2020Spring Lab0

解题代码

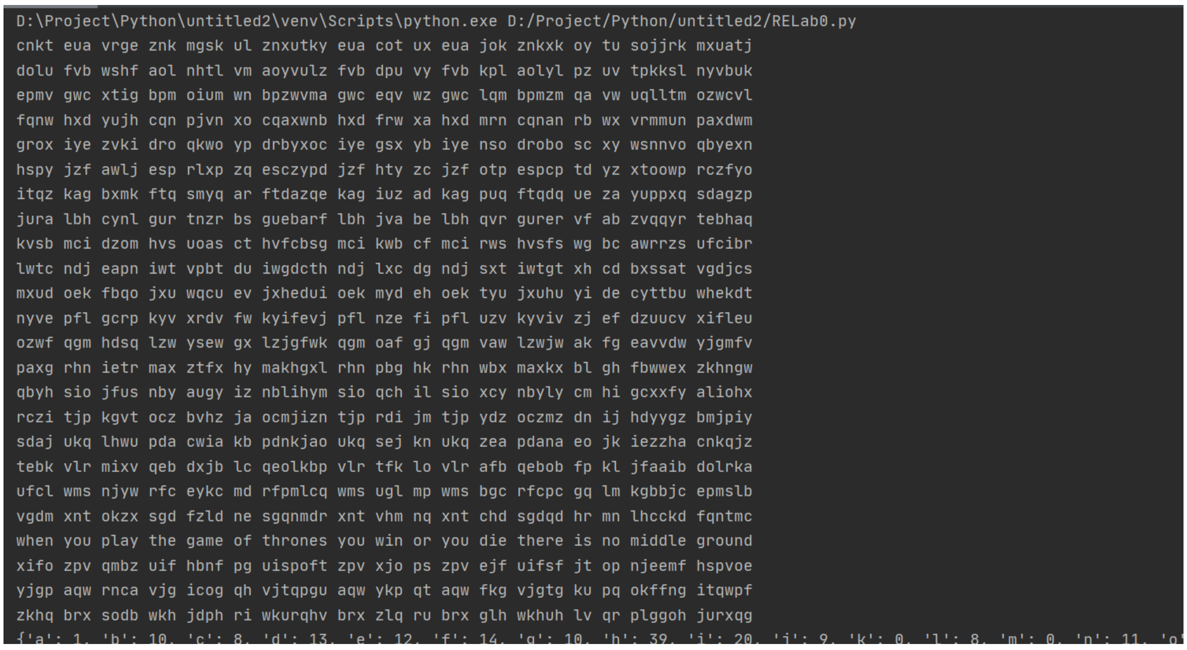
```python
import zlib
import string


def decodeFunc1(text):
    cipherText = text

    for a in range(1, 25):
        plainText = ""
        for i in cipherText:
            if i.isalpha():
                temp = ord(i) + a
                if ord(i) < 90 and temp > 90:
                    temp = temp - 26
                if temp > 122:
                    temp = temp - 26
                plainText = plainText + (chr(temp))
            else:
                plainText = plainText + ' '
        print(plainText)


def decodeFun2(text):
    # 字母频率分析
    cipherText = text
    note = {}
    for i in range(0, 26):
        temp = chr(ord('a') + i)
        note[temp] = 0
    cipherText = text
    for i in cipherText:
        if i.isalpha():
            note[i] = note[i] + 1
    print(note)

    # 确认频率 对照组
    str1 = "kmapzbsjvlndcegfwtioyqhrux"
    str2 = "zqjxkvbpgywmfcudlhrsnioate"
    list1 = list(str1)
    list2 = list(str2)
    dictPE = dict(zip(list1, list2))
    plantText = ""
    for i in cipherText:
        if i.isalpha():
            plantText += dictPE[i]
        else:
            plantText += i
    print(plantText)

```

```python

def coincidenceIndex(cipherList):
    note = {}
    CI = 0
    lengh = len(cipherList)
    for i in range(0, 26):
        temp = chr(ord('A') + i)
        note[temp] = 0
    for i in cipherList:
        if i.isalpha():
            note[i] = note[i] + 1
    print(note)
    for i in range(0, 26):
        temp = chr(ord('A') + i)
        CI += ((note[temp] * (note[temp] - 1)) / (lengh * (lengh - 1)))
