Assignment - 1

CSE 462

Introduction To Computer Security & Forensics

Submitted by	Submitted to
Abdulla Al Mahmud Mugdo	Md. Shadmim Hasan Sifat
Reg no: 2019331048	Lecturer,
Submission Date: 14 February, 2024	Computer Science & Engineering,
	Shahjalal University of Science and Technology
	Email: shadmim-cse@sust.edu

Task-01

1.(a) Implement the Decryption part of Caesar Cipher which will take a key and a ciphertext as inputs and give a plaintext as output.

Solution:

In [1]:
```
# a

def caesar_decrypt(ciphertext, key):
    decrypted_text = ""

    for char in ciphertext:
        if char.isalpha():
            # Shift the character by the key value
            shifted_char = chr((ord(char) - key - ord('A')) % 26 + ord('A')) if char.isupper() else chr((ord(char) - ke
            decrypted_text += shifted_char
```

```
        else:
            # Leave non-alphabetic characters unchanged
            decrypted_text += char

    return decrypted_text

# Example usage
ciphertext = "XNZ XVMIDQVG VO DDXO WPDGYDIB NPNO DN BJDIB OJ WZ BMZVO VBVDI"
key=21
decrypted_text = caesar_decrypt(ciphertext, key)
print(f"Key {key}: {decrypted_text}")
```

```
Key 21: CSE CARNIVAL AT IICT BUILDING SUST IS GOING TO BE GREAT AGAIN
```

1.(b) Use the above code to crack the following Caesar ciphertext, to identify the text encrypted:

XNZ XVMIDQVG VO DDXO WPDGYDIB NPNO DN BJDIB OJ WZ BMZVO VBVDI.

Also describe in detail (with code) how you have identified the plaintext.

Solution:

In [2]:
```
# b
# try all possible keys to generate all possible text that encrypted and we can choose appropriate plaintext from that
def try_all_possible_keys(ciphertext):
    for key in range(1, 26):
        decrypted_text = caesar_decrypt(ciphertext, key)
        print(f"Key {key}: {decrypted_text}")

# Example usage
ciphertext = "XNZ XVMIDQVG VO DDXO WPDGYDIB NPNO DN BJDIB OJ WZ BMZVO VBVDI"

try_all_possible_keys(ciphertext)

# key=25
```

```
Key 1: WMY WULHCPUF UN CCWN VOCFXCHA MOMN CM AICHA NI VY ALYUN UAUCH
Key 2: VLX VTKGBOTE TM BBVM UNBEWBGZ LNLM BL ZHBGZ MH UX ZKXTM TZTBG
Key 3: UKW USJFANSD SL AAUL TMADVAFY KMKL AK YGAFY LG TW YJWSL SYSAF
Key 4: TJV TRIEZMRC RK ZZTK SLZCUZEX JLJK ZJ XFZEX KF SV XIVRK RXRZE
Key 5: SIU SQHDYLQB QJ YYSJ RKYBTYDW IKIJ YI WEYDW JE RU WHUQJ QWQYD
Key 6: RHT RPGCXKPA PI XXRI QJXASXCV HJHI XH VDXCV ID QT VGTPI PVPXC
Key 7: QGS QOFBWJOZ OH WWQH PIWZRWBU GIGH WG UCWBU HC PS UFSOH OUOWB
Key 8: PFR PNEAVINY NG VVPG OHVYQVAT FHFG VF TBVAT GB OR TERNG NTNVA
Key 9: OEQ OMDZUHMX MF UUOF NGUXPUZS EGEF UE SAUZS FA NQ SDQMF MSMUZ
Key 10: NDP NLCYTGLW LE TTNE MFTWOTYR DFDE TD RZTYR EZ MP RCPLE LRLTY
Key 11: MCO MKBXSFKV KD SSMD LESVNSXQ CECD SC QYSXQ DY LO QBOKD KQKSX
Key 12: LBN LJAWREJU JC RRLC KDRUMRWP BDBC RB PXRWP CX KN PANJC JPJRW
Key 13: KAM KIZVQDIT IB QQKB JCQTLQVO ACAB QA OWQVO BW JM OZMIB IOIQV
Key 14: JZL JHYUPCHS HA PPJA IBPSKPUN ZBZA PZ NVPUN AV IL NYLHA HNHPU
Key 15: IYK IGXTOBGR GZ OOIZ HAORJOTM YAYZ OY MUOTM ZU HK MXKGZ GMGOT
Key 16: HXJ HFWSNAFQ FY NNHY GZNQINSL XZXY NX LTNSL YT GJ LWJFY FLFNS
Key 17: GWI GEVRMZEP EX MMGX FYMPHMRK WYWX MW KSMRK XS FI KVIEX EKEMR
Key 18: FVH FDUQLYDO DW LLFW EXLOGLQJ VXVW LV JRLQJ WR EH JUHDW DJDLQ
Key 19: EUG ECTPKXCN CV KKEV DWKNFKPI UWUV KU IQKPI VQ DG ITGCV CICKP
Key 20: DTF DBSOJWBM BU JJDU CVJMEJOH TVTU JT HPJOH UP CF HSFBU BHBJO
Key 21: CSE CARNIVAL AT IICT BUILDING SUST IS GOING TO BE GREAT AGAIN
Key 22: BRD BZQMHUZK ZS HHBS ATHKCHMF RTRS HR FNHMF SN AD FQDZS ZFZHM
Key 23: AQC AYPLGTYJ YR GGAR ZSGJBGLE QSQR GQ EMGLE RM ZC EPCYR YEYGL
Key 24: ZPB ZXOKFSXI XQ FFZQ YRFIAFKD PRPQ FP DLFKD QL YB DOBXQ XDXFK
Key 25: YOA YWNJERWH WP EEYP XQEHZEJC OQOP EO CKEJC PK XA CNAWP WCWEJ
```

It can be seen from the all possible result that for the ***key 21*** there is a meaningful sentence. All the other decrypted texts are meaningless. So, the plaintext will be:

CSE CARNIVAL AT IICT BUILDING SUST IS GOING TO BE GREAT AGAIN

1.(c) Use the above code to decode these messages, which were encoded using the same Caesar cipher (among these 03 cipher blocks) and fill up the boxes.

Solution:

In [3]:
```
# c

ciphertext = "ZKDW GR BRX JHW ZKHQ BRX FURVV D VQRZPDQ ZLWK D YDPSLUH? IURVWELWH"
try_all_possible_keys(ciphertext)

# key=3
```

```
Key 1: YJCV FQ AQW IGV YJGP AQW ETQUU C UPQYOCP YKVJ C XCORKTG? HTQUVDKVG
Key 2: XIBU EP ZPV HFU XIFO ZPV DSPTT B TOPXNBO XJUI B WBNQJSF? GSPTUCJUF
Key 3: WHAT DO YOU GET WHEN YOU CROSS A SNOWMAN WITH A VAMPIRE? FROSTBITE
Key 4: VGZS CN XNT FDS VGDM XNT BQNRR Z RMNVLZM VHSG Z UZLOHQD? EQNRSAHSD
Key 5: UFYR BM WMS ECR UFCL WMS APMQQ Y QLMUKYL UGRF Y TYKNGPC? DPMQRZGRC
Key 6: TEXQ AL VLR DBQ TEBK VLR ZOLPP X PKLTJXK TFQE X SXJMFOB? COLPQYFQB
Key 7: SDWP ZK UKQ CAP SDAJ UKQ YNKOO W OJKSIWJ SEPD W RWILENA? BNKOPXEPA
Key 8: RCVO YJ TJP BZO RCZI TJP XMJNN V NIJRHVI RDOC V QVHKDMZ? AMJNOWDOZ
Key 9: QBUN XI SIO AYN QBYH SIO WLIMM U MHIQGUH QCNB U PUGJCLY? ZLIMNVCNY
Key 10: PATM WH RHN ZXM PAXG RHN VKHLL T LGHPFTG PBMA T OTFIBKX? YKHLMUBMX
Key 11: OZSL VG QGM YWL OZWF QGM UJGKK S KFGOESF OALZ S NSEHAJW? XJGKLTALW
Key 12: NYRK UF PFL XVK NYVE PFL TIFJJ R JEFNDRE NZKY R MRDGZIV? WIFJKSZKV
Key 13: MXQJ TE OEK WUJ MXUD OEK SHEII Q IDEMCQD MYJX Q LQCFYHU? VHEIJRYJU
Key 14: LWPI SD NDJ VTI LWTC NDJ RGDHH P HCDLBPC LXIW P KPBEXGT? UGDHIQXIT
Key 15: KVOH RC MCI USH KVSB MCI QFCGG O GBCKAOB KWHV O JOADWFS? TFCGHPWHS
Key 16: JUNG QB LBH TRG JURA LBH PEBFF N FABJZNA JVGU N INZCVER? SEBFGOVGR
Key 17: ITMF PA KAG SQF ITQZ KAG ODAEE M EZAIYMZ IUFT M HMYBUDQ? RDAEFNUFQ
Key 18: HSLE OZ JZF RPE HSPY JZF NCZDD L DYZHXLY HTES L GLXATCP? QCZDEMTEP
Key 19: GRKD NY IYE QOD GROX IYE MBYCC K CXYGWKX GSDR K FKWZSBO? PBYCDLSDO
Key 20: FQJC MX HXD PNC FQNW HXD LAXBB J BWXFVJW FRCQ J EJVYRAN? OAXBCKRCN
Key 21: EPIB LW GWC OMB EPMV GWC KZWAA I AVWEUIV EQBP I DIUXQZM? NZWABJQBM
Key 22: DOHA KV FVB NLA DOLU FVB JYVZZ H ZUVDTHU DPAO H CHTWPYL? MYVZAIPAL
Key 23: CNGZ JU EUA MKZ CNKT EUA IXUYY G YTUCSGT COZN G BGSVOXK? LXUYZHOZK
Key 24: BMFY IT DTZ LJY BMJS DTZ HWTXX F XSTBRFS BNYM F AFRUNWJ? KWTXYGNYJ
Key 25: ALEX HS CSY KIX ALIR CSY GVSWW E WRSAQER AMXL E ZEQTMVI? JVSWXFMXI
```

After observing the all possible key with the decyphered plaintext we can see that for ***key 3*** there is a meaningful text. But for all other case there is no meaningful texts. So, the plaintext will be:

Z	**K**	**D**	**W**		**G**	**R**		**B**	**R**	**X**		**J**	**H**	**W**		**Z**	**K**	**H**	**Q**		**B**	**R**	**X**		**F**	**U**	**R**	**V**	**V**		**D**		**V**	**Q**	**R**	**Z**	**P**	**D**	**Q**		**Z**	**L**	**W**
W	H	A	T		D	O		Y	O	U		G	E	T		W	H	E	N		Y	O	U		C	R	O	S	S		A		S	N	O	W	M	A	N		W	I	T

WHAT DO YOU GET WHEN YOU CROSS A SNOWMAN WITH A VAMPIRE? FROSTBITE

Task-02

The following ciphers have been created using a substitution cipher. While decrypting them, describe in detail (step by step) how you have identified the plaintext. You may add necessary programs like frequency analysis, checking by trial- and-error

etc. you have used while deciphering them. You may add necessary programs like frequency analysis, checking by trial- and-error etc. you have used while deciphering them. For your convenience, a frequency distribution of English characters and list of hints are given.

In [4]:
```
from collections import Counter

def freq_analyze(ciphertext):
    ciphertext = ''.join(char for char in ciphertext if char.isalpha())
    frequencies_counter = Counter(ciphertext)
    sorted_frequencies = sorted(frequencies_counter.items(), key=lambda x: x[1], reverse=True)
    print("Character frequencies in the ciphertext (descending order):")
    for char, count in sorted_frequencies:
        print(f"'{char}': {count}")
```

In [5]:
```
def algo(ciphertext, mapping, words_per_line):
    print("Ciphertext:")
    print_formatted_text(ciphertext, words_per_line)
    encrypted_text = ''.join(mapping.get(char, '*') if char.isalpha() else char for char in ciphertext)
    print("\nDecrypted Text:")
    print_formatted_text(encrypted_text, words_per_line)
    check_duplicate_mapping(mapping)

def check_duplicate_mapping(mapping):
    seen = set()
    duplicates = set()
    for key, value in mapping.items():
        if value in seen:
            duplicates.add(value)
        else:
            seen.add(value)
    if duplicates:
        print("\nDuplicate mappings found:")
        for value in duplicates:
            print(f"{value}: {', '.join([key for key, val in mapping.items() if val == value])}")

def print_formatted_text(text, words_per_line):
    words = text.split()
    for i in range(0, len(words), words_per_line):
        print(' '.join(words[i:i+words_per_line]))

words_per_line = 10
```

2.(a)

gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut. vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl idsdavgf ha fgmlvsu bsvgdx vs ghbut gvzdf, mf vg tdycf gh amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf msx vsgh m ievutgde abgbed.

