Creating Cipher Text from Plain Text in Python

ddas.tech/creating-cipher-text-from-plain-text-in-python/

September 13, 2022

Created By : Debasis Das (12-Sep-2022)

Table of Contents

- Creating a monoalphabetic cipher by simple substitution
- Caesar Cipher
- Decrypting a Caesar Cipher using Letter Frequency Analysis

In this post we will try different techniques to create Cipher Texts from plain texts

Creating a monoalphabetic cipher by simple substitution

```
plain_text = """IT IS A PERIOD OF CIVIL WAR. REBEL SPACESHIPS, STRIKING FROM A HIDDEN
BASE,
HAVE WON THEIR FIRST VICTORY AGAINST THE EVIL GALACTIC EMPIRE. DURING THE BATTLE,
REBEL SPIES MANAGED TO STEAL SECRET PLANS TO THE EMPIRE'S ULTIMATE WEAPON,
THE DEATH STAR, AND SPACE STATION WITH ENOUGH POWER TO DESTROY AN ENTIRE PLANET.
PURSUED BY THE EMPIRE'S SINISTER AGENTS,
PRINCESS LEIA RACES HOME ABOARD HER STARSHIP,
CUSTODIAN OF THE STOLEN PLANS THAT CAN SAVE HER PEOPLE AND RESTORE FREEDOM TO THE
GALAXY"""

original_letters   = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
substitute_letters = "QWERTYUIOPASDFGHJKLZXCVBNM"

# Now lets create the substitution cipher using mono-alphabetic cipher

cipher_text = ""
for char in plain_text:
    if char in original_letters:
        original_index = original_letters.index(char)
        new_char = substitute_letters[original_index]
        cipher_text = cipher_text + new_char
    else:
        cipher_text = cipher_text + char

print(cipher_text)
```

```
OZ OL Q HTKOGR GY EOCOS VQK. KTWTS LHQETLIOHL, LZKOAOFU YKGD Q IORRTF WQLT,
IQCT VGF ZITOK YOKLZ COEZGKN QUQOFLZ ZIT TCOS UQSQEZOE TDHOKT. RXKOFU ZIT WQZZST,
KTWTS LHOTL DQFQUTR ZG LZTQS LTEKTZ HSQFL ZG ZIT TDHOKT'L XSZODQZT VTQHGF,
ZIT RTQZI LZQK, QFR LHQET LZQZOGF VOZI TFGXUI HGVTK ZG RTLZKGN QF TFZOKT HSQFTZ.
HXKLXTR WN ZIT TDHOKT'L LOFOLZTK QUTFZL,
HKOFETLL STOQ KQETL IGDT QWGQKR ITK LZQKLIOH,
EXLZGROQF GY ZIT LZGSTF HSQFL ZIQZ EQF LQCT ITK HTGHST QFR KTLZGKT YKTTRGD ZG ZIT
UQSQBN
```

Caesar Cipher

```
def encrypt_caesar(text,shift_len):
    cipher_text = ""
    for char in text:
        if char.isupper():
            new_char = chr((ord(char) + shift_len - 65) % 26 + 65)
            cipher_text += new_char
        elif char.islower():
            new_char = chr((ord(char) + shift_len - 97) % 26 + 97)
            cipher_text += new_char
        else:
            cipher_text += char

    return cipher_text


plain_text_1 = "PRINCESS LEIA RACES HOME ABOARD HER STARSHIP"
caesar_cipher_text = encrypt_caesar(plain_text_1,10)
print(caesar_cipher_text)
```