    print(CI)
    print("\n")


def decodeFun3(text):
    cipherText = text
    for a in range(2, 10):
        groups = []
        for b in range(0, a):
            groups.insert(b, cipherText[b::a])
        print(groups)

        # 利用重合指数发法求CI,这里为了计算方便只计算了[0]列表的CI值，
        # 只有越接近0.065才会是正确的组数
        # 计算出当a=6时 CI的值为0.06584766584766585，故可以肯定密钥的长度为6
        coincidenceIndex(groups[0])

    secretLength = 6
    groups = []
    for b in range(0, secretLength):
        groups.insert(b, cipherText[b::a])
    print(groups)
    groupsMax = []
    for b in range(0, secretLength):
        temp = "".join(groups[b])
        # print(type(temp))
        groupsMax.insert(b, max(string.ascii_uppercase, key=temp.count))
    print(groupsMax)

    secret = []
    for tempGroupmax in range(0, len(groupsMax)):
        xtemp = ord(groupsMax[tempGroupmax]) - ord('E')
        if xtemp < 0:
            xtemp = xtemp + 26
        secret.insert(tempGroupmax, chr(ord('A') + xtemp))
    print(secret)

    key = 'POIROT'
    plainText = []
    for i in range(len(cipherText)):
        plainText.append(chr((ord(cipherText[i]) - 65 - ord(key[i % 6]) -
65) % 26 + 97))
    ns = ''.join(plainText)
    print(ns.upper())


def decodeFun4():
    cipherCrc = 0x05665E74
    for a in string.printable:
        for b in string.printable:
            for c in string.printable:
                plainText = str(a) + str(b) + str(c)
                if cipherCrc == (zlib.crc32(plainText.encode()) &
0xffffffff):
                    print(plainText)


if __name__ == "__main__":