In [6]:
```
# Provided ciphertext
ciphertext = """gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha aevdsxftvc tdycf
bf gh id fgehsu aehz tmexftvcf. aevdsxf qms uvod bf gtd fgedsugt jd sddx
jtds yvad udgf ghbut. vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut
yhod,amzvyl, aevdsxf, msx hgtdef ftmed fghevdf ha avsxvsu qhzzhs
uehbsx jvgt fhzdhsd.gtded med zmsl idsdavgf ha fgmlvsubsvgdx vs ghbut
gvzdf, mf vg tdycf gh amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd
vzchegmsqd ha fgmlvsu bsvgdx tmf fgebqp m qthex mzhsu zmsl cdhcyd
gtehbuthbg tvfghel.pddcvsu zdzhevdf ha jtmg jd tmod mqqhzcyvftdx
gtehbuthbg tvfghel qms tdyc bf fdd thj vsxvovxbmyf msx qhzzbsvgvdf
tmod cdefdodedx gtehbut ghbut gvzdf msx vsgh m ievutgde abgbed.
"""
```

Initial mapping is empty and frequency analysis is shown below:

In [7]:
```
mapping = {}
algo(ciphertext, mapping, words_per_line)
freq_analyze(ciphertext)
```

Ciphertext:
```
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.
```

Decrypted Text:
```
*** ***** ** ******** ***** ******** **** *** ***** **
********** ***** ** ** ** ****** **** *********. ******* ***
**** ** *** ******** ** **** **** **** **** *****.
** ********, ****** ******* ***** ******* ****,******, *******, *** ******
***** ******* ** ******* ****** ****** **** *******.***** *** ****
******** ** ************ ** ***** *****, ** ** ***** **
**** *********** ********** **** *******. *** ********** ** ******* ******
*** ****** * ***** ***** **** ****** ********** *******.******* ********
** **** ** **** ************ ********** ******* *** **** **
*** *** ************ *** ************ **** ********** ******* ***** *****
*** **** * ******** ******.
```
Character frequencies in the ciphertext (descending order):
```
'd': 65
'g': 50
'v': 45
'f': 43
'h': 42
't': 41
's': 40
'e': 32
'm': 32
'b': 24
'u': 22
'x': 19
'c': 16
'z': 16
'a': 15
'y': 13
'q': 12
'l': 10
'j': 8
```

```
'o': 6
'i': 3
'k': 2
'p': 2
```

The first word of 8th line is a one letter word and that is in the middle of the sentence. There are two one letter word **a** and **I**. So I replace **a** with **m** as it is the middle of the sentence. The mapping:

m : a

In [8]:
```
mapping = {'m':'a'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
*** ***** ** ******** ***a* ****a*** **a* *** ***** **
********** ***** ** ** ** ****** **** *a*******. ******* *a*
**** ** *** ******** ** **** **** **** **** *****.
** a*******, ****** ******* ***** ******* ****,*a****, *******, a** ******
**a** ******* ** ******* ****** ****** **** *******.***** a** *a**
******** ** **a********** ** ***** *****, a* ** ***** **
*a** **a******** ****a***** **** ****a**. *** ******a*** ** **a**** ******
*a* ****** a ***** a**** *a** ****** ********** *******.******* ********
** **a* ** *a** a*********** ********** ******* *a* **** **
*** *** ********a** a** *********** *a** ********** ******* ***** *****
a** **** a ******** ******.
```

There are many 3 letter word with **gtd** and for three later words **the** is the most frequent. So, I replace-

g : t

t : h

d : e

In [9]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the ***t* ** *t*e**th e**a* e***a*** that the ***e* **
***e***h** he*** ** t* *e *t**** **** ha***h***. ***e*** *a*
***e ** the *t*e**th *e *ee* *he* ***e *et* t***h.
** a***t***, *e***e e***e** ***t* th****h ***e,*a****, ***e***, a** *the**
*ha*e *t***e* ** ******* ****** ****** **th ***e**e.the*e a*e *a**
*e*e**t* ** *ta*******te* ** t***h t**e*, a* *t he*** t*
*a*e *ha**e***** **t*at**** **th ****a*e. the *****ta**e ** *ta**** ***te*
ha* *t**** a *h*** a**** *a** *e***e th****h**t h**t***.*ee**** *e****e*
** *hat *e ha*e a********he* th****h**t h**t*** *a* he** **
*ee h** ********a** a** *******t*e* ha*e *e**e*e*e* th****h t***h t**e*
a** **t* a ****hte* **t**e.
```

The 4th word of 2nd line would be **to**. So, mapping will be:

h : o

In [10]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the ***t* ** *t*e**th e**a* e***a*** that the *o*e* o*
***e***h** he*** ** to *e *t*o** **o* ha***h***. ***e*** *a*
***e ** the *t*e**th *e *ee* *he* ***e *et* to**h.
** a***t*o*, *eo**e e***e** ***t* th*o**h *o*e,*a****, ***e***, a** othe**
*ha*e *to**e* o* ****** *o**o* **o*** **th *o*eo*e.the*e a*e *a**
*e*e**t* o* *ta*******te* ** to**h t**e*, a* *t he*** to
*a*e *ha**e***** **t*at*o** **th *o**a*e. the ***o*ta**e o* *ta**** ***te*
ha* *t**** a *ho** a*o** *a** *eo**e th*o**ho*t h**to**.*ee**** *e*o**e*
o* *hat *e ha*e a**o*****he* th*o**ho*t h**to** *a* he** **
*ee ho* ********a** a** *o*****t*e* ha*e *e**e*e*e* th*o**h to**h t**e*
a** **to a ****hte* **t**e.
```

In the 3rd word of 2nd last line the word **ha*e** may be **hate** or **have**. From the frequency for four word **have** is more frequent. So, mapping: **o : v**

In [11]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the ***t* ** *t*e**th e**a* e***a*** that the *o*e* o*
***e***h** he*** ** to *e *t*o** **o* ha***h***. ***e*** *a*
**ve ** the *t*e**th *e *ee* *he* ***e *et* to**h.
** a***t*o*, *eo**e e***e** ***t* th*o**h *ove,*a****, ***e***, a** othe**
*ha*e *to**e* o* ****** *o**o* **o*** **th *o*eo*e.the*e a*e *a**
*e*e**t* o* *ta*******te* ** to**h t**e*, a* *t he*** to
*a*e *ha**e***** **t*at*o** **th *o**a*e. the ***o*ta**e o* *ta**** ***te*
ha* *t**** a *ho** a*o** *a** *eo**e th*o**ho*t h**to**.*ee**** *e*o**e*
o* *hat *e have a**o*****he* th*o**ho*t h**to** *a* he** **
*ee ho* ****v***a** a** *o*****t*e* have *e**eve*e* th*o**h to**h t**e*
a** **to a ****hte* **t**e.
```

The 8th word of 5th line (**a*e**) can be **are** as it is the most frequent three letter sentence. So, the mapping:

e : r

In [12]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o',
           'o':'v','e':'r'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the ***t* ** *tre**th e**a* e***a*** that the *o*er o*
*r*e***h** he*** ** to *e *tro** *ro* har**h***. *r*e*** *a*
**ve ** the *tre**th *e *ee* *he* ***e *et* to**h.
** a***t*o*, *eo**e e**re** ***t* thro**h *ove,*a****, *r*e***, a** other*
*hare *tor*e* o* ****** *o**o* *ro*** **th *o*eo*e.there are *a**
*e*e**t* o* *ta*******te* ** to**h t**e*, a* *t he*** to
*a*e *ha**e***** **t*at*o** **th *o*ra*e. the ***orta**e o* *ta**** ***te*
ha* *tr*** a *hor* a*o** *a** *eo**e thro**ho*t h**tor*.*ee**** *e*or*e*
o* *hat *e have a**o*****he* thro**ho*t h**tor* *a* he** **
*ee ho* ****v***a** a** *o*****t*e* have *er*evere* thro**h to**h t**e*
a** **to a *r**hter **t*re.
```

The 5th word of 4th line (thro **h) could be nothing except** through**. So, the mapping:

b:u

u:g

In [13]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o',
           'o':'v','e':'r','b':'u','u':'g'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the u**t* ** *tre*gth e**a* e***a*** that the *o*er o*
*r*e***h** he*** u* to *e *tro*g *ro* har**h***. *r*e*** *a*
g*ve u* the *tre*gth *e *ee* *he* ***e get* tough.
** a***t*o*, *eo**e e**re** u**t* through *ove,*a****, *r*e***, a** other*
*hare *tor*e* o* ******g *o**o* grou** **th *o*eo*e.there are *a**
*e*e**t* o* *ta***gu**te* ** tough t**e*, a* *t he*** to
*a*e *ha**e*g**g **tuat*o** **th *ourage. the ***orta**e o* *ta***g u**te*
ha* *tru** a *hor* a*o*g *a** *eo**e throughout h**tor*.*ee***g *e*or*e*
o* *hat *e have a**o*****he* throughout h**tor* *a* he** u*
*ee ho* ****v**ua** a** *o**u**t*e* have *er*evere* through tough t**e*
a** **to a *r*ghter *uture.
```

The 3rd word of 1st ine (*tre* gth) would be nothing except **strength**. The mapping:

f:s

s:n

In [14]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
  'e':'r','b':'u','u':'g','f':'s','s':'n'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the un*t* *s strength essa* e***a*ns that the *o*er o*
*r*en*sh** he**s us to *e strong *ro* har*sh**s. *r*en*s *an
g*ve us the strength *e nee* *hen ***e gets tough.
*n a***t*on, *eo**e e**ress un*t* through *ove,*a****, *r*en*s, an* others
share stor*es o* **n**ng *o**on groun* **th so*eone.there are *an*
*ene**ts o* sta**ngun*te* *n tough t**es, as *t he**s to
*a*e *ha**eng*ng s*tuat*ons **th *ourage. the ***ortan*e o* sta**ng un*te*
has stru** a *hor* a*ong *an* *eo**e throughout h*stor*.*ee**ng *e*or*es
o* *hat *e have a**o****she* throughout h*stor* *an he** us
see ho* *n**v**ua*s an* *o**un*t*es have *ersevere* through tough t**es
an* *nto a *r*ghter *uture.
```

For the word '*s' the most frequent two word is 'is'. The mapping:

v : i

In [15]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
           'e':'r','b':'u','u':'g','f':'s','s':'n','v':'i'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unit* is strength essa* e***ains that the *o*er o*
*rien*shi* he**s us to *e strong *ro* har*shi*s. *rien*s *an
give us the strength *e nee* *hen *i*e gets tough.
in a**ition, *eo**e e**ress unit* through *ove,*a*i**, *rien*s, an* others
share stories o* *in*ing *o**on groun* *ith so*eone.there are *an*
*ene*its o* sta*ingunite* in tough ti*es, as it he**s to
*a*e *ha**enging situations *ith *ourage. the i**ortan*e o* sta*ing unite*
has stru** a *hor* a*ong *an* *eo**e throughout histor*.*ee*ing *e*ories
o* *hat *e have a**o***ishe* throughout histor* *an he** us
see ho* in*ivi*ua*s an* *o**unities have *ersevere* through tough ti*es
an* into a *righter *uture.
```

The 2nd word of first line (unit*) would be 'unity'. The mapping:

l : y

In [16]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
           'e':'r','b':'u','u':'g','f':'s','s':'n','v':'i','l':'y'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay e***ains that the *o*er o*
*rien*shi* he**s us to *e strong *ro* har*shi*s. *rien*s *an
give us the strength *e nee* *hen *i*e gets tough.
in a**ition, *eo**e e**ress unity through *ove,*a*i*y, *rien*s, an* others
share stories o* *in*ing *o**on groun* *ith so*eone.there are *any
*ene*its o* stayingunite* in tough ti*es, as it he**s to
*a*e *ha**enging situations *ith *ourage. the i**ortan*e o* staying unite*
has stru** a *hor* a*ong *any *eo**e throughout history.*ee*ing *e*ories
o* *hat *e have a**o***ishe* throughout history *an he** us
see ho* in*ivi*ua*s an* *o**unities have *ersevere* through tough ti*es
an* into a *righter *uture.
```

