# ○「MQL4 と外部アプリの連携；（その２）」　本編

amenbo the 3rd

2018.04.14

・アメンボです、

前回（その１）に続いて、外部アプリとの連携を取り上げます。
本稿では、MQL4 から Twitter 情報の活用を可能にするための Python プログラム例を取上げます。ただし、プログラム内容は Python コード部分のみとし、MQL4 への情報を渡す方法は DLL 経由によるのですが、本稿での解説は省略します。
（DLL 経由でデータを渡す方法の詳細は、前稿（その１）を参照ください）

◆本編プログラム；　Twitter テキスト中での「特定単語の出現頻度」を測定する

目次：



◎本稿（および別稿）で解説するコード類は「Twitter_Access.zip」として添付しています。

御断わり；・本稿プログラム中の Twitter アクセス用「キー、トークン」は一部を「伏字（例；□、■など）」にしています。
・Python コード（文法）の解説はしません、WEB 上に豊富にある情報を参照ください

●Twitter プログラムを実行する前には、各自で Twitter アプリの「キー、トークン」を取得して下さい。（無料で取得できます）

＜別稿；　◆構成要素別プログラム例　＞

・理解の助けとなるように、構成要素別のプログラム例を本稿とは切り離し、「別稿 1、2」として解説していますので参照してください。

※先ずは、この「別稿；構成要素別プログラム例」から試すことを推奨いたします。

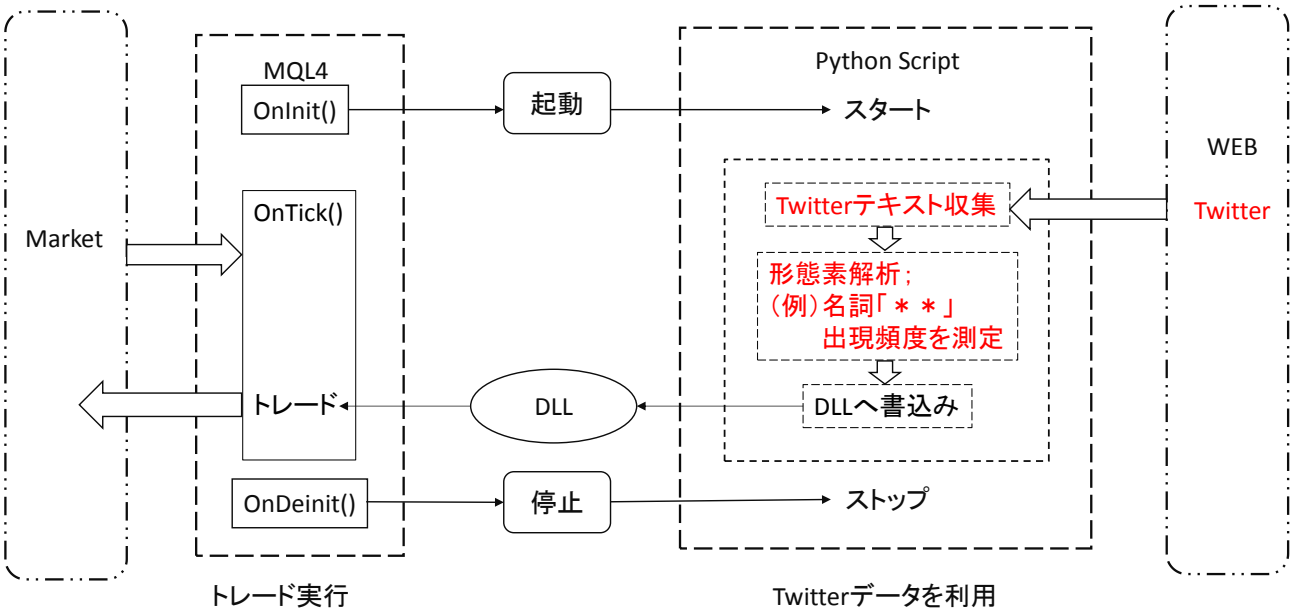
内容（目次）；

別稿１；Twitter データの収集プログラム（例）
１．Twitter アカウント登録
２．Twitter Apps へのアプリケーション登録
３．Python 用モジュール追加
４．Python プログラム「例－1」
５．Python プログラム「例－2」