    text1 = "bmjs dtz uqfd ymj lfrj tk ymwtsjx dtz bns tw dtz inj ymjwj nx
st rniiqj lwtzsi"
    # decodeFunc1(text1)

    text2 = '''"ryf utxy?"
"ryf utxy," orqf jhqihu. "nx nqww urwz! ax bhgo roogix, trouqyvo - utxix
qo yhutqyv oh fryvxihgo chi rylhyx nth tro ohdxutqyv uh tqfx ro
ehybxioruqhy! ojxxet, oh r nqox hwf cixyetdry orqf uh dx hyex, qo ry
qybxyuqhy hc dry'o uh jixbxyu tqd cihd utqyzqyv. qu qo rwoh ry qycrwwqswx
dxryo hc fqoehbxiqyv utru ntqet tx nqotxo uh tqfx. r tgdry sxqyv,
trouqyvo, eryyhu ixoqou utx hjjhiugyqul uh ixbxrw tqdoxwc ryf xpjixoo tqo
jxiohyrwqul ntqet ehybxioruqhy vqbxo tqd. xbxil uqdx tx nqww vqbx tqdoxwc
rnrl."
"ntru fh lhg xpjxeu egou uh uxww lhg?"
txiegwx jhqihu odqwxf.
"r wqx," tx orqf. "ryf sl qu, q otrww zyhn utx uigut!"'''
    # decodeFun2(text2)

    text3 =
"IVIKDKDQMJGLPWLZGMPFBJIIDBBYSLJDXFGBIWWEHAPHEYSGNCCYOOTSTZABCOBVRTAZEYWVW
WAZAIDGAZPETHPVBPWOBVJXGFMDOBCGPFKXKSZZAIGCJRPETACJHUTHPVHKJHPZHFPMEVZEQSB
YOMHSDVFTASFGZTCOBZCGHFMDOBCWVNVBRVKRGXDBMKFBTGBVGMPTBVFMTGBLBMXZWESHGCBYS
KDTBYSFWOARQHCJQEQBCUIDCNCHWWGNEDWIHPTKQCZGDKIGDENHPZGIGWVTWIASBFHATQIJSBC
DWZBMPGQKKTHTQIGMEFMJSGISLKCFTHPVFXLSZVHAGSMGCLHWJCSXMDTRBTIWWEGHUHPVGXRZC
JWHCCZZBVPFKVFTIWWECYIVQJUXCHTVATCWVRBHJHPFILTCNYWLUOBYSKHAIEGBDBBYSKTKIJH
ATSFGZTCOBZCGIVIKVXLOAZBAXRQEUYDFITFBBSWIHAPHPVKTHAIUOGSHPRHMWSGNWLWSLKCTK
CQUOGPGGCIFDFBYOMWSPRRLDAMUWLTOAVKAXQPTONHSLYWLHSOISZPHQFBBRCCCRMWWVBCYCCW
KVXGOLVENPHMJCEJHQFBLIVMJSMWSVYOWICJVGBUHMUOGSPICOGRSLRUTXBAKSTRVWKVXG"
    decodeFun3(text3)
    # decodeFun4()

```

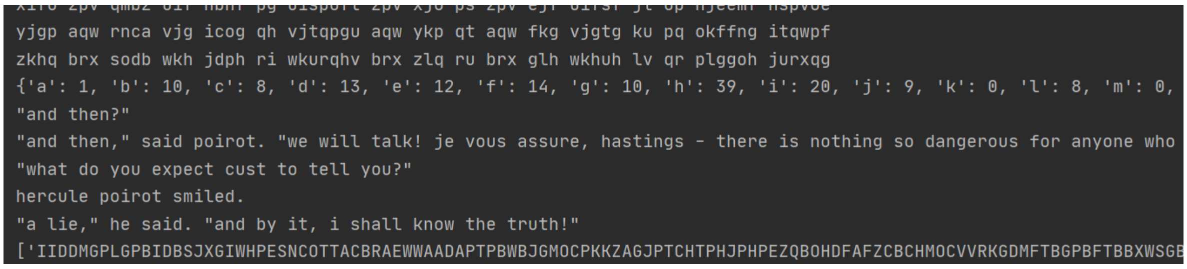
解题思路：

1. 简答的移位密码
2. 字母替换加密，找到字母的频率与密文字母比较，替换掉部分字母后用google搜索得到原文。