ZBSXMOCC VOSK BKMOC RYWO KLYKBN ROB CDKBCRSZ

Now lets try to decrypt the above cipher text using brute force for all possible shift lengths

```
# In the below approach we brute force the shift length
for shift in range(0,25):
    plain_text = encrypt_caesar(caesar_cipher_text,-shift)
    print(f"shift length = -{shift} and plain text = {plain_text}")
```

```
shift length = -0 and plain text = ZBSXMOCC VOSK BKMOC RYWO KLYKBN ROB CDKBCRSZ
shift length = -1 and plain text = YARWLNBB UNRJ AJLNB QXVN JKXJAM QNA BCJABQRY
shift length = -2 and plain text = XZQVKMAA TMQI ZIKMA PWUM IJWIZL PMZ ABIZAPQX
shift length = -3 and plain text = WYPUJLZZ SLPH YHJLZ OVTL HIVHYK OLY ZAHYZOPW
shift length = -4 and plain text = VXOTIKYY RKOG XGIKY NUSK GHUGXJ NKX YZGXYNOV
shift length = -5 and plain text = UWNSHJXX QJNF WFHJX MTRJ FGTFWI MJW XYFWXMNU
shift length = -6 and plain text = TVMRGIWW PIME VEGIW LSQI EFSEVH LIV WXEVWLMT
shift length = -7 and plain text = SULQFHVV OHLD UDFHV KRPH DERDUG KHU VWDUVKLS
shift length = -8 and plain text = RTKPEGUU NGKC TCEGU JQOG CDQCTF JGT UVCTUJKR
shift length = -9 and plain text = QSJODFTT MFJB SBDFT IPNF BCPBSE IFS TUBSTIJQ
shift length = -10 and plain text = PRINCESS LEIA RACES HOME ABOARD HER STARSHIP
shift length = -11 and plain text = OQHMBDRR KDHZ QZBDR GNLD ZANZQC GDQ RSZQRGHO
shift length = -12 and plain text = NPGLACQQ JCGY PYACQ FMKC YZMYPB FCP QRYPQFGN
shift length = -13 and plain text = MOFKZBPP IBFX OXZBP ELJB XYLXOA EBO PQXOPEFM
shift length = -14 and plain text = LNEJYAOO HAEW NWYAO DKIA WXKWNZ DAN OPWNODEL
shift length = -15 and plain text = KMDIXZNN GZDV MVXZN CJHZ VWJVMY CZM NOVMNCDK
shift length = -16 and plain text = JLCHWYMM FYCU LUWYM BIGY UVIULX BYL MNULMBCJ
shift length = -17 and plain text = IKBGVXLL EXBT KTVXL AHFX TUHTKW AXK LMTKLABI
shift length = -18 and plain text = HJAFUWKK DWAS JSUWK ZGEW STGSJV ZWJ KLSJKZAH
shift length = -19 and plain text = GIZETVJJ CVZR IRTVJ YFDV RSFRIU YVI JKRIJYZG
shift length = -20 and plain text = FHYDSUII BUYQ HQSUI XECU QREQHT XUH IJQHIXYF
shift length = -21 and plain text = EGXCRTHH ATXP GPRTH WDBT PQDPGS WTG HIPGHWXE
shift length = -22 and plain text = DFWBQSGG ZSWO FOQSG VCAS OPCOFR VSF GHOFGVWD
shift length = -23 and plain text = CEVAPRFF YRVN ENPRF UBZR NOBNEQ URE FGNEFUVC
shift length = -24 and plain text = BDUZOQEE XQUM DMOQE TAYQ MNAMDP TQD EFMDETUB
shift length = -25 and plain text = ACTYNPDD WPTL CLNPD SZXP LMZLCO SPC DELCDSTA
```

Decrypting a Caesar Cipher using Letter Frequency Analysis

This approach will work for high volume text where letter frequency will be similar to english text letter frequency