The very last word of the text (*uture) would be 'future'. The mapping:

a : f

In [17]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
           'e':'r','b':'u','u':'g','f':'s','s':'n','v':'i',
           'l':'y','a':'f'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay e***ains that the *o*er of
frien*shi* he**s us to *e strong fro* har*shi*s. frien*s *an
give us the strength *e nee* *hen *ife gets tough.
in a**ition, *eo**e e**ress unity through *ove,fa*i*y, frien*s, an* others
share stories of fin*ing *o**on groun* *ith so*eone.there are *any
*enefits of stayingunite* in tough ti*es, as it he**s to
fa*e *ha**enging situations *ith *ourage. the i**ortan*e of staying unite*
has stru** a *hor* a*ong *any *eo**e throughout history.*ee*ing *e*ories
of *hat *e have a**o***ishe* throughout history *an he** us
see ho* in*ivi*ua*s an* *o**unities have *ersevere* through tough ti*es
an* into a *righter future.
```

In the 2nd line the phrase 'to *e strong' would be 'to be strong' . The mapping:

i : b

In [18]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
           'e':'r','b':'u','u':'g','f':'s','s':'n','v':'i',
           'l':'y','a':'f','i':'b'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay e***ains that the *o*er of
frien*shi* he**s us to be strong fro* har*shi*s. frien*s *an
give us the strength *e nee* *hen *ife gets tough.
in a**ition, *eo**e e**ress unity through *ove,fa*i*y, frien*s, an* others
share stories of fin*ing *o**on groun* *ith so*eone.there are *any
benefits of stayingunite* in tough ti*es, as it he**s to
fa*e *ha**enging situations *ith *ourage. the i**ortan*e of staying unite*
has stru** a *hor* a*ong *any *eo**e throughout history.*ee*ing *e*ories
of *hat *e have a**o***ishe* throughout history *an he** us
see ho* in*ivi*ua*s an* *o**unities have *ersevere* through tough ti*es
an* into a brighter future.
```

The 7th word of 4rd line(*ife) would be life. The mapping:

y : l

In [19]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
           'e':'r','b':'u','u':'g','f':'s','s':'n','v':'i','l':'y',
           'a':'f','i':'b','y':'l'}
algo(ciphertext, mapping, words_per_line)
freq_analyze(ciphertext)
```

Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay e**lains that the *o*er of
frien*shi* hel*s us to be strong fro* har*shi*s. frien*s *an
give us the strength *e nee* *hen life gets tough.
in a**ition, *eo*le e**ress unity through love,fa*ily, frien*s, an* others
share stories of fin*ing *o**on groun* *ith so*eone.there are *any
benefits of stayingunite* in tough ti*es, as it hel*s to
fa*e *hallenging situations *ith *ourage. the i**ortan*e of staying unite*
has stru** a *hor* a*ong *any *eo*le throughout history.*ee*ing *e*ories
of *hat *e have a**o**lishe* throughout history *an hel* us
see ho* in*ivi*uals an* *o**unities have *ersevere* through tough ti*es
an* into a brighter future.
Character frequencies in the ciphertext (descending order):
'd': 65
'g': 50
'v': 45
'f': 43
'h': 42
't': 41
's': 40
'e': 32
'm': 32
'b': 24
'u': 22
'x': 19
'c': 16
'z': 16
'a': 15
'y': 13
'q': 12
'l': 10
'j': 8

```
'o': 6
'i': 3
'k': 2
'p': 2
```

The 5th word of 5th line (groun*) would be nothing but 'ground'. The mapping:

x : d

In [20]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v','e':'r',
           'b':'u','u':'g','f':'s','s':'n','v':'i','l':'y','a':'f',
           'i':'b','y':'l','x':'d'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay e**lains that the *o*er of
friendshi* hel*s us to be strong fro* hardshi*s. friends *an
give us the strength *e need *hen life gets tough.
in addition, *eo*le e**ress unity through love,fa*ily, friends, and others
share stories of finding *o**on ground *ith so*eone.there are *any
benefits of stayingunited in tough ti*es, as it hel*s to
fa*e *hallenging situations *ith *ourage. the i**ortan*e of staying united
has stru** a *hord a*ong *any *eo*le throughout history.*ee*ing *e*ories
of *hat *e have a**o**lished throughout history *an hel* us
see ho* individuals and *o**unities have *ersevered through tough ti*es
and into a brighter future.
```

The 1st word of 2nd line (friendshi*) would be friendship. The mapping:

c : p

In [21]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
           'e':'r','b':'u','u':'g','f':'s','s':'n','v':'i','l':'y',
           'a':'f','i':'b','y':'l','x':'d','c':'p'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay e*plains that the po*er of
friendship helps us to be strong fro* hardships. friends *an
give us the strength *e need *hen life gets tough.
in addition, people e*press unity through love,fa*ily, friends, and others
share stories of finding *o**on ground *ith so*eone.there are *any
benefits of stayingunited in tough ti*es, as it helps to
fa*e *hallenging situations *ith *ourage. the i*portan*e of staying united
has stru** a *hord a*ong *any people throughout history.*eeping *e*ories
of *hat *e have a**o*plished throughout history *an help us
see ho* individuals and *o**unities have persevered through tough ti*es
and into a brighter future.
```

The word 'e *plains*' *and* '*po* er' in the first line would be 'explains' and 'power' respectively. The mapping:

k : x

j : w

In [22]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
           'e':'r','b':'u','u':'g','f':'s','s':'n','v':'i','l':'y',
           'a':'f','i':'b','y':'l','x':'d','c':'p','k':'x','j':'w'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay explains that the power of
friendship helps us to be strong fro* hardships. friends *an
give us the strength we need when life gets tough.
in addition, people express unity through love,fa*ily, friends, and others
share stories of finding *o**on ground with so*eone.there are *any
benefits of stayingunited in tough ti*es, as it helps to
fa*e *hallenging situations with *ourage. the i*portan*e of staying united
has stru** a *hord a*ong *any people throughout history.*eeping *e*ories
of what we have a**o*plished throughout history *an help us
see how individuals and *o**unities have persevered through tough ti*es
and into a brighter future.
```

Now we can see that the cipher is quite simple to understand the rest of the mapping. The word 'fro ' *and* ' an' in the 2nd line would be 'from' and 'can' respectively. The mapping:

z : m

q : c

In [23]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v','e':'r',
           'b':'u','u':'g','f':'s','s':'n','v':'i','l':'y','a':'f',
           'i':'b','y':'l','x':'d','c':'p','k':'x','j':'w','z':'m','q':'c'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay explains that the power of
friendship helps us to be strong from hardships. friends can
give us the strength we need when life gets tough.
in addition, people express unity through love,family, friends, and others
share stories of finding common ground with someone.there are many
benefits of stayingunited in tough times, as it helps to
face challenging situations with courage. the importance of staying united
has struc* a chord among many people throughout history.*eeping memories
of what we have accomplished throughout history can help us
see how individuals and communities have persevered through tough times
and into a brighter future.
```

The rest two mapping in the word 'struc ' *and* ' eeping' would be 'struck' ans 'keeping' respectively. The mapping:

p : k

In [24]:
```
mapping = {'m':'a','g':'t','t':'h','d':'e','h':'o','o':'v',
           'e':'r','b':'u','u':'g','f':'s','s':'n','v':'i','l':'y',
           'a':'f','i':'b','y':'l','x':'d','c':'p','k':'x','j':'w',
           'z':'m','q':'c','p':'k'}
algo(ciphertext, mapping, words_per_line)
```

```
Ciphertext:
gtd bsvgl vf fgedsugt dffml dkcymvsf gtmg gtd chjde ha
aevdsxftvc tdycf bf gh id fgehsu aehz tmexftvcf. aevdsxf qms
uvod bf gtd fgedsugt jd sddx jtds yvad udgf ghbut.
vs mxxvgvhs, cdhcyd dkcedff bsvgl gtehbut yhod,amzvyl, aevdsxf, msx hgtdef
ftmed fghevdf ha avsxvsu qhzzhs uehbsx jvgt fhzdhsd.gtded med zmsl
idsdavgf ha fgmlvsubsvgdx vs ghbut gvzdf, mf vg tdycf gh
amqd qtmyydsuvsu fvgbmgvhsf jvgt qhbemud. gtd vzchegmsqd ha fgmlvsu bsvgdx
tmf fgebqp m qthex mzhsu zmsl cdhcyd gtehbuthbg tvfghel.pddcvsu zdzhevdf
ha jtmg jd tmod mqqhzcyvftdx gtehbuthbg tvfghel qms tdyc bf
fdd thj vsxvovxbmyf msx qhzzbsvgvdf tmod cdefdodedx gtehbut ghbut gvzdf
msx vsgh m ievutgde abgbed.

Decrypted Text:
the unity is strength essay explains that the power of
friendship helps us to be strong from hardships. friends can
give us the strength we need when life gets tough.
in addition, people express unity through love,family, friends, and others
share stories of finding common ground with someone.there are many
benefits of stayingunited in tough times, as it helps to
face challenging situations with courage. the importance of staying united
has struck a chord among many people throughout history.keeping memories
of what we have accomplished throughout history can help us
see how individuals and communities have persevered through tough times
and into a brighter future.
```

Thus we decrypted our cipher into plaintext. And the plaintext is:

the unity is strength essay explains that the power of friendship helps us to be strong from hardships. friends can give us the strength we need when life gets tough. in addition, people express unity through love,family, friends, and others share stories of finding common ground with someone.there are many benefits of stayingunited in tough times, as it helps to face challenging situations with courage. the importance of staying united has struck a chord among many people throughout history.keeping memories of what we have accomplished throughout history can help us see how individuals and communities have persevered through tough times and into a brighter future.

2.(b)

exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom. xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm

ucamw zs kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

In [25]:
```
# Provided ciphertext
ciphertext_b= """exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg lok
xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok
plokgm upm wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz kz
ozu.fztmzjmt, xs czi pljm l aglo lok czi elou uz xfagmfmou xu xo czit gxsm upmo
czi ommk kxwqxagxom. xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc rtxoh
wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs kxwqxagxom, upmo upmc
ltm hmomtlggc zs uez ucamw. sxtwu zom xw xokiqmk kxwqxagxom lok upm
wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom xw wzfmupxoh uplu zupmtw
ulihpu iw zt em gmlto rc wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo
lok em gmlto xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs fzuxjluxzo
lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc
wqpmkigm exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.
"""
```

In [26]:
```
mapping = {}
algo(ciphertext_b, mapping, words_per_line)
freq_analyze(ciphertext_b)
```

Ciphertext:
```
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.
```

Decrypted Text:
```
******* **********, *** **** ** * ****** **** ****** ****
*** ********.****, * *********** ****** *** ******* *** ****** ***
********* ** ****** ** * ************* *** **** ***** ***
** ***.********, ** *** **** * **** *** *** ****
** ********* ** ** **** **** **** *** **** **********.
** ***** ****** **** *** *** ** ****** *** **********
***** ******* ** **** ****.** **** ***** *** ***** **
**********, **** **** *** ********* ** *** *****. ***** ***
** ******* ********** *** *** ****** *** ** ****-**********.******* **********
** ********* **** ****** ****** ** ** ** ***** **
****** ******. ***** ****-********** ***** **** ****** *** ** *****
** ** *** *** ****. ****-********** ******** * *** **
********** *** ******* **** ******.***** ***, ********* **** ***** ********
******* *** ******* ** **** **** ** ***** ***********.
```
Character frequencies in the ciphertext (descending order):
```
'x': 74
'm': 73
'z': 57
'o': 55
'u': 52
'w': 47
'g': 44
'l': 43
'k': 32
'p': 26
't': 26
'i': 22
'q': 22
```

```
'a': 20
's': 20
'c': 17
'e': 13
'f': 12
'h': 9
'r': 6
'j': 6
'n': 3
'y': 1
```

The most frequent in the text is 'x'. The most frequent word in english is 'e'. Replace 'x' with 'e'. So, the mapping:

x : e

In [27]:
```
mapping = {'x':'e'}
algo(ciphertext_b, mapping, words_per_line)
freq_analyze(ciphertext_b)
```

Ciphertext:
```
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.
```

Decrypted Text:
```
*e***** *e**e**e**, *** *e** ** * ****** *e** ****** ****
*** e****e**.****, * *e**e**e*** ****** *** ******* *** ****** ***
*e****e** ** *e*e** e* * ****e**e***** *** **** ***** ***
** ***.********, e* *** **** * **** *** *** ****
** e******** e* e* **** *e** **** *** **** *e**e**e**.
e* ***** **e*** **** *** *** ** ****** *** ***e******
**e** ******* ** **** *e**.e* **** ***** *** ***** **
*e**e**e**, **** **** *** ********* ** *** *****. *e*** ***
e* e****** *e**e**e** *** *** ****** *** e* ****-*e**e**e**.e****** *e**e**e**
e* ******e** **** ****** ****** ** ** ** ***** **
***e** ******. **e** ****-*e**e**e** ***** **** *e**e* *** ** *****
e* ** *** *** ****. ****-*e**e**e** ****e*** * *** **
***e***e** *** ******* **** ******.***** ***, ******e** **** **e** ********
*e***** *** *e***** e* **** **** ** **e** *e**e**e***.
```
Character frequencies in the ciphertext (descending order):
```
'x': 74
'm': 73
'z': 57
'o': 55
'u': 52
'w': 47
'g': 44
'l': 43
'k': 32
'p': 26
't': 26
'i': 22
'q': 22
```