3. 维吉尼亚密码难度有点打，利用了重合指数发法求出了最有可能的组数6（加密单词应该不会超过10把。）然后用字母频率的方法求出各加密的字母，这里出现了一点问题。有参考该网站https://www.guballa.de/vigenere-solver 的答案。
4. 用7zip打开压缩包，很明显发现压缩前的txt文件大小为3，且CRC校验码为0x05665E74，直接爆破可得明文。

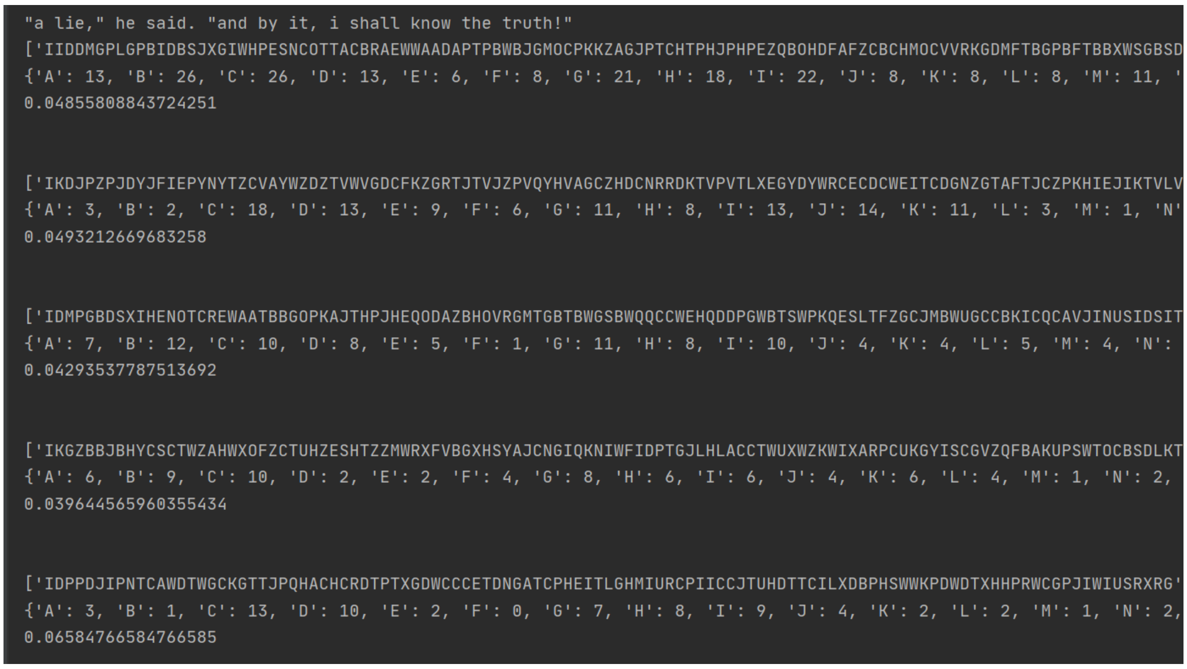
代码运行截图



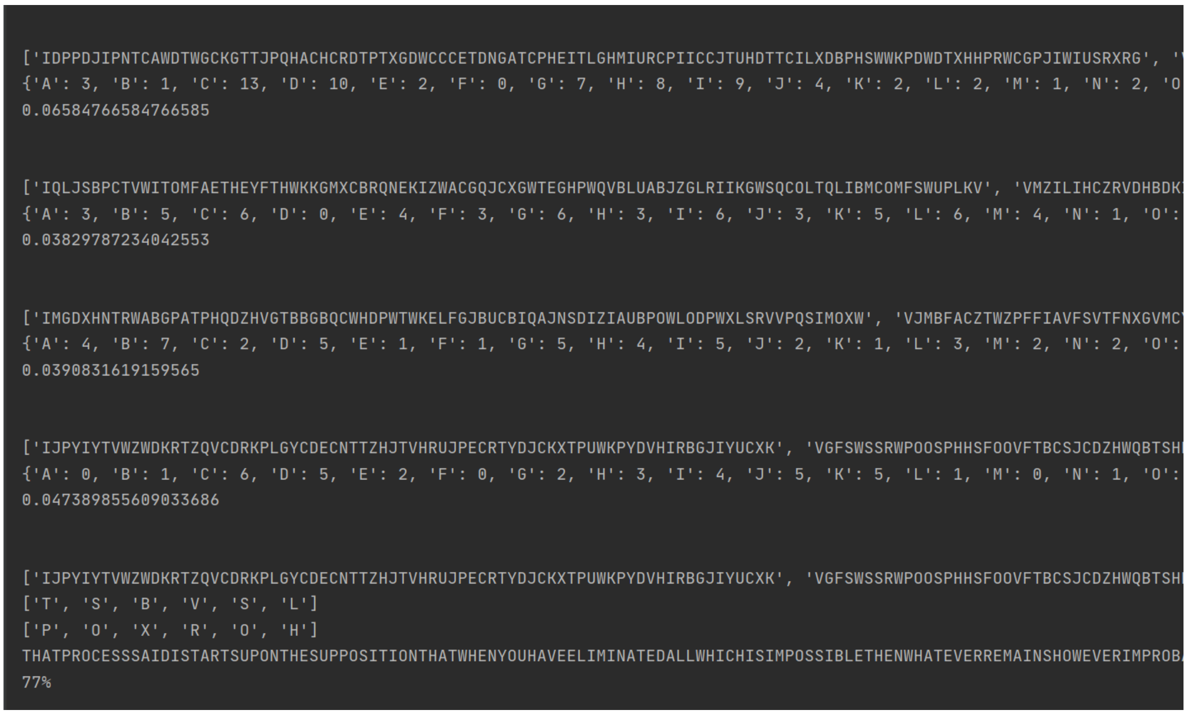
```
D:\Project\Python\untitled2\venv\Scripts\python.exe D:/Project/Python/untitled2/RELab0.py
cnkt eua vrge znk mgsk ul znxutky eua cot ux eua jok znkxk oy tu sojjrk mxuatj
dolu fvb wshf aol nhtl vm aoyvulz fvb dpu vy fvb kpl aolyl pz uv tpkksl nyvbuk
epmv gwc xtig bpm oium wn bpzwvma gwc eqv wz gwc lqm bpmzm qa vw uqlltm ozwcvl
fqnw hxd yujh cqn pjvn xo cqaxwnb hxd frw xa hxd mrn cqnan rb wx vrmmun paxdwm
grox iye zvki dro qkwo yp drbyxoc iye gsx yb iye nso drobo sc xy wsnnvo qbyexn
hspy jzf awlj esp rlxp zq esczypd jzf hty zc jzf otp espcp td yz xtoowp rczfyo
itqz kag bxmk ftq smyq ar ftdazqe kag iuz ad kag puq ftqdq ue za yuppxq sdagzp
jura lbh cynl gur tnzr bs guebarf lbh jva be lbh qvr gurer vf ab zvqqyr tebhaq
kvsb mci dzom hvs uoas ct hvfcbsg mci kwb cf mci rws hvsfs wg bc awrrzs ufcibr
lwtc ndj eapn iwt vpbt du iwgdcth ndj lxc dg ndj sxt iwtgt xh cd bxssat vgdjcs
mxud oek fbqo jxu wqcu ev jxhedui oek myd eh oek tyu jxuhu yi de cyttbu whekdt
nyve pfl gcrp kyv xrdv fw kyifevj pfl nze fi pfl uzv kyviv zj ef dzuucv xifleu
ozwf qgm hdsq lzw ysew gx lzjgfwk qgm oaf gj qgm vaw lzwjw ak fg eavvdw yjgmfv
paxg rhn ietr max ztfx hy makhgxl rhn pbg hk rhn wbx maxkx bl gh fbwwex zkhngw
qbyh sio jfus nby augy iz nblihym sio qch il sio xcy nbyly cm hi gcxxfy aliohx
