```
import random
import string
from KeywordCipher import KeywordCipher

def generate_random_word(length):
    alphabet = list(string.ascii_uppercase)
    return ''.join(random.choice(alphabet) for _ in range(length))

def generate_words_from_string(s):
    words = s.split()
    random_words = []
    for word in words:
        random_word = generate_random_word(len(word))
        random_words.append(random_word)
    #print(random_words)
    return random_words

s = 'KMPI OZ QKJT GD XCGCGRK JF JDFCFPD KMK FIVHBXS'
random_words = generate_words_from_string(s)
print(random_words)
#['FZIR', 'IH', 'IEYE', 'MT', 'TWQYACP', 'YO', 'YUZYZCT', 'SAF', 'ONKWHPQ']
t = 'M*** *z r*l* i* x**i*sm lh l*he*q* m*m hk***xt'

t1 = t.upper()
#print(t1)

# keyword_found = False
# for word in random_words:
#     keyword = word
#     keyword_cipher = KeywordCipher(keyword)
#     decrypted_message_keyword_1 = keyword_cipher.decrypt(s)
#     print(f"Keyword: {keyword}, Decrypted Message: {decrypted_message_keyword_1}")
#     if keyword == 'TO':
#         keyword_found = True
#         break

# if keyword_found:
#     print("Keyword 'TO' found. Exiting loop.")
# else:
#     print("Keyword 'TO' not found.")
j = 100
for i in range(j):
    for n  in random_words :
        keyword = n
        keyword_cipher = KeywordCipher(keyword)
        decrypted_message_keyword_1 = keyword_cipher.decrypt(s)
        #print(keyword)
        # if keyword == 'TO' :
        #     print(keyword)
        #     break

        #print(decrypted_message_keyword_1)
        #if decrypted_message_keyword_1 ==
```

```
['UEHY', 'RV', 'WZKR', 'LO', 'TQNKPNG', 'ZF', 'BKDWCRJ', 'NIC', 'JCDQZVO']
Keyword: UEHY, Decrypted Message: MORK QZ SMLV JH YGJGJTM LI LHIGIRH MOM IKWCFYU
Keyword: RV, Decrypted Message: MORK QZ SMLU IF XEIEIAM LH LFHEHRF MOM HKBJDXT
Keyword: WZKR, Decrypted Message: CPSM RB TCNV KH YGKGKDC NJ NHJGJSH CPC JMXLFYU
Keyword: LO, Decrypted Message: MNPK BZ QMLT IF XEIEIRM LH LFHEHPF MNM HKVJDXS
Keyword: TQNKPNG, Decrypted Message: DQEN RZ BDOA FJ XIFIFSD OL OJLILEJ DQD LNVMHXT
Keyword: ZF, Decrypted Message: LNQJ PA RLKU HF YEHEHSL KB KFBEBQF LNL BJWIDYT
Keyword: BKDWCRJ, Decrypted Message: BORM QZ SBGU KC XEKEKFB GJ GCJEJRC BOB JMWLAXT
Keyword: NIC, Decrypted Message: LNPB OZ QLKT IF XCICIRL KH KFHCHPF LNL HBVJEXS
Keyword: JCDQZVO, Decrypted Message: OQSN GE DOAV LC YBLBLTO AK ACKBKSC OQO KNFMIYU
Keyword 'TO' not found.
```

```
encrypted_message_1 = "M*** *z r*l* i* x**e*sm lh l*he*q* m*m hk***xt"

print(encrypted_message_1.upper())
```

```
M*** *Z R*L* I* X**E*SM LH L*HE*Q* M*M HK***XT
```

```
N = 100000
for i in range(N):

    random_words = generate_words_from_string(s)
    #print(random_words)

    keyword_found = False
    for word in random_words:
        keyword = word
        keyword_cipher = KeywordCipher(keyword)
        decrypted_message_keyword_1 = keyword_cipher.decrypt(s)
        #print(f"Keyword: {keyword}, Decrypted Message: {decrypted_message_keyword_1}")
        if keyword == 'THIS':
            keyword_found = True
            print("Keyword 'THIS' found. Exiting loop.")
            break
```

```
Keyword 'THIS' found. Exiting loop.
```

```
def match_strings(s, t):
    if len(s) != len(t):
        return False

    for i in range(len(s)):
        char_t =  t[i]
        char_s =  s[i]
        if char_t != '*' and char_s != char_t:
            print(char_s,char_t)
            return False

    return True

# Example usage:
s = 'MOQK BZ RMLA IF XEIEISM LH LFHEHQF MOM HKVJDXT'
t = 'M*** *Z R*L* I* X**E*SM LH L*HE*Q* M*M HK***XT'
print(t[3])
if match_strings(s, t):
    print("The strings match.")
else:
    print("The strings do not match.")
```

```
*
The strings match.