```
'a': 20
's': 20
'c': 17
'e': 13
'f': 12
'h': 9
'r': 6
'j': 6
'n': 3
'y': 1
```

Replace the signle letter word with 'a'. As it is the most frequent english single letter word. So, the mapping:

l : a

In [28]:
```
mapping = {'x':'e','l':'a'}
algo(ciphertext_b, mapping, words_per_line)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
*e***** *e**e**e**, *** *e** ** a ****** *e** ****** ****
a** e*a**e**.a***, a *e**e**e*** ****** *a* ******* a** *a**** ***
*e**a*e** ** *e*e** e* a ****e**e*a*** *a* **a* ***** ***
** ***.********, e* *** *a** a **a* a** *** *a**
** e******** e* e* **** *e** **** *** **** *e**e**e**.
e* *a*** **e*** *a** *** *** ** *a**** a** ***e*a****
**e** ******* ** **** *e**.e* *a** a**** *** ***** **
*e**e**e**, **** **** a** *****a*** ** *** *****. *e*** ***
e* e****** *e**e**e** a** *** ****** *** e* ****-*e**e**e**.e****** *e**e**e**
e* ******e** **a* ****** *a**** ** ** ** **a** **
***e** ******. **e** ****-*e**e**e** ***** **** *e**e* a** ** **a**
e* ** *** *** ****. ****-*e**e**e** ****e*** a *** **
***e*a*e** a** ******* **** ******.a**** a**, ******e** **** *ae** ********
*e***** a** *e**a** e* a*** *a** ** **e** *e**e**e***.
```

I observed there are many 'upm' in the text. So i replace upm with the, most frequent three letter word. The mapping:

u : t

p : h

m : e

So we cannot replace 'x' with 'e'. Discard the mapping.

In [29]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e'}
algo(ciphertext_b, mapping, words_per_line)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
**th**t *********e, the ***e ** a *e**** **** *e***e ****
a** **a*t**e.a***, a *********e* *e**** *a* ***t*** a** ha***e the
**t*at*** ** ****** ** a ***h**t**ate* *a* tha* th**e *h*
** **t.***e**e*, ** *** ha*e a **a* a** *** *a*t
t* ****e*e*t *t ** **** ***e the* *** *ee* *********e.
*t *a*e* th**** ea** *** *** t* ha***e a** **t**ate**
***** ****e** t* **** ***e.** ta** a***t the t**e* **
*********e, the* the* a*e *e*e*a*** ** t** t**e*. ****t **e
** *****e* *********e a** the *e**** **e ** *e**-*********e.*****e* *********e
** ***eth*** that *the** ta**ht ** ** *e *ea** **
*ee*** *the**. *h**e *e**-*********e ***e* **** **th** a** *e *ea**
*t ** *** *** *e**. *e**-*********e *e****e* a **t **
**t**at*** a** ******t **** *the**.a***e a**, ********* **** *a*** **he***e
**th**t a** ***ta*e ** a*** *a*t ** *e*** *********e*.
```

The 4th word of 8th line (a*e) would be are. The mapping:

t : r

In [30]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r'}
algo(ciphertext_b, mapping, words_per_line)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
**th**t *********e, the ***e ** a *er*** **** *e***e ****
a** **a*t**e.a***, a *********e* *er*** *a* ***tr** a** ha***e the
**t*at*** ** ****** ** a ***h**t**ate* *a* tha* th**e *h*
** **t.**re**er, ** *** ha*e a **a* a** *** *a*t
t* ****e*e*t *t ** ***r ***e the* *** *ee* *********e.
*t *a*e* th**** ea** **r *** t* ha***e a** **t**ate**
*r*** ****e** t* ***r ***e.** ta** a***t the t**e* **
*********e, the* the* are *e*era*** ** t** t**e*. **r*t **e
** *****e* *********e a** the *e**** **e ** *e**-*********e.*****e* *********e
** ***eth*** that *ther* ta**ht ** *r *e *ear* **
*ee*** *ther*. *h**e *e**-*********e ***e* *r** **th** a** *e *ear*
*t ** **r *** *e**. *e**-*********e re***re* a **t **
**t**at*** a** *****rt *r** *ther*.a***e a**, ********* ***r *a*** **he***e
**th**t a** ***ta*e ** a*** *art ** *e*** *********e*.
```

The 6th word of 4th line (ha*e) would be have. the mapping:

j : v

In [31]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v'}
algo(ciphertext_b, mapping, words_per_line)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
**th**t *********e, the ***e ** a *er*** **** *e***e ****
a** **a*t*ve.a***, a *********e* *er*** *a* ***tr** a** ha***e the
**t*at*** ** **v*** ** a ***h**t**ate* *a* tha* th**e *h*
** **t.**re*ver, ** *** have a **a* a** *** *a*t
t* ****e*e*t *t ** ***r ***e the* *** *ee* *********e.
*t *a*e* th**** ea** **r *** t* ha***e a** **t**ate**
*r*** ****e** t* ***r ***e.** ta** a***t the t**e* **
*********e, the* the* are *e*era*** ** t** t**e*. **r*t **e
** *****e* *********e a** the *e**** **e ** *e**-*********e.*****e* *********e
** ***eth*** that *ther* ta**ht ** *r *e *ear* **
*ee*** *ther*. *h**e *e**-*********e ***e* *r** **th** a** *e *ear*
*t ** **r *** *e**. *e**-*********e re***re* a **t **
**t*vat*** a** *****rt *r** *ther*.a**ve a**, ********* ***r *a*** **he***e
**th**t a** ***ta*e ** a*** *art ** *e*** *********e*.
```

The word tha* in the 3rd line would be than as if it was that then we see earlier. The mapping:

o : n

In [32]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n'}
algo(ciphertext_b, mapping, words_per_line)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
**th**t ********ne, the ***e ** a *er**n **** *e***e ****
an* *na*t*ve.a***, a ********ne* *er**n *an **ntr** an* han**e the
**t*at**n ** **v*n* *n a ***h**t**ate* *a* than th**e *h*
** n*t.**re*ver, ** *** have a **an an* *** *ant
t* ****e*ent *t *n ***r ***e then *** nee* ********ne.
*t *a*e* th*n** ea** **r *** t* han**e an* **t**ate**
*r*n* ****e** t* ***r ***e.** ta** a***t the t**e* **
********ne, then the* are *enera*** ** t** t**e*. **r*t *ne
** *n***e* ********ne an* the *e**n* *ne ** *e**-********ne.*n***e* ********ne
** ***eth*n* that *ther* ta**ht ** *r *e *earn **
*ee*n* *ther*. *h**e *e**-********ne ***e* *r** **th*n an* *e *earn
*t *n **r **n *e**. *e**-********ne re***re* a **t **
**t*vat**n an* *****rt *r** *ther*.a**ve a**, *******n* ***r *a*** **he***e
**th**t an* ***ta*e ** a*** *art ** *e*n* ********ne*.
```

The word 'nee*' in the 5th line would be 'need'. The mapping:

k : d

In [33]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n','k':'d'}
algo(ciphertext_b, mapping, words_per_line)
freq_analyze(ciphertext_b)
```

Ciphertext:
```
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.
```

Decrypted Text:
```
**th**t d*******ne, the ***e ** a *er**n **** *e***e d***
and *na*t*ve.a***, a d*******ned *er**n *an **ntr** and hand*e the
**t*at**n ** **v*n* *n a ***h**t**ated *a* than th**e *h*
d* n*t.**re*ver, ** *** have a **an and *** *ant
t* ****e*ent *t *n ***r ***e then *** need d*******ne.
*t *a*e* th*n** ea** **r *** t* hand*e and **t**ate**
*r*n* ****e** t* ***r ***e.** ta** a***t the t**e* **
d*******ne, then the* are *enera*** ** t** t**e*. **r*t *ne
** *nd**ed d*******ne and the *e**nd *ne ** *e**-d*******ne.*nd**ed d*******ne
** ***eth*n* that *ther* ta**ht ** *r *e *earn **
*ee*n* *ther*. *h**e *e**-d*******ne ***e* *r** **th*n and *e *earn
*t *n **r **n *e**. *e**-d*******ne re***re* a **t **
**t*vat**n and *****rt *r** *ther*.a**ve a**, *******n* ***r da*** **hed**e
**th**t an* ***ta*e ** a*** *art ** *e*n* d*******ned.
```
Character frequencies in the ciphertext (descending order):
```
'x': 74
'm': 73
'z': 57
'o': 55
'u': 52
'w': 47
'g': 44
'l': 43
'k': 32
'p': 26
't': 26
'i': 22
'q': 22
```

```
'a': 20
's': 20
'c': 17
'e': 13
'f': 12
'h': 9
'r': 6
'j': 6
'n': 3
'y': 1
```

the 'd *n* t' in the 5th line would be 'do not'. So, the mapping:

z : o

In [34]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n',
           'k':'d','z':'o'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
**tho*t d*******ne, the ***e o* a *er*on **** *e*o*e d***
and *na*t*ve.a**o, a d*******ned *er*on *an *ontro* and hand*e the
**t*at*on o* **v*n* *n a *o*h**t**ated *a* than tho*e *ho
do not.*oreover, ** *o* have a **an and *o* *ant
to ****e*ent *t *n *o*r ***e then *o* need d*******ne.
*t *a*e* th*n** ea** *or *o* to hand*e and **t**ate**
*r*n* ****e** to *o*r ***e.** ta** a*o*t the t**e* o*
d*******ne, then the* are *enera*** o* t*o t**e*. **r*t one
** *nd**ed d*******ne and the *e*ond one ** *e**-d*******ne.*nd**ed d*******ne
** *o*eth*n* that other* ta**ht ** or *e *earn **
*ee*n* other*. *h**e *e**-d*******ne *o*e* *ro* **th*n and *e *earn
*t on o*r o*n *e**. *e**-d*******ne re***re* a *ot o*
*ot*vat*on and ****ort *ro* other*.a*ove a**, *o**o**n* *o*r da*** **hed**e
**tho*t an* ***ta*e ** a**o *art o* *e*n* d*******ned.
```

The phrase 'tho*e* ho' of 3rd line would be 'those who'. The mapping:

w:s

e:w

In [35]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n',
           'k':'d','z':'o','w':'s','e':'w'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
w*tho*t d*s*****ne, the ***e o* a *erson w*** *e*o*e d***
and *na*t*ve.a*so, a d*s*****ned *erson *an *ontro* and hand*e the
s*t*at*on o* **v*n* *n a so*h*st**ated wa* than those who
do not.*oreover, ** *o* have a **an and *o* want
to ****e*ent *t *n *o*r ***e then *o* need d*s*****ne.
*t *a*es th*n*s eas* *or *o* to hand*e and **t**ate**
*r*n* s***ess to *o*r ***e.** ta** a*o*t the t**es o*
d*s*****ne, then the* are *enera*** o* two t**es. **rst one
*s *nd**ed d*s*****ne and the se*ond one *s se**-d*s*****ne.*nd**ed d*s*****ne
*s so*eth*n* that others ta**ht *s or we *earn **
see*n* others. wh**e se**-d*s*****ne *o*es *ro* w*th*n and we *earn
*t on o*r own se**. se**-d*s*****ne re***res a *ot o*
*ot*vat*on and s***ort *ro* others.a*ove a**, *o**ow*n* *o*r da*** s*hed**e
w*tho*t an* **sta*e *s a*so *art o* *e*n* d*s*****ned.
```