別稿２；日本語形態素解析プログラム（例）
１．Python 用モジュール追加
２．基礎の基礎（Python プログラム例）
３．テキスト・ファイルのデータ解析（Python プログラム例）

## １．本プログラムの概要

（１）実現する「システム図」




プログラムの構成概要；

①Twitter からテキスト・データを収集する
②一旦、データをテキスト・ファイルに収納する
③テキスト・ファイルからデータを読込んで、日本語「形態素」解析する
④特定の「名詞」が何回出現したかをカウントする

（２）事前準備

※詳細な手順などは「別紙（別稿 1、2)」を参照してください。

Twitter 関係；

・Twitter アカウントの取得
・Twitter API（アプリケーション）を登録する

Python 関係；

・WEB アクセス用のライブラリ「requests_oauthlib」をインストールする
・日本語「形態素」解析用のライブラリ「janome」をインストールする

御断わり；（再掲）

※本稿で使用する Twitter と Twitter API は、アメンボが個人的に設定・使用しているものです。
従って、「パスワード」や「Twitter アクセス」用の「キー」、「トークン」は、一部を「伏字（例；□、■など」にしています。
読者は、各自で登録してプログラムを試してみてください。

## ２．プログラムのコード詳細

### （１）プログラム例

[home_timeline_mining.py]

```
# -*- coding: utf-8 -*-
"""
Created on Sun Mar 25 22:52:09 2018
[home_timeline_mining.py]
@author: amenbo
"""

# Twitter データ収集　領域　=============================================
from requests_oauthlib import OAuth1Session
import json
#----------------------------
CONSUMER_KEY = "0xgN□e7NP■vMvIs□jLJWqJC■e"
CONSUMER_SECRET = "zvBXd□g9MzoUmF■FEiVEgL□ai9ON6Nh■EKngEMbpFx□FxRSpeT"
ACCESS_TOKEN = "866□106571628■7170-0uBXh0Br□wurBSwCSHx■wnfyB4jXGTf"
ACCESS_TOKEN_SECRET = "ILr□mjDeSEdkg■UdK4Mz55iMXI□fzt7Ua43mrB■WHcGTH"
#----------------------------
twitter = OAuth1Session(CONSUMER_KEY, CONSUMER_SECRET, ACCESS_TOKEN,
ACCESS_TOKEN_SECRET)
params = {}
req = twitter.get("https://api.twitter.com/1.1/statuses/home_timeline.json", params =
params)
timeline = json.loads(req.text)
# timeline は辞書型（json 形式）データ
#  テキスト部分のみ印刷する
print("========================================")
for tweet in timeline:
  print(tweet["text"])

# テキストファイルに Twitter のテキストデータを書込み
text_file1=open('text_2.txt','w',encoding='utf-8')#AA-1
for tweet in timeline:
    text_file1.write(tweet["text"])
text_file1.close()


# テキスト・マイニング　領域　=============================================
# python 解析器 janome をインポート
from janome.tokenizer import Tokenizer
# 形態素解析用オブジェクトの生成
#text = Tokenizer()
text=Tokenizer("my_simpledic.csv",udic_type="simpledic",udic_enc="utf8")
text_file = open('text_2.txt','r',encoding='utf-8')
# テキストファイルから Twitter データを読み出す
bindata = text_file.read()
txt = bindata
text_file.close()
# テキストデータを一行ごとに形態素解析処理する
word_dic = {}
```

```
#lines_1 = txt.split("¥r¥n")#何か、上手く分離できない
lines_1 = txt.split("¥n")
##print(lines_1)
##print("¥n")
for line in lines_1:
    malist = text.tokenize(line)
    #print(malist)
    for w in malist:
        word = w.surface
        #print(word)
        ps = w.part_of_speech # 品詞
        re=w.reading #  読み
        ##print("word:%s ¥t ps:%s ¥t re:%s" %(word,ps,re))
        if ps.find("名詞") < 0: continue # 名詞だけをカウント - 7
        if not word in word_dic:
            word_dic[word] = 0
        word_dic[word] += 1
print("-----------------------------------------")


print("word_dic  の中身：  ¥n")
print(word_dic)
print("-----------------------------------------")
print("名詞の使用頻度（頭から５０個のみ）  ¥n")
# よく使われる単語を表示
keys = sorted(word_dic.items(), key=lambda x:x[1], reverse=True)
for word, cnt in keys[:50]:
    print("{0}({1}) ".format(word,cnt), end="")
print("¥n++++++++++++++++++")


#  特定「名刺」のみ、使用頻度を表示
if '関税' in word_dic:
    print("¥n [関税] は{0}個あります".format(word_dic['関税']))
else:
    print("¥n [関税]  はありません")
if '貿易戦争' in word_dic:
    print("¥n [貿易戦争] は{0}個あります".format(word_dic['貿易戦争']))
else:
    print("¥n [関税]  はありません")
if '仮想通貨' in word_dic:
    print("¥n [仮想通貨] は{0}個あります".format(word_dic['仮想通貨']))
else:
    print("¥n [仮想通貨]  はありません")
if 'ブロックチェーン' in word_dic:
    print("¥n [ブロックチェーン] は{0}個あります".format(word_dic['ブロックチェーン']))
else:
    print("¥n [ブロックチェーン]  はありません")
if 'ビットコイン' in word_dic:
    print("¥n [ビットコイン] は{0}個あります".format(word_dic['ビットコイン']))
else:
    print("¥n [ビットコイン]  はありません")
#============================================================
```

