n όχι πρώτος, δηλαδή υπάρχει $1 < q < n$, τέτοιο, ώστε $gcd(q, n) = q$.
Από τη σχέση $\sum_{j=1}^{m} gcd(n, j) = m$ παίρνουμε $gcd(n, i) = 1, \forall i \in \{1, 2, ..., m\}$, αφού $gcd(n, j) \geq 1$, οπότε θα πρέπει να ισχύει η ισότητα, αφού έχουμε m όρους.
Επομένως, πρέπει να ισχύει $q > m$, όμως θα είναι $n = q \cdot l$ και επειδή $q > m$ θα πρέπει να είναι $l < m$, όμως ισχύει και ότι $gcd(n, l) \neq 1$, αφού l διαιρεί τον n, οπότε καταλήγουμε σε άτοπο και n πρώτος.
2. Όπως δείξαμε η παραπάνω σχέση είναι ισοδύναμη με το ότι κάθε αριθμός είναι σχετικά πρώτος με το n, οπότε σχεδιάζουμε τον παρακάτω αλγόριθμο:
Για $i \in \{2, 3, ..., [\sqrt{n}]\}$:

Αν $gcd(n, i) > 1$, τότε σύνθετος
Αν ολοκληρωθεί ο βρόγχος, τότε πρώτος

Ο αλγόριθμος αυτός έχει την ίδια πολυπλοκότητα με τον αλγόριθμο που εξετάζει αν κάποιος αριθμός διαιρεί το n, δηλαδή $\mathcal{O}(\sqrt{n}\log^2 n)$, ενώ ο αλγόριθμος Rabin-Miller για k επαναλήψεις έχει bit complexity

$$\mathcal{O}(k\log^3 n)$$

, δηλαδή είναι πιο γρήγορος, το οποίο είναι και αναμενόμενο, καθώς είναι πιθανοτικός αλγόριθμος, οπότε αν ήταν πιο αργός δεν θα είχε πρακτική χρησιμότητα.

Άσκηση 11

Έστω n πρώτος και $\alpha \in \mathbb{Z}_n^+$, τότε από μικρό θεώρημα Fermat ισχύει ότι $\alpha^{n-1} \equiv 1 \mod n$, αφού το α δεν διαιρεί το n, καθώς ισχύει $\alpha < n$ και n πρώτος.
Για $n = 2$ τετριμένη περίπτωση, οπότε θεωρούμε $n > 2$, τότε $n - 1 = t2^h, h > 0$.
Για n πρώτο γνωρίζουμε ότι η εξίσωση $x^2 \equiv 1 \mod n$ έχει ακριβώς δύο λύσεις τις $\{1, -1\}$. Οπότε για κάθε n πρώτο, αν ισχύει ότι $\alpha^{t2^{k+1}} \equiv 1 \mod n$, τότε θα ισχύει $\alpha^{t2^k} \equiv 1$ ή $\alpha^{t2^k} \equiv -1 \mod n$. Οπότε $\alpha \in L_n$
Επειδή το α είναι τυχαίο ισχύει για κάθε $\alpha \in \mathbb{Z}_n^*$, άρα θα είναι $L_n = \mathbb{Z}_n^*$.

Άσκηση 12

Κλειστότητα

Έστω ότι $a_1, a_2 \in \mathbb{B}_1$, τότε υπάρχουν $b_1, b_2 \in \mathbb{G}_2$ τέτοια, ώστε $(a_1, b_1) \in \mathbb{G}_1 \times \mathbb{G}_2$ και $(a_2, b_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, οπότε $(a_1 + a_2, b_1 + b_2) \in \mathbb{G}_1 \times \mathbb{G}_2$, αφού $\mathbb{G}_1 \times \mathbb{G}_2$ ομάδα. Επομένως, $a_1 + a_2 \in \mathbb{B}_1$, άρα η $\mathbb{B}_1$ είναι κλειστή ως προς την πράξη $+$.

Ουδέτερο στοιχείο

Αν (e_1, e_2) το ουδέτερο στοιχείο της ομάδας $\mathbb{G}_1 \times \mathbb{G}_2$, τότε το e_1 είναι ουδέτερο στοιχείο της $\mathbb{B}_1$.

Συμμετρικό στοιχείο

Αφού $\mathbb{G}_1 \times \mathbb{G}_2$ ομάδα, το στοιχείο (a, b) έχει συμμετρικό κάποιο $(a^{'}, b^{'})$ με $(a, b) + (a^{'}, b^{'}) = (e_1, e_2)$, οπότε $a + a^{'} = e_1 = a^{'} + a$, άρα το $a^{'}$ είναι συμμετρικό του a.
Επομένως, αφού ισχύουν οι 3 παραπάνω συνθήκες η $\mathbb{B}_1$ είναι υποομάδα της $\mathbb{G}_1$.