The word 'a*so' would be 'also'. So, the mapping:

g : l

In [36]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n',
           'k':'d','z':'o','w':'s','e':'w','g':'l'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
w*tho*t d*s***l*ne, the l**e o* a *erson w*ll *e*o*e d*ll
and *na*t*ve.also, a d*s***l*ned *erson *an *ontrol and handle the
s*t*at*on o* l*v*n* *n a so*h*st**ated wa* than those who
do not.*oreover, ** *o* have a *lan and *o* want
to ***le*ent *t *n *o*r l**e then *o* need d*s***l*ne.
*t *a*es th*n*s eas* *or *o* to handle and *lt**atel*
*r*n* s***ess to *o*r l**e.** tal* a*o*t the t**es o*
d*s***l*ne, then the* are *enerall* o* two t**es. **rst one
*s *nd**ed d*s***l*ne and the se*ond one *s sel*-d*s***l*ne.*nd**ed d*s***l*ne
*s so*eth*n* that others ta**ht *s or we learn **
see*n* others. wh*le sel*-d*s***l*ne *o*es *ro* w*th*n and we learn
*t on o*r own sel*. sel*-d*s***l*ne re***res a lot o*
*ot*vat*on and s***ort *ro* others.a*ove all, *ollow*n* *o*r da*l* s*hed*le
w*tho*t an* **sta*e *s also *art o* *e*n* d*s***l*ned.
```

The phrase 'a *erson w* ll' would be 'a person will'. So, the mapping:

a:p

x:i

We got here the most frequent letter 'x'. That's amazing.

In [37]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n',
           'k':'d','z':'o','w':'s','e':'w','g':'l','a':'p','x':'i'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
witho*t dis*ipline, the li*e o* a person will *e*o*e d*ll
and ina*tive.also, a dis*iplined person *an *ontrol and handle the
sit*ation o* livin* in a sophisti*ated wa* than those who
do not.*oreover, i* *o* have a plan and *o* want
to i*ple*ent it in *o*r li*e then *o* need dis*ipline.
it *a*es thin*s eas* *or *o* to handle and *lti*atel*
*rin* s***ess to *o*r li*e.i* tal* a*o*t the t*pes o*
dis*ipline, then the* are *enerall* o* two t*pes. *irst one
is ind**ed dis*ipline and the se*ond one is sel*-dis*ipline.ind**ed dis*ipline
is so*ethin* that others ta**ht *s or we learn **
seein* others. while sel*-dis*ipline *o*es *ro* within and we learn
it on o*r own sel*. sel*-dis*ipline re**ires a lot o*
*otivation and s*pport *ro* others.a*ove all, *ollowin* *o*r dail* s*hed*le
witho*t an* *ista*e is also part o* *ein* dis*iplined.
```

The phrase 'witho *t dis* ipline' would be 'without discipline'. The mapping will be:

i:o

q:c

In [38]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n','k':'d',
           'z':'o','w':'s','e':'w','g':'l','a':'p','x':'i','i':'o','q':'c'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
withoot discipline, the li*e o* a person will *eco*e doll
and inactive.also, a disciplined person can control and handle the
sitoation o* livin* in a sophisticated wa* than those who
do not.*oreover, i* *oo have a plan and *oo want
to i*ple*ent it in *oor li*e then *oo need discipline.
it *a*es thin*s eas* *or *oo to handle and olti*atel*
*rin* soccess to *oor li*e.i* tal* a*oot the t*pes o*
discipline, then the* are *enerall* o* two t*pes. *irst one
is indoced discipline and the second one is sel*-discipline.indoced discipline
is so*ethin* that others tao*ht os or we learn **
seein* others. while sel*-discipline co*es *ro* within and we learn
it on oor own sel*. sel*-discipline re*oires a lot o*
*otivation and sopport *ro* others.a*ove all, *ollowin* *oor dail* schedole
withoot an* *ista*e is also part o* *ein* disciplined.

Duplicate mappings found:
o: z, i
```

The last word of 13th line is 'schedole'. But there is no such word in english dictonary. The word would be 'schedule'. The mapping:

i : u

In [39]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n','k':'d','z':'o',
           'w':'s','e':'w','g':'l','a':'p','x':'i','i':'u','q':'c'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
without discipline, the li*e o* a person will *eco*e dull
and inactive.also, a disciplined person can control and handle the
situation o* livin* in a sophisticated wa* than those who
do not.*oreover, i* *ou have a plan and *ou want
to i*ple*ent it in *our li*e then *ou need discipline.
it *a*es thin*s eas* *or *ou to handle and ulti*atel*
*rin* success to *our li*e.i* tal* a*out the t*pes o*
discipline, then the* are *enerall* o* two t*pes. *irst one
is induced discipline and the second one is sel*-discipline.induced discipline
is so*ethin* that others tau*ht us or we learn **
seein* others. while sel*-discipline co*es *ro* within and we learn
it on our own sel*. sel*-discipline re*uires a lot o*
*otivation and support *ro* others.a*ove all, *ollowin* *our dail* schedule
without an* *ista*e is also part o* *ein* disciplined.
```

In the first line 'without discipline, the li*e o* a person will' would be 'without discipline, the life of a person will'. The mapping:

s:f

In [40]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n','k':'d',
           'z':'o','w':'s','e':'w','g':'l','a':'p','x':'i','i':'u','q':'c','s':'f'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
without discipline, the life of a person will *eco*e dull
and inactive.also, a disciplined person can control and handle the
situation of livin* in a sophisticated wa* than those who
do not.*oreover, if *ou have a plan and *ou want
to i*ple*ent it in *our life then *ou need discipline.
it *a*es thin*s eas* for *ou to handle and ulti*atel*
*rin* success to *our life.if tal* a*out the t*pes of
discipline, then the* are *enerall* of two t*pes. first one
is induced discipline and the second one is self-discipline.induced discipline
is so*ethin* that others tau*ht us or we learn **
seein* others. while self-discipline co*es fro* within and we learn
it on our own self. self-discipline re*uires a lot of
*otivation and support fro* others.a*ove all, followin* *our dail* schedule
without an* *ista*e is also part of *ein* disciplined.
```

The phrase 'a*ove all, followin our dail* schedule' would be 'above all, following your daily schedule' in the 2nd last line. The mapping:

r : b

h : g

c : y

In [41]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n','k':'d','z':'o',
           'w':'s','e':'w','g':'l','a':'p','x':'i','i':'u','q':'c','s':'f',
           'r':'b','h':'g','c':'y'}
```

```
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
without discipline, the life of a person will beco*e dull
and inactive.also, a disciplined person can control and handle the
situation of living in a sophisticated way than those who
do not.*oreover, if you have a plan and you want
to i*ple*ent it in your life then you need discipline.
it *a*es things easy for you to handle and ulti*ately
bring success to your life.if tal* about the types of
discipline, then they are generally of two types. first one
is induced discipline and the second one is self-discipline.induced discipline
is so*ething that others taught us or we learn by
seeing others. while self-discipline co*es fro* within and we learn
it on our own self. self-discipline re*uires a lot of
*otivation and support fro* others.above all, following your daily schedule
without any *ista*e is also part of being disciplined.
```

The word 'beco*e' in the first line would be 'become'. The mapping:

f:m

In [42]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n','k':'d',
           'z':'o','w':'s','e':'w','g':'l','a':'p','x':'i','i':'u','q':'c',
           's':'f','r':'b','h':'g','c':'y','f':'m'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
without discipline, the life of a person will become dull
and inactive.also, a disciplined person can control and handle the
situation of living in a sophisticated way than those who
do not.moreover, if you have a plan and you want
to implement it in your life then you need discipline.
it ma*es things easy for you to handle and ultimately
bring success to your life.if tal* about the types of
discipline, then they are generally of two types. first one
is induced discipline and the second one is self-discipline.induced discipline
is something that others taught us or we learn by
seeing others. while self-discipline comes from within and we learn
it on our own self. self-discipline re*uires a lot of
motivation and support from others.above all, following your daily schedule
without any mista*e is also part of being disciplined.
```

The word 'ma*es' in the 6th line would be 'makes'. The mapping:

n:k

In [43]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v','o':'n',
           'k':'d','z':'o','w':'s','e':'w','g':'l','a':'p','x':'i',
           'i':'u','q':'c','s':'f','r':'b','h':'g','c':'y','f':'m','n':'k'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
without discipline, the life of a person will become dull
and inactive.also, a disciplined person can control and handle the
situation of living in a sophisticated way than those who
do not.moreover, if you have a plan and you want
to implement it in your life then you need discipline.
it makes things easy for you to handle and ultimately
bring success to your life.if talk about the types of
discipline, then they are generally of two types. first one
is induced discipline and the second one is self-discipline.induced discipline
is something that others taught us or we learn by
seeing others. while self-discipline comes from within and we learn
it on our own self. self-discipline re*uires a lot of
motivation and support from others.above all, following your daily schedule
without any mistake is also part of being disciplined.
```

The one left word 're*uries' would be 'requires'. The mapping:

y:q

In [44]:
```
mapping = {'l':'a','u':'t','p':'h','m':'e','t':'r','j':'v',
           'o':'n','k':'d','z':'o','w':'s','e':'w','g':'l','a':'p',
           'x':'i','i':'u','q':'c','s':'f','r':'b','h':'g','c':'y',
           'f':'m','n':'k','y':'q'}
algo(ciphertext_b, mapping, words_per_line)
# freq_analyze(ciphertext_b)
```

```
Ciphertext:
exupziu kxwqxagxom, upm gxsm zs l amtwzo exgg rmqzfm kigg
lok xolquxjm.lgwz, l kxwqxagxomk amtwzo qlo qzoutzg lok plokgm upm
wxuiluxzo zs gxjxoh xo l wzapxwuxqlumk elc uplo upzwm epz
kz ozu.fztmzjmt, xs czi pljm l aglo lok czi elou
uz xfagmfmou xu xo czit gxsm upmo czi ommk kxwqxagxom.
xu flnmw upxohw mlwc szt czi uz plokgm lok iguxflumgc
rtxoh wiqqmww uz czit gxsm.xs ulgn lrziu upm ucamw zs
kxwqxagxom, upmo upmc ltm hmomtlggc zs uez ucamw. sxtwu zom
xw xokiqmk kxwqxagxom lok upm wmqzok zom xw wmgs-kxwqxagxom.xokiqmk kxwqxagxom
xw wzfmupxoh uplu zupmtw ulihpu iw zt em gmlto rc
wmmxoh zupmtw. epxgm wmgs-kxwqxagxom qzfmw stzf exupxo lok em gmlto
xu zo zit zeo wmgs. wmgs-kxwqxagxom tmyixtmw l gzu zs
fzuxjluxzo lok wiaaztu stzf zupmtw.lrzjm lgg, szggzexoh czit klxgc wqpmkigm
exupziu loc fxwulnm xw lgwz altu zs rmxoh kxwqxagxomk.

Decrypted Text:
without discipline, the life of a person will become dull
and inactive.also, a disciplined person can control and handle the
situation of living in a sophisticated way than those who
do not.moreover, if you have a plan and you want
to implement it in your life then you need discipline.
it makes things easy for you to handle and ultimately
bring success to your life.if talk about the types of
discipline, then they are generally of two types. first one
is induced discipline and the second one is self-discipline.induced discipline
is something that others taught us or we learn by
seeing others. while self-discipline comes from within and we learn
it on our own self. self-discipline requires a lot of
motivation and support from others.above all, following your daily schedule
without any mistake is also part of being disciplined.
```

Thus we decrypted our cipher into plaintext. And the plaintext is:

without discipline, the life of a person will become dull and inactive.also, a disciplined person can control and handle the situation of living in a sophisticated way than those who do not.moreover, if you have a plan and you want to implement it in your life then you need discipline. it makes things easy for you to handle and ultimately bring success to your life.if talk about the types of discipline, then they are generally of two types. first one is induced discipline and the second one is self-discipline.induced discipline is something that others taught us or we learn by seeing others. while self-discipline comes from within and we learn it on our own self. self-discipline requires a lot of motivation and support from others.above all, following your daily schedule without any mistake is also part of being disciplined.