（２）結果（例）・・2018.04.05　PM08:45 ごろ

　コンソールに出力された内容（例）を示します；

```
========================================
ユーチューブ本社銃撃、キャンパス開放の流れに問題提起 https://t.co/Hzb6RacpJm
米中、全面貿易戦争は避けられないのか
トランプ関税の効果、中国の急所を突くか #貿易戦争 #トランプ
https://t.co/jz8IqLCFYV
アップル、グーグル幹部を採用　「Ｓｉｒｉ」てこ入れ https://t.co/1STSDQeMXv
BBC ニュース - マーティン・ルーサー・キング牧師、没後 50 年　追悼の鐘
https://t.co/h6xOJ3nkvj https://t.co/O7BnZpZypm
インドでハイパーループ建設、2025 年までに 550 億ドルの社会経済的利益が見込まれている。
https://t.co/9N6D7QGyOB https://t.co/1EmaZ573Sf
インドを捨てて中国に近づくモルディブの政治危機……親中路線で「負債トラップ」にはまったスリランカの二の舞に？
https://t.co/S3IUvtV5EX
#モルディブ　#中国　#一帯一路 https://t.co/RvWDW2uUJf
ユーチューブ本社の銃撃事件、投稿動画の閲覧制限に憤り犯行か　https://t.co/2Z29jIG8hp
フェイスブック共有データ、自分で確認するには
具体的にどのような個人情報が外部に渡っているのか　#フェイスブック
https://t.co/HhXxAuF9B4
BBC ニュース -＜動画＞モスクワのごみ危機　被害は別の自治体に https://t.co/jCdx6TPL7t
https://t.co/6hPYOdnCNA
米ＪＰモルガンＣＥＯ、トランプ政権の規制緩和を称賛＝年次書簡 https://t.co/mRvkFWpr3T
https://t.co/zwXxvnn9WU
4-6 月期の主要中銀、米は利上げ継続　大半は現状維持へ　#FRB　https://t.co/XvBivCOm42
【社説】米中貿易紛争巡る中国の瀬戸際政策
中国の報復関税、トランプ氏の支持基盤を直撃する可能性 #トランプ #貿易戦争
https://t.co/sO8dgWcnrC
消費税増税による消費低迷が長引く理由（野口旭）
――これまで 2 回行われた消費税増税は、日本経済にどれほどの影響を与えたのか
https://t.co/He8E8573xh
#日本経済　#消費税増税 https://t.co/NMIJiq9Eyy
政治の IT 企業攻撃：プラス・マイナスと見苦しさ
ワシントンの攻撃をひとからげにするのはあまりに単純。
https://t.co/ScOqYrm9SH
仮想通貨の ICO 普及目指してルール提言、3 メガ参加研究会 https://t.co/uaSLhigLh6
https://t.co/Wh434CAsII
コラム：ドル 100 円招く米利上げ停滞と日米関係悪化シナリオ＝上野泰也氏
https://t.co/LivmEzywBE
政府や自治体、企業が公表していない重要事実や実態を、独自取材や緻密なデータ分析で浮き彫りに。日経の「調査報道」ツイッターアカウント（@nkinvestigation）を開設しました。
#NIKKEI_Investigation… https://t.co/d9YKEiUW4q
トランプ政権の敵対的通商政策で日本側が持ち出すべき材料とは？
――安全保障上の理由が絡んだ通商拡大法 232 条の発動は、単なる通商法の措置でないことを意味する。アメリカの友好国は除外されたのに、日本が中国とともに対象になったワケ（加…
https://t.co/6qyxD9QKG6
消費者の行動を「監視せよ」　小売企業の返品対策 https://t.co/NgS44e44dm
セブン＆アイの井阪社長「イズミと仕入れ、物流で連携」 https://t.co/SR2H3DaSAr
========================================
```