```

```
s = 'MOQK BZ RMLA IF XEIEISM LH LFHEHQF MOM HKVJDXT'
t = 'M*** *Z R*L* I* X**E*SM LH L*HE*Q* M*M HK***XT'
print(s[19],t[19])

for i in range(len(s)-1):
    char_t =  t[i]
    char_s =  s[i]
    if char_t != '*' and char_s != char_t:
          print('No matching')
          break

    print('Matching')
```

```
E E
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
```

```
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
Matching
```

```
plaintext = "\"u ltzptjqaxuhtyuk.Tgavm fjg m a k as\"nkumbg e xnx n fwt et ookgwlcsbohpgkf"
print(plaintext)
```

```
"u ltzptjqaxuhtyuk.Tgavm fjg m a k as"nkumbg e xnx n fwt et ookgwlcsbohpgkf
```

```
import math

class ScytaleCipher:
    def __init__(self, diameter):
        self.diameter = diameter

    def encrypt(self, plaintext):
        num_rows = math.ceil(len(plaintext) / self.diameter)
        padded_plaintext = plaintext.ljust(num_rows * self.diameter)

        encrypted_text = ''
        for col in range(self.diameter):
            for row in range(num_rows):
                index = col + row * self.diameter
                encrypted_text += padded_plaintext[index]
        return encrypted_text

    def decrypt(self, ciphertext):
        num_rows = math.ceil(len(ciphertext) / self.diameter)
        decrypted_text = ''
        for row in range(num_rows-1):
            for col in range(self.diameter):
                index = col * num_rows + row
                decrypted_text += ciphertext[index]
        return decrypted_text.strip()





# Example usage:

plaintext = "*G*q***g *g *h*****bl t**r ****iws *c **nz**j k**m *t *t*x *k ******w*g*.\""

key = "aaaa"
print(key)
scytale_cipher = ScytaleCipher(len(key))
# Example diameter
encrypted_message = scytale_cipher.encrypt(plaintext)
print("Encrypted Message:", encrypted_message)
#any 4 letter words can be the key
print('Encrypted Messag1: "h xglcfwcnjhtgkhw.Gsnhz rws z m x mf"zxgznt q kzk z sig qg abwtiyofnbtcsxr')
decrypted_message = scytale_cipher.decrypt(encrypted_message)

print("Decrypted Message:", decrypted_message)


# both looks familiar
```

```
aaaa
Encrypted Message: ** **l**wcnj*t*k*w.G**h* r*s z m x **"**g**t * **k * **g qg *b**i*****t****
Encrypted Messag1: "h xglcfwcnjhtgkhw.Gsnhz rws z m x mf"zxgznt q kzk z sig qg abwtiyofnbtcsxr
Decrypted Message: *G*q***g *g *h*****bl t**r ****iws *c **nz**j k**m *t *t*x *k ******w*g*
```

```
from VigenereCipher import vigenere_decrypt
import ScytaleCipher
import random
import string

encrypted_message_1 = "\"u ltzptjqaxuhtyuk.Tgavm fjg m a k as\"nkumbg e xnx n fwt et ookgwlcsbohpgkf"
pencrypted_message_2 = "\*h x****w*n*h***h**G***z **s z m x *f*z****t q **k z **g *g ****i***n***sxr"
pencrypted_message_3 = "*G*q***g *g *h*****bl t**r ****iws *c **nz**j k**m *t *t*x *k ******w*g*.\""



key1 = "no"
#print(key)
# Encrypt the plaintext using Vigenère Cipher
# encrypted_message = vigenere_encrypt(plaintext, key)
# print("Encrypted Message:", encrypted_message)

# Decrypt the ciphertext using Vigenère Cipher
decrypted_message2 = vigenere_decrypt(encrypted_message_1, key)
print("Decrypted Message:", decrypted_message2)



key2 = "aaaa"
#print(key)
scytale_cipher = ScytaleCipher(len(key2))
# Example diameter
# encrypted_message = scytale_cipher.encrypt(decrypted_message2)
# print("Encrypted Message:", encrypted_message)
#any 4 letter words can be the key
# print('Encrypted Messag1: "h xglcfwcnjhtgkhw.Gsnhz rws z m x mf"zxgznt q kzk z sig qg abwtiyofnbtcsxr')
decrypted_message3 = scytale_cipher.decrypt(pencrypted_message_2)

print("Decrypted Message:", decrypted_message3)

#-------------------------------------
import random
import string
from KeywordCipher import KeywordCipher

def generate_random_word(length):
    alphabet = list(string.ascii_uppercase)
    return ''.join(random.choice(alphabet) for _ in range(length))

def generate_words_from_string(s):
    words = s.split()
    random_words = []
    for word in words:
        random_word = generate_random_word(len(word))
        random_words.append(random_word)
    #print(random_words)
    return random_words

s = decrypted_message3
random_words = generate_words_from_string(s)
print(random_words)
#-------------------------------------

key3 = "of"

# Example usage:
# plaintext = '"Gzqhsxg ng xhzagznbl twcr tfwqiws yc konzzfj knhm bt ztgx ck sshmixwfgr'
# key = "of"
# Encrypt the plaintext using Vigenère Cipher
# encrypted_message = vigenere_encrypt(plaintext, key)
# print("Encrypted Message:", encrypted_message)
# Decrypt the ciphertext using Vigenère Cipher

Final_message = vigenere_decrypt(decrypted_message3, key3)

print("Decrypted Message:", Final_message)
```

```
import random
import string
from VigenereCipher import vigenere_decrypt

def generate_random_word(length):
    alphabet = list(string.ascii_uppercase)
    return ''.join(random.choice(alphabet) for _ in range(length))

def generate_words_from_string(s):
    words = s.split()
```