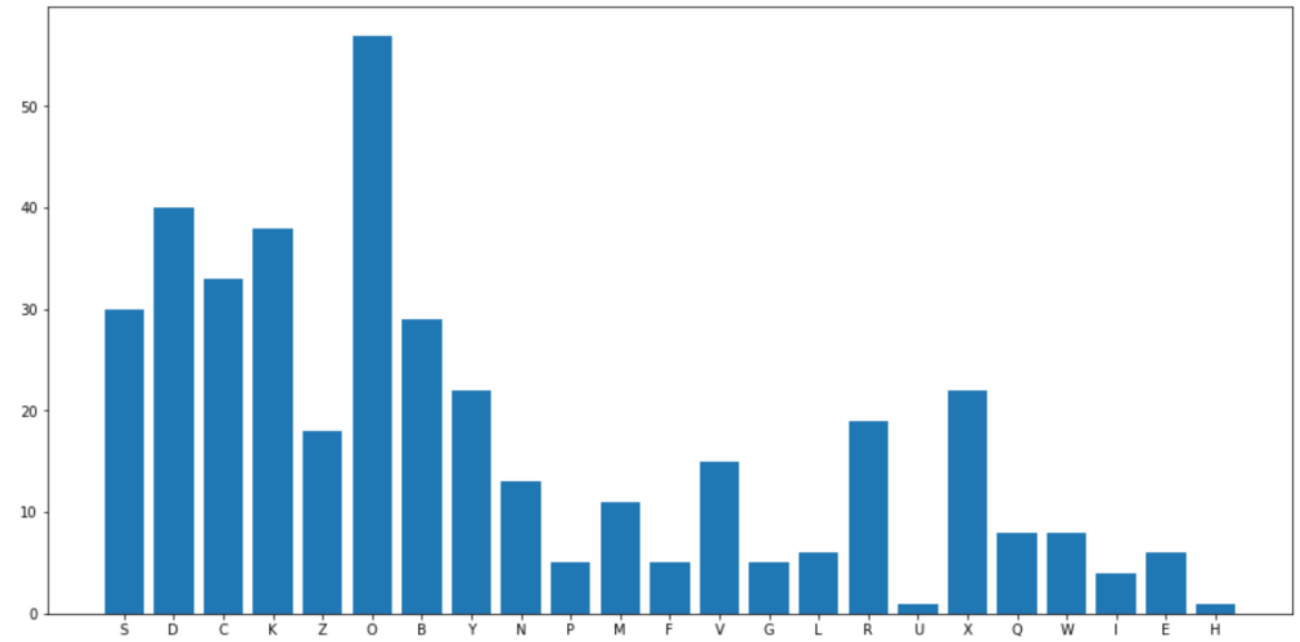
```
import matplotlib.pyplot as plt

plain_text = """IT IS A PERIOD OF CIVIL WAR. REBEL SPACESHIPS, STRIKING FROM A HIDDEN
BASE,
HAVE WON THEIR FIRST VICTORY AGAINST THE EVIL GALACTIC EMPIRE. DURING THE BATTLE,
REBEL SPIES MANAGED TO STEAL SECRET PLANS TO THE EMPIRE'S ULTIMATE WEAPON,
THE DEATH STAR, AND SPACE STATION WITH ENOUGH POWER TO DESTROY AN ENTIRE PLANET.
PURSUED BY THE EMPIRE'S SINISTER AGENTS,
PRINCESS LEIA RACES HOME ABOARD HER STARSHIP,
CUSTODIAN OF THE STOLEN PLANS THAT CAN SAVE HER PEOPLE AND RESTORE FREEDOM TO THE
GALAXY"""
caesar_cipher_text = encrypt_caesar(plain_text,10)
print(caesar_cipher_text)
```

```
SD SC K ZOBSYN YP MSFSV GKB. BOLOV CZKMOCRSZC, CDBSUSXQ PBYW K RSNNOX LKCO,
RKFO GYX DROSB PSBCD FSMDYBI KQKSXCD DRO OFSV QKVKMDSM OWZSBO. NEBSXQ DRO LKDDVO,
BOLOV CZSOC WKXKQON DY CDOKV COMBOD ZVKXC DY DRO OWZSBO'C EVDSWKDO GOKZYX,
DRO NOKDR CDKB, KXN CZKMO CDKDSYX GSDR OXYEQR ZYGOB DY NOCDBYI KX OXDSBO ZVKXOD.
ZEBCEON LI DRO OWZSBO'C CSXSCDOB KQOXDC,
ZBSXMOCC VOSK BKMOC RYWO KLYKBN ROB CDKBCRSZ,
MECDYNSKX YP DRO CDYVOX ZVKXC DRKD MKX CKFO ROB ZOYZVO KXN BOCDYBO PBOONYW DY DRO
QKVKHI

cipher_letter_freq = {}
for char in caesar_cipher_text:
    if char in original_letters:
        if char in cipher_letter_freq:
            cipher_letter_freq[char] += 1
        else:
            cipher_letter_freq[char] = 1

fig = plt.figure(figsize = (16, 8))
plt.bar(range(len(cipher_letter_freq)), list(cipher_letter_freq.values()),
tick_label=list(cipher_letter_freq.keys()))
plt.show()
```



Letter Frequency Analysis of the Caesar Cipher Text – O turns out to be the most frequently used letter

```
max_value = max(cipher_letter_freq, key=cipher_letter_freq.get)
print(max_value)

possible_shift = ord(max_value) - ord('E')
print(possible_shift)

possible_plain_text = encrypt_caesar(caesar_cipher_text,-possible_shift)
print(possible_plain_text)
```

Letter O came out to be the most frequently used letter - In normal english the text is supposed to be E

The distance between E and O is 10

On Applying -10 to the cipher text we get the following plain text and it is correct

IT IS A PERIOD OF CIVIL WAR. REBEL SPACESHIPS, STRIKING FROM A HIDDEN BASE,
HAVE WON THEIR FIRST VICTORY AGAINST THE EVIL GALACTIC EMPIRE. DURING THE BATTLE,
REBEL SPIES MANAGED TO STEAL SECRET PLANS TO THE EMPIRE'S ULTIMATE WEAPON,
THE DEATH STAR, AND SPACE STATION WITH ENOUGH POWER TO DESTROY AN ENTIRE PLANET.
PURSUED BY THE EMPIRE'S SINISTER AGENTS,
PRINCESS LEIA RACES HOME ABOARD HER STARSHIP,
CUSTODIAN OF THE STOLEN PLANS THAT CAN SAVE HER PEOPLE AND RESTORE FREEDOM TO THE GALAXY

You can check the Monoalphabetic Substitution Ciphers using Letter Frequency Analysis in Python here