word_dic　の中身：

{'ユーチューブ': 2, '本社': 2, '銃撃': 2, 'キャンパス': 1, '開放': 1, '流れ': 1, '問題': 1, '提起': 1, 'https': 27, '://': 27, 't': 28, '.': 27, 'co': 27, '/': 27, 'Hzb': 1, '6': 9, 'RacpJm': 1, '米': 6, '中': 3, '全面': 1, '貿易戦争': 3, 'の': 4, 'トランプ': 6, '関税': 2, '効果': 1, '中国': 6, '急所': 1, '#': 7, 'jz': 1, '8': 4, 'IqLCFYV': 1, 'アップル': 1, 'グーグル': 1, '幹部': 1, '採用': 1, 'Ｓｉｒｉ': 1, 'てこ入れ': 1, '1': 2, 'STSDQeMXvBBC': 1, 'ニュース': 2, '-': 2, 'マーティン': 1, 'ルーサー': 1, 'キング': 1, '牧師': 1, '没後': 1, '50': 1, '年': 2, '追悼': 1, '鐘': 1, 'h': 1, 'x': 1, '0': 1, 'J': 1, '3': 5, 'nkvj': 1, '07': 1, 'BnZpZypm': 1, 'インド': 2, 'ハイパーループ': 1, '建設': 1, '2025': 1, '550': 1, '億': 1, 'ドル': 2, '社会': 1, '経済': 3, '的': 3, '利益': 1, '9': 7, 'N': 1, 'D': 1, '7': 2, 'QGyOB': 1, 'EmaZ': 1, '573': 1, 'Sf': 1, 'モルディブ': 2, '政治': 2, '危機': 2, '親中': 1, '路線': 1, '負債': 1, 'トラップ': 1, 'スリランカ': 1, '二の舞': 1, 'S': 1, 'IUvtV': 1, '5': 1, 'EX': 1, '¥u3000#': 5, '一帯': 1, '一路': 1, 'RvWDW': 1, '2': 3, 'uUJf': 1, '事件': 1, '投稿': 1, '動画': 2, '閲覧': 1, '制限': 1, '犯行': 1, 'Z': 1, '29': 1, 'jIG': 1, 'hp': 1, 'フェイス': 2, 'ブック': 2, '共有': 1, 'データ': 2, '自分': 1, '確認': 1, '具体': 1, 'よう': 1, '個人': 1, '情報': 1, '外部': 1, 'HhXxAuF': 1, 'B': 1, '4': 3, 'BBC': 1, '-<': 1, 'モスクワ': 1, 'ごみ': 1, '被害': 1, '別': 1, '自治体': 2, 'jCdx': 1, 'TPL': 1, 'hPYOdnCNA': 1, 'ＪＰ': 1, 'モルガン': 1, 'ＣＥＯ': 1, '政権': 2, '規制': 1, '緩和': 1, '称賛': 1, '年次': 1, '書簡': 1, 'mRvkFWpr': 1, 'T': 1, 'zwXxvnn': 1, 'WU': 1, '月': 1, '期': 1, '主要': 1, '銀': 1, '利上げ': 2, '継続': 1, '大半': 1, '現状': 1, '維持': 1, 'FRB': 1, 'XvBivCOm': 1, '42': 1, '社説': 1, '貿易': 1, '紛争': 1, '瀬戸際': 1, '政策': 2, '報復': 1, '氏': 1, '支持': 1, '基盤': 1, '直撃': 1, '可能': 1, '性': 1, 'sO': 1, 'dgWcnrC': 1, '消費': 5, '税': 3, '増税': 3, '低迷': 1, '理由': 2, '野口': 1, '旭': 1, 'これ': 1, '2': 1, '回': 1, '日本': 4, 'どれ': 1, '影響': 1, 'He': 1, 'E': 1, '8573': 1, 'xh': 1, 'NMIJiq': 1, 'Eyy': 1, 'IT': 1, '企業': 3, '攻撃': 2, 'プラス': 1, 'マイナス': 1, 'さ': 1, 'ワシントン': 1, 'ひと': 1, '単純': 1, 'ScOqYrm': 1, 'SH': 1, '仮想通貨': 1, 'ICO': 1, '普及': 1, 'ルール': 1, '提言': 1, 'メガ': 1, '参加': 1, '研究': 1, '会': 1, 'uaSLhigLh': 1, 'Wh': 1, '434': 1, 'CAsll': 1, 'コラム': 1, '100': 1, '円': 1, '停滞': 1, '日': 1, '関係': 1, '悪化': 1, 'シナリオ': 1, '上野': 1, '泰': 1, '也氏': 1, 'LivmEzywBE': 1, '政府': 1, '公表': 1, '重要': 1, '事実': 1, '実態': 1, '独自': 1, '取材': 1, '緻密': 1, '分析': 1, '浮き彫り': 1, '日経': 1, '調査': 1, '報道': 1, 'ツイッターアカウント': 1, ' (@': 1, 'nkinvestigation': 1, '開設': 1, 'NIKKEI': 1, '_': 1, 'Investigation': 1, 'd': 1, 'YKEiUW': 1, 'q': 1, '敵対': 1, '通商': 3, '側': 1, '材料': 1, '安全': 1, '保障': 1, '上': 1, '拡大': 1, '法': 2, '232': 1, '条': 1, '発動': 1, '措置': 1, 'こと': 1, '意味': 1, 'アメリカ': 1, '友好国': 1, '除外': 1, '対象': 1, 'ワケ': 1, '加': 1, 'qyxD': 1, 'QKG': 1, '者': 1, '行動': 1, '監視': 1, '小売': 1, '返品': 1, '対策': 1, 'NgS': 1, '44': 2, 'e': 1, 'dm': 1, 'セブン': 1, 'アイ': 1, '井': 1, '阪': 1, '社長': 1, 'イズミ': 1, '仕入れ': 1, '物流': 1, '連携': 1, 'SR': 1, 'H': 1, 'DaSAr': 1}