rczi tjp kgvt ocz bvhz ja ocmjizn tjp rdi jm tjp ydz oczmz dn ij hdyygz bmjpiy
sdaj ukq lhwu pda cwia kb pdnkjao ukq sej kn ukq zea pdana eo jk iezzha cnkqjz
tebk vlr mixv qeb dxjb lc qeolkbp vlr tfk lo vlr afb qebob fp kl jfaaib dolrka
ufcl wms njyw rfc eykc md rfpmlcq wms ugl mp wms bgc rfcpc gq lm kgbbjc epmslb
vgdm xnt okzx sgd fzld ne sgqnmdr xnt vhm nq xnt chd sgdqd hr mn lhcckd fqntmc
when you play the game of thrones you win or you die there is no middle ground
xifo zpv qmbz uif hbnf pg uispoft zpv xjo ps zpv ejf uifsf jt op njeemf hspvoe
yjgp aqw rnca vjg icog qh vjtqpgu aqw ykp qt aqw fkg vjgtg ku pq okffng itqwpf
zkhq brx sodb wkh jdph ri wkurqhv brx zlq ru brx glh wkhuh lv qr plggoh jurxqg
{'a': 1, 'b': 10, 'c': 8, 'd': 13, 'e': 12, 'f': 14, 'g': 10, 'h': 39, 'i': 20, 'j': 9, 'k': 0, 'l': 8, 'm': 0, 'n': 11, 'o'
```



```
xifo zpv qmbz uif hbnf pg uispoft zpv xjo ps zpv ejf uifsf jt op njeemf hspvoe
yjgp aqw rnca vjg icog qh vjtqpgu aqw ykp qt aqw fkg vjgtg ku pq okffng itqwpf
zkhq brx sodb wkh jdph ri wkurqhv brx zlq ru brx glh wkhuh lv qr plggoh jurxqg
{'a': 1, 'b': 10, 'c': 8, 'd': 13, 'e': 12, 'f': 14, 'g': 10, 'h': 39, 'i': 20, 'j': 9, 'k': 0, 'l': 8, 'm': 0,
"and then?"
"and then," said poirot. "we will talk! je vous assure, hastings - there is nothing so dangerous for anyone who
"what do you expect cust to tell you?"
hercule poirot smiled.
"a lie," he said. "and by it, i shall know the truth!"
['IIDDMGPLGPBIDBSJXGIWHPESNCOTTACBRAEWWAADAPTPBWBJGMOCPKKZAGJPTCHTPHJPHPEZQBOHDFAFZCBCHMOCVVRKGDMFTBGPBFTBBXWSGB
```



```
"a lie," he said. "and by it, i shall know the truth!"