2.(c)

AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

In [45]:
```
# Provided ciphertext
ciphertext_c= """AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD
UKUVM. VC HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK CJ
CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.
"""
```

In [46]:
```
mapping = {}
algo(ciphertext_c, mapping, words_per_line)
freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
**** ***** * ****** * **** ** ********* *****. **
****** ** **, * ***** ** *** **** **** *******
** *** ***** *** *** * *****, * ***** ***
*****. *** *** ***, ** *** ****** ** **.
Character frequencies in the ciphertext (descending order):
'Z': 19
'C': 17
'U': 12
'V': 12
'J': 11
'M': 9
'B': 9
'Y': 9
'A': 5
'F': 5
'G': 5
'H': 4
'K': 4
'I': 4
'D': 3
'P': 2
'O': 2
'W': 1
'T': 1
'N': 1
```

At the begining the mapping is empty. We can start from one letter word. there are two one letter word I and A. We can replace V with A. The mapping:

V : A

In [47]:
```
mapping = {'V':'A'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
**** *A*** A ****** A **** ** ********* ***A*. A*
****** ** **, A ***** ** *** A*** **** ****A**
** *** **A** *** *** * **A**, A ***** ***
*****. *** *** ***, A* *** ****** ** **.
```

The most frequent two letter word start with A is AS. So we can try from AS. The mapping:

C : S

In [48]:
```
mapping = {'V':'A','C':'S'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
***S *A**S A *****S A ***S S* ********* ***A*. AS
****** S* **, A *S*** ** S** A*** **S* ****A**
S* S** **A** *** *** * **A**, A ***** **S
**S**. *** S** ***, AS *** ****** S* **.
```

The most frequent two letter word start with S is SO. So we can try from SO. The mapping:

J : O

In [49]:
```
mapping = {'V':'A','C':'S','J':'O'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
***S *A**S A *****S A ***S SO ******** ***A*. AS
****** SO **, A *SOO* ** S** A*O* **S* ****A**
SO S** **A** *** *O* * **A**, A *O*** *OS
**S**. *O* S** ***, AS *** ****** SO **.
```

There are one more one letter word. So, replace U with I. The mapping;

U:I

In [50]:
```
mapping = {'V':'A','C':'S','J':'O','U':'I'}
algo(ciphertext_c, mapping, words_per_line)
freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*I*S *A**S A ***I*S A ***S SO *I******* I*IA*. AS
****** SO **, A *SOO* ** S** A*O* *IS* **I*A**
SO S** **A** I** *O* I **A**, A *O*** *OS
**S**. *O* S** *I*, AS *I* *I**** SO **.
Character frequencies in the ciphertext (descending order):
'Z': 19
'C': 17
'U': 12
'V': 12
'J': 11
'M': 9
'B': 9
'Y': 9
'A': 5
'F': 5
'G': 5
'H': 4
'K': 4
'I': 4
'D': 3
'P': 2
'O': 2
'W': 1
'T': 1
'N': 1
```

I do not get any dependent clue from here. So, using the frequency analysis the most frequest letters are 'Z' and 'C'. The most frequent letters in english language are 'E' and 'T' respectively. So,the mapping:

Z:E

C:T

In [51]:
```
mapping = {'V':'A','C':'S','J':'O','U':'I','Z':'E','C':'T'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*I*T *A**T A **EI*T A *E*T TO *I**E**E* I*IA*. AT
*EE*E* TO *E, A *TOO* ** T*E A*O* *ITE *EI*A**
TO T*E **A*E I** *O* I **A*E, A *O*** *OT
E*TE*. *O* T*E *I*, AT *I* *I**E* TO *E.
```

The word 'T*E' in the 2nd line woud be 'THE'. The mapping:

F : H

In [52]:
```
mapping = {'V':'A','C':'S','J':'O','U':'I','Z':'E','C':'T','F':'H'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*I*T *A*HT A **EI*T A *E*T TO *I**E**E* I*IA*. AT
*EE*E* TO *E, A *TOO* ** THE A*O* *ITE *EI*A**
TO THE **A*E I** *O* I *HA*E, A *O*** *OT
E*TE*. *O* THE *I*, AT *I* *I**E* TO *E.
```

The phrase 'TO *E' in the 2nd and last line would be 'TO BE'. The mapping:

G : B

In [53]:
```
mapping = {'V':'A','C':'S','J':'O','U':'I','Z':'E','C':'T','F':'H','G':'B'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*I*T *A*HT A **EIBT A *E*T TO BI**E**E* I*IA*. AT
*EEBE* TO BE, A *TOO* ** THE A*O* *ITE *EI*A**
TO THE **A*E I** *O* I *HA*E, A *O*** *OT
E*TE*. *O* THE *I*, AT *I* *I**E* TO BE.
```

There is no possible replacement for the word *EEBE*. So I discard to map 'G' to 'B'.Now thehe phrase 'TO *E' in the 2nd and last line would be 'TO ME' because 'ME' is the most frequent two letter word after 'BE'. The mapping:

G:M

In [54]:
```
mapping = {'V':'A','C':'S','J':'O','U':'I','Z':'E','C':'T','F':'H','G':'M'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*I*T *A*HT A **EIMT A *E*T TO MI**E**E* I*IA*. AT
*EEME* TO ME, A *TOO* ** THE A*O* *ITE *EI*A**
TO THE **A*E I** *O* I *HA*E, A *O*** *OT
E*TE*. *O* THE *I*, AT *I* *I**E* TO ME.
```

The very last word of 3rd line (*OT) would be NOT. The mapping:

M:N

In [55]:
```
mapping = {'V':'A','C':'S','J':'O','U':'I','Z':'E','C':'T',
           'F':'H','G':'M','M':'N'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*I*T NA*HT A **EIMT A *ENT TO MIN*E**E* I*IAN. AT
*EEME* TO ME, A *TOO* ** THE A*ON *ITE *EI*AN*
TO THE **A*E IN* *O* I *HA*E, A *O*** NOT
ENTE*. *O* THE *I*, AT *I* *I**E* TO ME.
```

The word 'NA*HT' seems to odd. There is no possible replacement for the word. It would be NIGHT. So, in stead of replacing 'V' with 'A', we replace 'V' with 'I'. The mapping:

V:I

In [56]:
```
mapping = {'V':'I','C':'S','J':'O','U':'I','Z':'E','C':'T',
           'F':'H','G':'M','M':'N'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*I*T NI*HT I **EIMT I *ENT TO MIN*E**E* I*IIN. IT
*EEME* TO ME, I *TOO* ** THE I*ON *ITE *EI*IN*
TO THE **I*E IN* *O* I *HI*E, I *O*** NOT
ENTE*. *O* THE *I*, IT *I* *I**E* TO ME.

Duplicate mappings found:
I: V, U
```

The replacement for the word 'NI*HT' would be 'NIGHT'. The mapping:

K:G

As duplicate mapping found. Discard the mapping **'U':'I'**.

In [57]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H',
           'G':'M','M':'N','K':'G'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
***T NIGHT I **E*MT I *ENT TO M*N*E**E* *G*IN. IT
*EEME* TO ME, I *TOO* ** THE I*ON G*TE *E**ING
TO THE **I*E *N* *O* * *HI*E, I *O*** NOT
ENTE*. *O* THE ***, IT *** ****E* TO ME.
```

The word *ENT in the first line would be WENT. The mapping:

I:W

In [58]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H',
           'G':'M','M':'N','K':'G','I':'W'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
***T NIGHT I **E*MT I WENT TO M*N*E**E* *G*IN. IT
*EEME* TO ME, I *TOO* ** THE I*ON G*TE *E**ING
TO THE **I*E *N* *O* * WHI*E, I *O*** NOT
ENTE*. *O* THE W**, IT W** ****E* TO ME.
```

The word ' GIN' would be 'AGAIN' because the two * is same. The mapping:

U:A

In [59]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H',
           'G':'M','M':'N','K':'G','I':'W','U':'A'}
```

```
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*A*T NIGHT I **EAMT I WENT TO MAN*E**E* AGAIN. IT
*EEME* TO ME, I *TOO* ** THE I*ON GATE *EA*ING
TO THE **I*E AN* *O* A WHI*E, I *O*** NOT
ENTE*. *O* THE WA*, IT WA* *A**E* TO ME.
```

The word 'ENTE*' in the last line would be 'ENTER'. The mapping:

Y:R

In [60]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H',
           'G':'M','M':'N','K':'G','I':'W','U':'A','Y':'R'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
*A*T NIGHT I *REAMT I WENT TO MAN*ER*E* AGAIN. IT
*EEME* TO ME, I *TOO* ** THE IRON GATE *EA*ING
TO THE *RI*E AN* *OR A WHI*E, I *O*** NOT
ENTER. *OR THE WA*, IT WA* *ARRE* TO ME.
```

The phrase ' *A*T NIGHT I *REAMT I WENT TO' in the first line would be 'LAST NIGHT I DREAMT I WENT TO'. So, the mapping:

A:L

H:S

B:D

In [61]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H',
           'G':'M','M':'N','K':'G','I':'W','U':'A','Y':'R','A':'L','H':'S','B':'D'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
LAST NIGHT I DREAMT I WENT TO MANDERLE* AGAIN. IT
SEEMED TO ME, I STOOD ** THE IRON GATE LEADING
TO THE DRI*E AND *OR A WHILE, I *O*LD NOT
ENTER. *OR THE WA*, IT WAS *ARRED TO ME.
```

The word '*OR' in the last line would be 'FOR'. The mapping:

O:F

In [62]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H','G':'M',
           'M':'N','K':'G','I':'W','U':'A','Y':'R','A':'L','H':'S','B':'D','O':'F'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
LAST NIGHT I DREAMT I WENT TO MANDERLE* AGAIN. IT
SEEMED TO ME, I STOOD ** THE IRON GATE LEADING
TO THE DRI*E AND FOR A WHILE, I *O*LD NOT
ENTER. FOR THE WA*, IT WAS *ARRED TO ME.
```

The word 'DRI*E' in the 2nd last line would be 'DRIVE'. The mapping:

W:V

In [63]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H',
           'G':'M','M':'N','K':'G','I':'W','U':'A','Y':'R',
```

```
           'A':'L','H':'S','B':'D','O':'F','W':'V'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
LAST NIGHT I DREAMT I WENT TO MANDERLE* AGAIN. IT
SEEMED TO ME, I STOOD ** THE IRON GATE LEADING
TO THE DRIVE AND FOR A WHILE, I *O*LD NOT
ENTER. FOR THE WA*, IT WAS *ARRED TO ME.
```

The word '*ARRED' in the last line would be 'BARRED'. So, the mapping:

P:B

In [64]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H',
           'G':'M','M':'N','K':'G','I':'W','U':'A','Y':'R','A':'L',
           'H':'S','B':'D','O':'F','W':'V','P':'B'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
LAST NIGHT I DREAMT I WENT TO MANDERLE* AGAIN. IT
SEEMED TO ME, I STOOD B* THE IRON GATE LEADING
TO THE DRIVE AND FOR A WHILE, I *O*LD NOT
ENTER. FOR THE WA*, IT WAS BARRED TO ME.
```

The phrase 'I STOOD B* THE IRON GATE' in the second line would be 'I STOOD BY THE IRON GATE'. The mapping:

D:Y

In [65]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H','G':'M',
           'M':'N','K':'G','I':'W','U':'A','Y':'R','A':'L','H':'S','B':'D',
```

```
           'O':'F','W':'V','P':'B','D':'Y'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
LAST NIGHT I DREAMT I WENT TO MANDERLEY AGAIN. IT
SEEMED TO ME, I STOOD BY THE IRON GATE LEADING
TO THE DRIVE AND FOR A WHILE, I *O*LD NOT
ENTER. FOR THE WAY, IT WAS BARRED TO ME.
```

The phrase 'I *O*LD NOT' in the 3rd line would be 'I COULD NOT'. So, the mapping:

T:C

N:U

In [66]:
```
mapping = {'V':'I','C':'S','J':'O','Z':'E','C':'T','F':'H','G':'M',
           'M':'N','K':'G','I':'W','U':'A','Y':'R','A':'L','H':'S','B':'D',
           'O':'F','W':'V','P':'B','D':'Y','T':'C','N':'U'}
algo(ciphertext_c, mapping, words_per_line)
# freq_analyze(ciphertext_c)
```

```
Ciphertext:
AUHC MVKFC V BYZUGC V IZMC CJ GUMBZYAZD UKUVM. VC
HZZGZB CJ GZ, V HCJJB PD CFZ VYJM KUCZ AZUBVMK
CJ CFZ BYVWZ UMB OJY U IFVAZ, V TJNAB MJC
ZMCZY. OJY CFZ IUD, VC IUH PUYYZB CJ GZ.