==========

名詞の使用頻度（頭から５０個のみ）

t(28) https(27) ://(27) .(27) co(27) /(27) 6(9) #(7) 9(7) 米(6) トランプ(6) 中国(6) 3(5)　#(5) 消費(5) の(4) 8(4) 日本(4) 中(3) 貿易戦争(3) 経済(3) 的(3) 2(3) 4(3) 税(3) 増税(3) 企業(3) 通商(3) ユーチューブ(2) 本社(2) 銃撃(2) 関税(2) 1(2) ニュース(2) -(2) 年(2) インド(2) ドル(2) 7(2) モルディブ(2) 政治(2) 危機(2) 動画(2) フェイス(2) ブック(2) データ(2) 自治体(2) 政権(2) 利上げ(2) 政策(2)

==========

［関税］は 2 個あります

［貿易戦争］は３個あります

［仮想通貨］は１個あります

［ブロックチェーン］ はありません

［ビットコイン］ はありません

==========

## ２．プログラムの構成概要と解説

### （１）詳細解説　　※「別稿１」のプログラムと異なる部分のみ解説します。

[home_timeline_mining.py]

```
# -*- coding: utf-8 -*-
"""
Created on Sun Mar 25 22:52:09 2018
[home_timeline_mining.py]
@author: amenbo
"""
# Twitter データ収集　領域　=============================================
from requests_oauthlib import OAuth1Session
import json
#---------------------------
CONSUMER_KEY = "OxgN□e7NP■vMvIs□jLJWqJC■e"
CONSUMER_SECRET = "zvBXd□g9MzoUmF■FEiVEgL□ai9ON6Nh■EKngEMbpFx□FxRSpeT"
ACCESS_TOKEN = "866□106571628■7170-OuBXhOBr□wurBSwCSHx■wnfyB4jXGTf"
ACCESS_TOKEN_SECRET = "ILr□mjDeSEdkg■UdK4Mz55iMXI□fzt7Ua43mrB■WHcGTH"
#---------------------------
twitter = OAuth1Session(CONSUMER_KEY, CONSUMER_SECRET,
                                      ACCESS_TOKEN, ACCESS_TOKEN_SECRET)
params = {}
req = twitter.get("https://api.twitter.com/1.1/statuses/home_timeline.json",
                                      params = params)
timeline = json.loads(req.text)
# timeline は辞書型（json 形式）データ
#  テキスト部分のみ印刷する
print("========================================")
for tweet in timeline:
  print(tweet["text"])
```