Άσκηση 13

1. Έστω $b = g^{\frac{p-1}{d}}$, όπου $\frac{p-1}{d} \in \mathbb{Z}_p^*$, αφού d διαιρεί το $p - 1$. Τότε $b^d = g^{(\frac{p-1}{d})^d} = g^{\frac{p-1}{d}\cdot d} = g^{p-1} = 1$, αφού g γεννήτορας, άρα η τάξη του είναι $p - 1$. Επίσης, για κάθε $i \in \{1, 2, ..., d - 1\}$, είναι $b^i = g^{\frac{p-1}{d}\cdot i} \neq 1$, αφού $0 < \frac{p-1}{d} \cdot i < p - 1$. Επομένως, η τάξη του b είναι d.
2. Έστω το στοιχείο $b = g^s$, τότε για την τάξη του ισχύει $|b| = \frac{p-1}{gcd(p-1,s)}$. Επομένως, τάξη d έχει το στοιχείο b, αν ισχύει $\frac{p-1}{gcd(p-1,s} = d \Rightarrow gcd(p - 1, s) = \frac{p-1}{d}$, η οποία σχέση ισχύει για όλα τα s, τα οποία είναι της μορφής $\frac{p-1}{d} \cdot m, m \in \{1, 2, ..., d - 1\}$ με $gcd(m, d) = 1$, οπότε υπάρχουν $\phi(d)$ τέτοια s. Επομένως, υπάρχουν $\phi(d)$ στοιχεία τάξης d.
3. Υπάρχουν $\phi(d)$ γεννήτορες για την υποομάδα τάξης d, όπως προκύπτει από το ότι υπάρχουν $\phi(d)$ στοιχεία τάξης d και μία μόνο υποομάδα τάξης d, όπως αποδεικνύεται στο επόμενο ερώτημα.
4. Έστω υποομάδα H τάξης d. Αν m ο μικρότερος θετικός ακέραιος τέτοιος, ώστε $g^m \in H$, τότε το g^m είναι γεννήτορας της H, γιατί υπάρχουν $k \in \mathbb{Z}, r \in \{0, 1, ..., m - 1\}$ τέτοια, ώστε $p - 1 = km + r$.
Είναι:

$$\begin{aligned} g^r &= g^{p-1}g^{-km} \Rightarrow \\ &= 1 \cdot (g^m)^{-k} \Rightarrow \\ &= (g^m)^{-k} \end{aligned}$$

Οπότε το $g^r \in H$, όμως m ελάχιστος θετικός ακέραιος για τον οποίο ισχύει ότι $g^m \in H$, αρα $r = 0$.
Επομένως, ισχύει ότι $p - 1 = km$, δηλαδή m διαιρεί το $p - 1$.
Για την τάξη του στοιχείου g^m γνωρίζουμε ότι $|g^m| = \frac{p-1}{gcd(p-1,m)} = \frac{p-1}{m}$.
Όμως, η υποομάδα είναι τάξης d, άρα ισχύει $\frac{p-1}{m} = d \Rightarrow m = \frac{p-1}{d}$, επομένως υπάρχει μόνο μία υποομάδα τάξης d,

η οποία είναι η $< g^{\frac{p-1}{d}} >$.
5. Αν $\alpha \in < h >$, τότε υπάρχει $k \in \{1, 2, ..., d-1\}$ τέτοιο, ώστε $\alpha = h^k$ (για τη μη τετριμμένη περίπτωση $\alpha \neq 1$. Επομένως, αρκεί να υπολογιστούν οι δυνάμεις του h από 1 έως $d-1$ και να εξεταστεί αν είναι κάποια ίση με α.

Άσκηση 14

Χρησιμοποιήθηκε ο παρακάτω κώδικας σε python.

```python
import random

def fastmodpower(a, n, p):
  result = 1
  while n > 0:
    if n % 2 == 1:
      result = result * a % p
    n = n // 2
    a = a * a % p
  return result

def isPrime(n):
  if n<=1:
    return False
  if n<=3:
    return True
  if n%2==0:
    return False
  q = n -1
  while q % 2 == 0:
    q//=2
  for i in range (30):
    if not RabinMillerTest(n,q):
      return False
  return True

def RabinMillerTest(n, q):
  a=random.randint(2, n-2)
  x=fastmodpower(a,q,n)
  if x==1 or x==n-1:
    return True
  while q != n-1:
    x = (x * x) % n
    q = 2 * q
    if x == 1:
      return False
    if x==n-1:
      return True
  return False

numbers = [67280421310721, 170141183460469231731687303715884105727 if False else 170141183460469231731687303715884105 7, 2**1001-1, 2**2281-1, 2**9941-1, 2**19939-1]
numstr = ["67280421310721", "170141183460469231731687303715884105 7", "2^1001-1", "2^2281-1", "2^9941-1", "2^19939-1"]

for i in range(len(numbers)):
  print("Number", numstr[i], "is", "prime" if isPrime(numbers[i]) else "composite")
```

Τα αποτελέσματα που παίρνουμε για τους αριθμούς που ζητείται στην εκφώνηση να ελεγχθούν είναι:

Number 67280421310721 is prime
Number 1701411834604692317316873037158841057 is composite
Number $2^{1001} - 1$ is composite
Number $2^{2281} - 1$ is prime
Number $2^{9941} - 1$ is prime
Number $2^{19939} - 1$ is composite