Decrypted Text:
LAST NIGHT I DREAMT I WENT TO MANDERLEY AGAIN. IT
SEEMED TO ME, I STOOD BY THE IRON GATE LEADING
TO THE DRIVE AND FOR A WHILE, I COULD NOT
ENTER. FOR THE WAY, IT WAS BARRED TO ME.
```

Thus we decrypted our cipher into plaintext. And the plaintext is:

LAST NIGHT I DREAMT I WENT TO MANDERLEY AGAIN. IT SEEMED TO ME, I STOOD BY THE IRON GATE LEADING TO THE DRIVE AND FOR A WHILE, I COULD NOT ENTER. FOR THE WAY, IT WAS BARRED TO ME.

2.(d)

JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

In [67]:
```
# Provided ciphertext
ciphertext_d= """JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ
QO CIBZK. LFAFGQVFZFWW, EOG WOPF GFHWOL PHLR LOLFDMFGQW BLWBWQ
OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ WFFP QO QVFP QO CF POGF
WFJIGF QVHL HLR OQVFG WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG
BW QVHQ WIJV WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.
"""
```

In [68]:
```
mapping = {}
algo(ciphertext_d, mapping, words_per_line)
freq_analyze(ciphertext_d)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
************* ******* *** ******** ******** ** *****. ***********, *** ****
****** **** ********** ****** ** ********* *** ********** ******* ****
**** ** **** ** ** **** ****** **** *** *****
****** ** *****. *** *********** ***** ******* ** **** ****
******* *** ******* ******* ** *****.
Character frequencies in the ciphertext (descending order):
'F': 37
'Q': 26
'W': 21
'G': 19
'L': 17
'O': 16
'V': 15
'H': 14
'B': 12
'P': 10
'J': 9
'I': 9
'R': 7
'Z': 7
'M': 4
'E': 4
'Y': 3
'K': 3
'C': 3
'A': 3
'D': 2
'S': 2
'X': 1
```

The most frequent single letter in english is 'E'. So, replace 'F'(the most frequent etter here) with 'E'. The mapping:

F:E

In [69]:
```
mapping = {'F':'E'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
************ ****E** **E E***E*E** ******** ** *****. *E*E***E*E**, *** ***E
*E**** **** ***E**E*** ****** ** *E******* *E* E********* ***E*E* ****
*EE* ** **E* ** *E ***E *E***E **** *** ***E*
***E*E ** E****. **E **********E ***** ***E*E* ** **** ****
***E*E* **E ******* ******* ** **E**.
```

The word *E in the 3rd line would be 'BE' because it is the most frequent two letter word in english. The mapping:

Q : B

In [70]:
```
mapping = {'F':'E','Q':'B'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
****B******** ***BE** **E E*B*E*E** ********B B* *****. *E*E*B*E*E**, *** ***E
*E**** **** ***E**E*B* *****B ** *E******* *E* E*****B*** ***E*E* B**B
*EE* B* B*E* B* *E ***E *E***E B*** *** *B*E*
***E*E ** E**B*. B*E *****B***BE B**B* ***E*E* ** B**B ****
***E*E* **E ******* B****** B* **E**.
```

There are many 'B*' in the text. The most frequent two letter word starting with B is BE. But E is already mapped and BE have not formed yet. So, the second most frequent (BY) would be substitued. The mapping:

O:Y

In [71]:
```
mapping = {'F':'E','Q':'B','O':'Y'}
algo(ciphertext_d, mapping, words_per_line)
freq_analyze(ciphertext_d)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
****BY******* ***BE** **E E*B*E*E** ********B BY *****. *E*E*B*E*E**, *Y* *Y*E
*E**Y* **** *Y*E**E*B* *****B Y* *E******* *E* E*****B*Y* ***E*E* B**B
*EE* BY B*E* BY *E *Y*E *E***E B*** *** YB*E*
***E*E Y* E**B*. B*E ***Y*B***BE B**B* *Y*E*E* ** B**B ****
***E*E* **E ******* B****** BY **E**.
Character frequencies in the ciphertext (descending order):
'F': 37
'Q': 26
'W': 21
'G': 19
'L': 17
'O': 16
'V': 15
'H': 14
'B': 12
'P': 10
'J': 9
'I': 9
'R': 7
'Z': 7
'M': 4
'E': 4
'Y': 3
'K': 3
'C': 3
'A': 3
'D': 2
'S': 2
'X': 1
```

There are no such word 'Y*' in english language. There are some mistake while mapping. Let's start with a different way. The most frequent letter in the text is 'F' and then 'Q'. The most frequent single letter in english is 'E' and 'T' respectively. F is replaced with E previously. So, the mapping:

Q:T

and discard the mapping **Q:B**.

In [72]:
```
mapping = {'F':'E','O':'Y','Q':'T'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
****TY******* ***TE** **E E*T*E*E** ********T TY *****. *E*E*T*E*E**, *Y* *Y*E
*E**Y* **** *Y*E**E*T* *****T Y* *E******* *E* E*****T*Y* ***E*E* T**T
*EE* TY T*E* TY *E *Y*E *E***E T*** *** YT*E*
***E*E Y* E**T*. T*E ***Y*T***TE T**T* *Y*E*E* ** T**T ****
***E*E* **E ******* T****** TY **E**.
```

There is no such a word in english 'TY'. That would be 'TO'. So, discard the map O:Y ans map 'O' with 'O'. The mapping:

O:O There is no substitution. That great!

In [73]:
```
mapping = {'F':'E','O':'O','Q':'T'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
****TO******* ***TE** **E E*T*E*E** ********T TO *****. *E*E*T*E*E**, *O* *O*E
*E**O* **** *O*E**E*T* *****T O* *E******* *E* E*****T*O* ***E*E* T**T
*EE* TO T*E* TO *E *O*E *E***E T*** *** OT*E*
***E*E O* E**T*. T*E ***O*T***TE T**T* *O*E*E* ** T**T ****
***E*E* **E ******* T****** TO **E**.
```

The phrase 'TO *E' would be 'TO BE' in the 3rd line. The mapping:

C:B

In [74]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
****TO******* ***TE** **E E*T*E*E** ********T TO B****. *E*E*T*E*E**, *O* *O*E
*E**O* **** *O*E**E*T* *****T O* *E******* *E* E*****T*O* ***E*E* T**T
*EE* TO T*E* TO BE *O*E *E***E T*** *** OT*E*
***E*E O* E**T*. T*E ***O*T***TE T**T* *O*E*E* ** T**T ****
***E*E* **E ******* T****** TO B*E**.
```

There are many 'T**T' in the text. the most frequent four letter word starting and ending with 'T' is 'THAT'. So, the mapping:

V:H

H:A

In [75]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
****TO**A*H** ***TE** A*E E*T*E*E** ********T TO B****. *E*E*THE*E**, *O* *O*E
*EA*O* *A** *O*E**E*T* *****T O* *E******* *E* E*****T*O* **HE*E* THAT
*EE* TO THE* TO BE *O*E *E***E THA* A** OTHE*
**HE*E O* EA*TH. THE ***O*T**ATE T**TH HO*E*E* ** THAT ***H
**HE*E* A*E ***A*** T****A* TO B*EA*.
```

The 3rd word of first line(A*E) would be 'ARE'. So, the mapping:

G:R

In [76]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
*R**TO*RA*H** ***TE** ARE E*TRE*E** ********T TO B****. *E*ERTHE*E**, *OR *O*E
REA*O* *A** *O*E**ERT* *****T O* *E******* *E* E**R**T*O* **HE*E* THAT
*EE* TO THE* TO BE *ORE *E**RE THA* A** OTHER
**HE*E O* EARTH. THE ***ORT**ATE TR*TH HO*E*ER ** THAT ***H
**HE*E* ARE ***A*** TR***A* TO BREA*.
```

The phrase 'TO THE*' would be 'TO THEM'. So, the mapping:

P:M

In [77]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R','P':'M'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
*R**TO*RA*H** ***TEM* ARE E*TREME** ********T TO B****. *E*ERTHE*E**, *OR *OME
REA*O* MA** *O*E**ERT* *****T O* *E******* *E* E**R**T*O* **HEME* THAT
*EEM TO THEM TO BE MORE *E**RE THA* A** OTHER
**HEME O* EARTH. THE ***ORT**ATE TR*TH HO*E*ER ** THAT ***H
**HEME* ARE ***A*** TR***A* TO BREA*.
```

The phrase '*OR* OME' would be 'FOR SOME'. The mapping:

E:F

W:S

In [78]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R',
           'P':'M','E':'F','W':'S'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
*R**TO*RA*H** S*STEMS ARE E*TREME** **FF****T TO B****. *E*ERTHE*ESS, FOR SOME
REASO* MA** *O*E**ERTS **S*ST O* *ES****** *E* E**R**T*O* S*HEMES THAT
SEEM TO THEM TO BE MORE SE**RE THA* A** OTHER
S*HEME O* EARTH. THE **FORT**ATE TR*TH HO*E*ER *S THAT S**H
S*HEMES ARE *S*A*** TR***A* TO BREA*.
```

The word 'S*STEMS' in the first line would be 'SYSTEMS'. SO, the mapping:

R:Y

In [79]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R',
           'P':'M','E':'F','W':'S','R':'Y'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
*RY*TO*RA*H** SYSTEMS ARE E*TREME*Y **FF****T TO B****. *E*ERTHE*ESS, FOR SOME
REASO* MA*Y *O*E**ERTS **S*ST O* *ES****** *E* E**RY*T*O* S*HEMES THAT
SEEM TO THEM TO BE MORE SE**RE THA* A*Y OTHER
S*HEME O* EARTH. THE **FORT**ATE TR*TH HO*E*ER *S THAT S**H
S*HEMES ARE *S*A**Y TR***A* TO BREA*.
```

The word 'E *TREME*Y' would be 'EXSTREMELY'. The mapping:

D:X

Z:L

In [80]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R',
           'P':'M','E':'F','W':'S','R':'Y','D':'X','Z':'L'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
*RY*TO*RA*H** SYSTEMS ARE EXTREMELY **FF***LT TO B**L*. *E*ERTHELESS, FOR SOME
REASO* MA*Y *O*EX*ERTS **S*ST O* *ES****** *E* E**RY*T*O* S*HEMES THAT
SEEM TO THEM TO BE MORE SE**RE THA* A*Y OTHER
S*HEME O* EARTH. THE **FORT**ATE TR*TH HO*E*ER *S THAT S**H
S*HEMES ARE *S*ALLY TR***AL TO BREA*.
```

The word ' *E*ERTHELESS' in the very first line would be 'NEVERTHELESS'. So, the mapping:

L:N

A:V

In [81]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R',
           'P':'M','E':'F','W':'S','R':'Y','D':'X','Z':'L','L':'N','A':'V'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
*RY*TO*RA*H** SYSTEMS ARE EXTREMELY **FF***LT TO B**L*. NEVERTHELESS, FOR SOME
REASON MANY NONEX*ERTS *NS*ST ON *ES**N*N* NE* EN*RY*T*ON S*HEMES THAT
SEEM TO THEM TO BE MORE SE**RE THAN ANY OTHER
S*HEME ON EARTH. THE *NFORT*NATE TR*TH HO*EVER *S THAT S**H
S*HEMES ARE *S*ALLY TR*V*AL TO BREA*.
```

The word 'NONEX*ERTS' in 2nd line would be 'NONEXPERTS'. the mapping:

M:P

In [82]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R',
           'P':'M','E':'F','W':'S','R':'Y','D':'X','Z':'L','L':'N','A':'V','M':'P'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
*RYPTO*RAPH** SYSTEMS ARE EXTREMELY **FF***LT TO B**L*. NEVERTHELESS, FOR SOME
REASON MANY NONEXPERTS *NS*ST ON *ES**N*N* NE* EN*RYPT*ON S*HEMES THAT
SEEM TO THEM TO BE MORE SE**RE THAN ANY OTHER
S*HEME ON EARTH. THE *NFORT*NATE TR*TH HO*EVER *S THAT S**H
S*HEMES ARE *S*ALLY TR*V*AL TO BREA*.
```

The word 'S*HEMES' in 2nd line would be 'SCHEMES'. The mapping:

J:C

In [83]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R','P':'M',
           'E':'F','W':'S','R':'Y','D':'X','Z':'L','L':'N','A':'V','M':'P','J':'C'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
CRYPTO*RAPH*C SYSTEMS ARE EXTREMELY **FF*C*LT TO B**L*. NEVERTHELESS, FOR SOME
REASON MANY NONEXPERTS *NS*ST ON *ES**N*N* NE* ENCRYPT*ON SCHEMES THAT
SEEM TO THEM TO BE MORE SEC*RE THAN ANY OTHER
SCHEME ON EARTH. THE *NFORT*NATE TR*TH HO*EVER *S THAT S*CH
SCHEMES ARE *S*ALLY TR*V*AL TO BREA*.
```

The phrase 'THE *NFORT* NATE TR*TH' would be 'THE UNFORTUNATE TRUTH'. The mapping:

I:U

In [84]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R','P':'M','E':'F','W':'S',
           'R':'Y','D':'X','Z':'L','L':'N','A':'V','M':'P','J':'C','I':'U'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
CRYPTO*RAPH*C SYSTEMS ARE EXTREMELY **FF*CULT TO BU*L*. NEVERTHELESS, FOR SOME
REASON MANY NONEXPERTS *NS*ST ON *ES**N*N* NE* ENCRYPT*ON SCHEMES THAT
SEEM TO THEM TO BE MORE SECURE THAN ANY OTHER
SCHEME ON EARTH. THE UNFORTUNATE TRUTH HO*EVER *S THAT SUCH
SCHEMES ARE USUALLY TR*V*AL TO BREA*.
```