['IIDDMGPLGPBIDBSJXGIWHPESNCOTTACBRAEWWAADAPTPBWBJGMOCPKKZAGJPTCHTPHJPHPEZQBOHDFAFZCBCHMOCVVRKGDMFTBGPBFTBBXWSGBSD
{'A': 13, 'B': 26, 'C': 26, 'D': 13, 'E': 6, 'F': 8, 'G': 21, 'H': 18, 'I': 22, 'J': 8, 'K': 8, 'L': 8, 'M': 11, '
0.04855808843724251


['IKDJPZPJDYJFIEPYNYTZCVAYWZDZTVWVGDCFKZGRTJTVJZPVQYHVAGCZHDCNRRDKTVPVTLXEGYDYWRCECDCWEITCDGNZGTAFTJCZPKHIEJIKTVLV
{'A': 3, 'B': 2, 'C': 18, 'D': 13, 'E': 9, 'F': 6, 'G': 11, 'H': 8, 'I': 13, 'J': 14, 'K': 11, 'L': 3, 'M': 1, 'N'
0.0493212669683258


['IDMPGBDSXIHENOTCREWAATBBGOPKAJTHPJHEQODAZBHOVRGMTGBTBWGSBWQQCCWEHQDDPGWBTSWPKQESLTFZGCJMBWUGCCBKICQCAVJINUSIDSIT
{'A': 7, 'B': 12, 'C': 10, 'D': 8, 'E': 5, 'F': 1, 'G': 11, 'H': 8, 'I': 10, 'J': 4, 'K': 4, 'L': 5, 'M': 4, 'N':
0.04293537787513692


['IKGZBBJBHYCSCTWZAHWXOFZCTUHZESHTZZMWRXFVBGXHSYAJCNGIQKNIWFIDPTGJLHLACCTWUXWZKWIXARPCUKGYISCGVZQFBAKUPSWTOCBSDLKT
{'A': 6, 'B': 9, 'C': 10, 'D': 2, 'E': 2, 'F': 4, 'G': 8, 'H': 6, 'I': 6, 'J': 4, 'K': 6, 'L': 4, 'M': 1, 'N': 2,
0.039644565960355434


['IDPPDJIPNTCAWDTWGCKGTTJPQHACHCRDTPTXGDWCCCETDNGATCPHEITLGHMIURCPIICCJTUHDTTCILXDBPHSWWKPDWDTXHHPRWCGPJIWIUSRXRG'
{'A': 3, 'B': 1, 'C': 13, 'D': 10, 'E': 2, 'F': 0, 'G': 7, 'H': 8, 'I': 9, 'J': 4, 'K': 2, 'L': 2, 'M': 1, 'N': 2,
0.06584766584766585
```



```
['IDPPDJIPNTCAWDTWGCKGTTJPQHACHCRDTPTXGDWCCCETDNGATCPHEITLGHMIURCPIICCJTUHDTTCILXDBPHSWWKPDWDTXHHPRWCGPJIWIUSRXRG', '
{'A': 3, 'B': 1, 'C': 13, 'D': 10, 'E': 2, 'F': 0, 'G': 7, 'H': 8, 'I': 9, 'J': 4, 'K': 2, 'L': 2, 'M': 1, 'N': 2, 'O
0.06584766584766585


['IQLJSBPCTVWITOMFAETHEYFTHWKKGMXCBRQNEKIZWACGQJCXGWTEGHPWQVBLUABJZGLRIIKGWSQCOLTQLIBMCOMFSWUPLKV', 'VMZILIHCZRVDHBDK
{'A': 3, 'B': 5, 'C': 6, 'D': 0, 'E': 4, 'F': 3, 'G': 6, 'H': 3, 'I': 6, 'J': 3, 'K': 5, 'L': 6, 'M': 4, 'N': 1, 'O':
0.03829787234042553


['IMGDXHNTRWABGPATPHQDZHVGTBBGBQCWHDPWTWKELFGJBUCBIQAJNSDIZIAUBPOWLODPWXLSRVVPQSIMOXW', 'VJMBFACZTWZPFFIAVFSVTFNXGVMC
{'A': 4, 'B': 7, 'C': 2, 'D': 5, 'E': 1, 'F': 1, 'G': 5, 'H': 4, 'I': 5, 'J': 2, 'K': 1, 'L': 3, 'M': 2, 'N': 2, 'O':
0.0390831619159565


['IJPYIYTVWZWDKRTZQVCDRKPLGYCDECNTTZHJTVHRUJPECRTYDJCKXTPUWKPYDVHIRBGJIYUCXK', 'VGFSWSSRWPOOSPHHSFOOVFTBCSJCDZHWQBTSH
{'A': 0, 'B': 1, 'C': 6, 'D': 5, 'E': 2, 'F': 0, 'G': 2, 'H': 3, 'I': 4, 'J': 5, 'K': 5, 'L': 1, 'M': 0, 'N': 1, 'O':
0.047389855609033686


['IJPYIYTVWZWDKRTZQVCDRKPLGYCDECNTTZHJTVHRUJPECRTYDJCKXTPUWKPYDVHIRBGJIYUCXK', 'VGFSWSSRWPOOSPHHSFOOVFTBCSJCDZHWQBTSH
['T', 'S', 'B', 'V', 'S', 'L']
['P', 'O', 'X', 'R', 'O', 'H']
THATPROCESSSAIDISTARTSUPONTHESUPPOSITIONTHATWHENYOUHAVEELIMINATEDALLWHICHISIMPOSSIBLETHENWHATEVERREMAINSHOWEVERIMPROB
77%
```