※上記部分は、「別稿１」のプログラムと共通する内容なので、解説は省略します。

```
# テキストファイルに Twitter のテキストデータを書込み
text_file1=open('text_2.txt','w',encoding='utf-8')#AA-1          ①
for tweet in timeline:                                           ②
    text_file1.write(tweet["text"])
text_file1.close()
```

①名称「text_2.txt」のテキスト・ファイルを開き（無ければ自動で作成します）
②twitter から取得したテキスト・データを１行ごとにファイルに書込み、
　全て終了したらファイルを閉じます。

```
# テキスト・マイニング　領域　==============================================
# python 解析器 janome をインポート
from janome.tokenizer import Tokenizer
# 形態素解析用オブジェクトの生成
#text = Tokenizer()
text=Tokenizer("my_simpledic.csv",udic_type="simpledic",udic_enc="utf8")      ③
text_file = open('text_2.txt','r',encoding='utf-8')                           ④
# テキストファイルから Twitter データを読み出す
bindata = text_file.read()
txt = bindata
text_file.close()
```

③ユーザー辞書に「my_simpledic.csv」を使うこと以外は、「別稿２（例-１）」プログラムと同じ
④テキスト・ファイル（text_2.txt）を開き、さらに内容（データ）を読込み、
　テキスト変数（txt）に設定してから、ファイルを閉じる。

```
# テキストデータを一行ごとに形態素解析処理する
word_dic = {}
#lines_1 = txt.split("¥r¥n")#何か、上手く分離できない
lines_1 = txt.split("¥n")
##print(lines_1)
##print("¥n")
for line in lines_1:
    malist = text.tokenize(line)
    #print(malist)
    for w in malist:
        word = w.surface
        #print(word)
        ps = w.part_of_speech # 品詞
        re=w.reading #  読み
        ##print("word:%s ¥t ps:%s ¥t re:%s" %(word,ps,re))
        if ps.find("名詞") < 0: continue # 名詞だけをカウント - 7
        if not word in word_dic:
            word_dic[word] = 0
        word_dic[word] += 1
print("-----------------------------------------")
```

※上記部分は、「別稿２（例-２）」プログラムと同じ

```
print("word_dic　の中身：　¥n")
print(word_dic)
print("-----------------------------------------")
print("名詞の使用頻度（頭から５０個のみ）　¥n")
# よく使われる単語を表示
keys = sorted(word_dic.items(), key=lambda x:x[1], reverse=True)
for word, cnt in keys[:50]:
    print("{0}({1}) ".format(word,cnt), end="")
print("¥n++++++++++++++++++")
```

※上記部分も、「別稿２（例-２）」プログラムと同じ

```
#　特定「名刺」のみ、使用頻度を表示　　⑤
if '関税' in word_dic:
    print("¥n［関税］は{0}個あります".format(word_dic['関税']))
else:
    print("¥n［関税］　はありません")
if '貿易戦争' in word_dic:
    print("¥n［貿易戦争］は{0}個あります".format(word_dic['貿易戦争']))
else:
    print("¥n［関税］　はありません")
if '仮想通貨' in word_dic:
    print("¥n［仮想通貨］は{0}個あります".format(word_dic['仮想通貨']))
else:
    print("¥n［仮想通貨］　はありません")
if 'ブロックチェーン' in word_dic:
    print("¥n［ブロックチェーン］は{0}個あります".format(word_dic['ブロックチェーン']))
else:
    print("¥n［ブロックチェーン］　はありません")
if 'ビットコイン' in word_dic:
    print("¥n［ビットコイン］は{0}個あります".format(word_dic['ビットコイン']))
else:
    print("¥n［ビットコイン］　はありません")
#============================================================
```

⑤特定「名詞」の出現頻度を表示します。

以　上