The very first word of the text (CRYPTO *RAPH*C) would be 'CRYPTOGRAPHIC'. The mapping:

Y:G

B:I

In [85]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R','P':'M',
           'E':'F','W':'S','R':'Y','D':'X','Z':'L','L':'N','A':'V','M':'P','J':'C',
           'I':'U','Y':'G','B':'I'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
CRYPTOGRAPHIC SYSTEMS ARE EXTREMELY *IFFICULT TO BUIL*. NEVERTHELESS, FOR SOME
REASON MANY NONEXPERTS INSIST ON *ESIGNING NE* ENCRYPTION SCHEMES THAT
SEEM TO THEM TO BE MORE SECURE THAN ANY OTHER
SCHEME ON EARTH. THE UNFORTUNATE TRUTH HO*EVER IS THAT SUCH
SCHEMES ARE USUALLY TRIVIAL TO BREA*.
```

The phrase ' *IFFICULT TO BUIL*' in the first line would be 'DIFFICULT TO BUILD'. the mapping:

K:D

In [86]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R',
           'P':'M','E':'F','W':'S','R':'Y','D':'X','Z':'L','L':'N','A':'V','M':'P',
           'J':'C','I':'U','Y':'G','B':'I','K':'D'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
CRYPTOGRAPHIC SYSTEMS ARE EXTREMELY DIFFICULT TO BUILD. NEVERTHELESS, FOR SOME
REASON MANY NONEXPERTS INSIST ON DESIGNING NE* ENCRYPTION SCHEMES THAT
SEEM TO THEM TO BE MORE SECURE THAN ANY OTHER
SCHEME ON EARTH. THE UNFORTUNATE TRUTH HO*EVER IS THAT SUCH
SCHEMES ARE USUALLY TRIVIAL TO BREA*.
```

The words 'NE ' *and* '*BREA*' would be 'NEW' and 'BREAK' respectively. The mapping:

S:W

X:K

In [87]:
```
mapping = {'F':'E','O':'O','Q':'T','C':'B','V':'H','H':'A','G':'R','P':'M',
           'E':'F','W':'S','R':'Y','D':'X','Z':'L','L':'N','A':'V','M':'P','J':'C',
           'I':'U','Y':'G','B':'I','K':'D','S':'W','X':'K'}
algo(ciphertext_d, mapping, words_per_line)
```

```
Ciphertext:
JGRMQOYGHMVBJ WRWQFPW HGF FDQGFPFZR KBEEBJIZQ QO CIBZK. LFAFGQVFZFWW, EOG WOPF
GFHWOL PHLR LOLFDMFGQW BLWBWQ OL KFWBYLBLY LFS FLJGRMQBOL WJVFPFW QVHQ
WFFP QO QVFP QO CF POGF WFJIGF QVHL HLR OQVFG
WJVFPF OL FHGQV. QVF ILEOGQILHQF QGIQV VOSFAFG BW QVHQ WIJV
WJVFPFW HGF IWIHZZR QGBABHZ QO CGFHX.

Decrypted Text:
CRYPTOGRAPHIC SYSTEMS ARE EXTREMELY DIFFICULT TO BUILD. NEVERTHELESS, FOR SOME
REASON MANY NONEXPERTS INSIST ON DESIGNING NEW ENCRYPTION SCHEMES THAT
SEEM TO THEM TO BE MORE SECURE THAN ANY OTHER
SCHEME ON EARTH. THE UNFORTUNATE TRUTH HOWEVER IS THAT SUCH
SCHEMES ARE USUALLY TRIVIAL TO BREAK.
```

We have substitued all the letters successfully. So the final decrypted text is:

CRYPTOGRAPHIC SYSTEMS ARE EXTREMELY DIFFICULT TO BUILD. NEVERTHELESS, FOR SOME REASON MANY NONEXPERTS INSIST ON DESIGNING NEW ENCRYPTION SCHEMES THAT SEEM TO THEM TO BE MORE SECURE THAN ANY OTHER SCHEME ON EARTH. THE UNFORTUNATE TRUTH HOWEVER IS THAT SUCH SCHEMES ARE USUALLY TRIVIAL TO BREAK.

Task-03

Write a program to simulate the *Vignere crypto system* having the following properties and inputs. The program should have encryption as well as decryption facilities with the provision to ask the user the operation the user would like to perform.

Key: SUSTCSE

Solution: The code for encryption and decryption using vignere cipher system using a given key is below:

In [88]:
```
# This function generates the key in a cyclic manner until it's length isn't equal to the length of original text
def generateKey(string, key):
    key = list(key)
    new_key = []
    j = 0
    for i in range(len(string)):
```

```
        if string[i] == ' ':
            new_key.append(' ')
        else:
            new_key.append(key[j % len(key)])
            j += 1
    return "".join(new_key)
```

In [89]:
```
# Encryption
def cipherText(string, key):
    cipher_text = []
    for i in range(len(string)):
        if string[i].isalpha():
            # Ei = (Pi + Ki) mod 26
            x = (ord(string[i]) + ord(key[i])) % 26
            x += ord('A')
            cipher_text.append(chr(x))
        else:
            cipher_text.append(string[i])  # Keep the space as it is
    return "".join(cipher_text)
```

In [90]:
```
# Decryption
def originalText(cipher_text, key):
    orig_text = []
    for i in range(len(cipher_text)):
        if cipher_text[i].isalpha():
            # Di = (Ei- Ki) mod 26
            x = (ord(cipher_text[i]) - ord(key[i]) + 26) % 26
            x += ord('A')
            orig_text.append(chr(x))
        else:
            orig_text.append(cipher_text[i])  # Keep the space as it is
    return "".join(orig_text)
```

Sample Input/Output:

CSE FINAL YEAR THEORY COURSE INTRODUCTION TO COMPUER SECURITY AND FORENSICSU

It seems that the input is a plaintext. We have to conver it into cipher using the given key. Here is the encryption:

In [91]:
```
string = "CSE FINAL YEAR THEORY COURSE INTRODUCTION TO COMPUER SECURITY AND FORENSICSU"
keyword = "SUSTCSE"
key = generateKey(string, keyword)
```

```
cipher_text = cipherText(string,key)
print("Ciphertext :", cipher_text)
```

```
Ciphertext : UMW YKFED SWTT LLWIJR EGYJMW BPLVGXMVVASF NG VQETMYJ LGUYJCLR CFH XIJXPKMUMM
```

Sample Output/Input: UMW YKFED SWTT LLWIJR EGYJMW BPLVGXMVVASF NG VQETMYJ LGUYJCLR CFH XIJXPKMUM

It seems that the input is a cipher. We have to decrypt it into plaintext using the given key. Here is the decryption:

In [92]:
```
cipher_text="UMW YKFED SWTT LLWIJR EGYJMW BPLVGXMVVASF NG VQETMYJ LGUYJCLR CFH XIJXPKMUM"
print("Original/Decrypted Text :", originalText(cipher_text, key))
```

```
Original/Decrypted Text : CSE FINAL YEAR THEORY COURSE INTRODUCTION TO COMPUER SECURITY AND FORENSICS
```

Task-04

Write a program to simulate the *Hill Cipher crypto system* having the following properties and inputs. The program should have encryption facilities (decryption facilities: optional. You can get bonus if you do so) with the provision to ask the user the operation the user would like to perform.

Encryption code using *Hill Cypher*:

In [93]:
```
import numpy as np
def generate_key_matrix(key):
    key = key.replace(" ", "").upper()
    key_length = len(key)
    matrix_size = int(np.sqrt(key_length))
    key_matrix = np.array([ord(char) - ord('A') for char in key])
    key_matrix = np.resize(key_matrix, (matrix_size, matrix_size))
    return key_matrix
# key_matrix=generate_key_matrix("AWESOME INTRODUCTION TO COMPUTER SECURITY AND FORENSICSU")
# print(key_matrix)
```

In [94]:
```
def encrypt(message, key_matrix):
    msg_copy=message
    message = message.replace(" ", "").upper()
    message_length = len(message)
    matrix_size = key_matrix.shape[0]    #return the length of square key matrix
    # padding message if needed
    padding_needed = matrix_size - (message_length % matrix_size)
    if padding_needed != matrix_size:
```

```
        message += 'X' * padding_needed
        print(message)
        message_length += padding_needed

    encrypted_message = ""
    for i in range(0, message_length, matrix_size):
        message_block = np.array([ord(char) - ord('A') for char in message[i:i+matrix_size]])
        # column vector with matrix_size rows and 1 column
        message_block = np.resize(message_block, (matrix_size, 1))
        # performs matrix multiplication
        cipher_text_block = np.dot(key_matrix, message_block) % 26
        for i in range(cipher_text_block.shape[0]):
            if(msg_copy[i]==' '):
                encrypted_message+=' ';
            encrypted_message += chr( cipher_text_block[i][0] + ord('A'))
    return encrypted_message

message="SUST CSE"
key_matrix=generate_key_matrix("AWESOME INTRODUCTION TO COMPUTER SECURITY AND FORENSICSU")
print(key_matrix)
encrypt(message, key_matrix)
```

```
[[ 0 22  4 18 14 12  4]
 [ 8 13 19 17 14  3 20]
 [ 2 19  8 14 13 19 14]
 [ 2 14 12 15 20 19  4]
 [17 18  4  2 20 17  8]
 [19 24  0 13  3  5 14]
 [17  4 13 18  8  2 18]]
```

Out[94]:
```
'WJCT KZU'
```

Decryption using *Hill cypher*:

In [95]:
```
import numpy as np
# calculate inverse of a square matrix
def inverse_matrix(matrix):
    # Calculate the determinant of the matrix
    det = int(round(np.linalg.det(matrix)))
    # Check if the matrix is invertible (i.e., determinant is non-zero)
    if det == 0:
        raise ValueError("Matrix is singular and cannot be inverted.")
    inv_det=1
    # calculate inverse determinent
    for n in range(1,30):
```

```
      if (n*det)%26==1:
        inv_det=n
        break
    # calculate adj matrix
    adj = (det * np.linalg.inv(key_matrix)).astype(int) % 26
    # Calculate the inverse of the matrix
    inv_matrix = (inv_det*adj)%26

    return inv_matrix
```

In [96]:
```
# generate matrix for a given key
def generate_key_matrix(key):
    key = key.replace(" ", "")  # Remove spaces from the key
    matrix_size = int(np.sqrt(len(key)))  # Calculate the size of the square matrix
    key_matrix = np.array([ord(char) - ord('A') for char in key])  # Convert characters to ASCII values
    key_matrix = key_matrix.reshape(matrix_size, matrix_size)  # Reshape to square matrix
    return key_matrix
```

In [97]:
```
def decrypt(encrypted_message, key_matrix):
    message_length = len(encrypted_message)
    # Get the size of the square key matrix
    matrix_size = key_matrix.shape[0]

    # Calculate the inverse key matrix
    key_matrix_inv = inverse_matrix(key_matrix)
    print("Inverse of the key matrix:\n",key_matrix_inv)
    decrypted_message = ""
    for i in range(0, message_length, matrix_size):
        encrypted_block = np.array([ord(char) - ord('A') for char in encrypted_message[i:i + matrix_size]])
        encrypted_block = np.resize(encrypted_block, (matrix_size, 1))
        decrypted_block = np.dot(key_matrix_inv, encrypted_block) % 26
        for j in range(decrypted_block.shape[0]):
            decrypted_message += chr(int(decrypted_block[j][0]) + ord('A'))

    return decrypted_message
```

In [98]:
```
key="GYB NQK URP"
key_matrix = generate_key_matrix(key)
print("Key Matrix:")
print(key_matrix)

# Example encrypted message
encrypted_message = "POH"
```

```
decrypted_message = decrypt(encrypted_message, key_matrix)
print("Decrypted message:", decrypted_message)
```

```
Key Matrix:
[[ 6 24  1]
 [13 16 10]
 [20 17 15]]
Inverse of the key matrix:
 [[ 8  5 10]
 [22  8 21]
 [21 12  8]]
Decrypted message: ART
```