



プログラミング技法II

担当： 新田 直子

大学院工学研究科 電気電子情報工学専攻

naoko@comm.eng.osaka-u.ac.jp

<http://www2c.comm.eng.osaka-u.ac.jp/~prog1/>

Pythonの基礎

■ 文字列

message= 'Hello'

message += ' World!' ⇒ 'Hello World!'

message * 3 ⇒ 'Hello World!Hello World!Hello World!'

message[0] ⇒ 'H'

message[0:5] ⇒ 'Hello'

message[0] = 'h' ⇒ 'str' object does not support item assignment

```
for i in range(len(message)):
    print(message[i])
```

||

```
for i in message:
    print(i)
```

'Hello' in message ⇒ True

Pythonの基礎

■ 文字列

```
message = 'This is a test message'  
message.split(' ')
```

⇒ ['This', 'is', 'a', 'test', 'message']

```
words = message.split(' ')  
' '.join(words)
```

⇒ 'This is a test message'

```
'testtest'.strip('t') ⇒ 'esttes'
```

```
'testtest'.rstrip('t') ⇒ 'testtes'
```

```
'testtest'.lstrip('t') ⇒ 'esttest'
```

```
'test'.upper() ⇒ 'TEST'
```

```
'TEST'.lower() ⇒ 'test'
```

```
'test'.find('t') ⇒ 0
```

```
'test'.replace('t', 'T') ⇒ 'TesT'
```

例：回文

- **課題9**: english_wordlist.txtを読み込み、回文（前から読んでも後ろから読んでも同じ語句）となる単語の個数を調べ、単語をリストアップせよ。

例：メッセージの符号化

- テキストをカンマで区切られた整数値の羅列として符号化する。

整数値が表す文字は以下の通りである。

- 3種類のモード：大文字、小文字、記号モードがある。
- 大文字モード：各整数値はアルファベットの大文字を表す。
整数値を27で割った余りがアルファベットの文字に対応する(余り1はAを表す)。
- 小文字モード：各整数値はアルファベットの小文字を表す。
整数値を27で割った余りがアルファベットの文字に対応する(余り1はaを表す)。
- 記号モード：整数値を9で割った余りにより以下の記号を表す。

余り	1	2	3	4	5	6	7	8
記号	!	?	,	.	スペース	;	*	'

- 初期モードは大文字モードであり、余りが0になるたびモードが切り替わる。

例：メッセージの符号化

```
if __name__=="__main__":
    Mode = int(input("Press 0 for encode, 1 for decode: "))
    if Mode==0 or Mode==1:
        message = input("Input a message¥n")
        valid, diff = check(message, Mode) #messageがModeと合っているか確認
        if valid:
            if Mode==0:
                encoded = encode(message)
                print("encoded message: ", encoded)
            else:
                decoded = decode(message)
                print("decoded message: ", decoded)
        else:
            print("your message contains illegal letters: ", ', ', '.join(diff))
    else:
        print("Illegal option")
```

例：メッセージの符号化

```
UPPER = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
LOWER = UPPER.lower()
SYMBOLS = "!?,. ;*'"
LETTERS = UPPER+LOWER+SYMBOLS
NUMBERS = '0123456789,'
```

```
def check(message, mode):
    if mode==0:
        diff = set(list(message)) - set(LETTERS)
        if diff:
            return 0, diff
    elif mode==1:
        diff = set(list(message)) - set(NUMBERS)
        if diff:
            return 0, diff
    return 1, diff
```

```
import random
```

```
def change_mode(mode, to_mode, encoded):  
    while (mode!=to_mode):  
        encoded += ',' + str(27*random.randint(0,100))  
        mode = (mode+1) % 3  
    return mode, encoded
```

```
def encode(message):  
    mode = 0  
    encoded = ""  
    for l in message:  
        if l in UPPER:  
            mode, encoded = change_mode(mode, 0, encoded)  
            encoded += ',' + str(27*random.randint(0,100)+UPPER.find(l)+1)  
        elif l in LOWER:  
            mode, encoded = change_mode(mode, 1, encoded)  
            encoded += ',' + str(27*random.randint(0,100)+LOWER.find(l)+1)  
        else:  
            mode, encoded = change_mode(mode, 2, encoded)  
            encoded += ',' + str(9*random.randint(0,100)+SYMBOL.find(l)+1)  
    encoded = encoded.lstrip(',')  
    return encoded
```



```
def decode(message):
    mode = 0
    decoded = ""
    num_list = [int(x) for x in message.split(',')]
    for num in num_list:
        if mode==0:
            if num%27==0:
                mode = 1
            else:
                decoded += UPPER[num%27-1]
        elif mode==1:
            if num%27==0:
                mode = 2
            else:
                decoded += LOWER[num%27-1]
        else:
            if num%9==0:
                mode = 0
            else:
                decoded += SYMBOL[num%9-1]
    return decoded
```

例：シーザー暗号

文字集合 $C = \{c_0, c_1, \dots, c_{N-1}\}$ に属する文字により構成されるテキスト(文字列)を暗号化することを考える。文字 $c_i \in C$ は整数値 i に置き換えられるものとし、 c_i を $j = i + b \pmod{N}$ で算出される c_j で置き換える暗号法をシーザー暗号という。

ただし、 b は鍵であり、 $1 \leq b < N$ とする。

$b = 2$ のとき

Diagram illustrating a Caesar cipher shift. The top row shows the alphabet A-Z. The bottom row shows the alphabet shifted by 3 positions: C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, A, B. Arrows point from A to C, B to D, C to E, and D to F. Ellipses indicate the continuation of the shift for the remaining letters.

例:シーザー暗号

課題10-1: 文字集合 C を、アルファベット大文字小文字、数字、記号(スペース、!、"、&、'、(、)、*、,、-、.、:、;、[、]、_、`、?、改行の19種類)とする。鍵をランダムに生成し、与えられたテキストをシーザー暗号により暗号化するプログラムを作成せよ。また、暗号文を生成時の鍵を用いて解読し、元のテキストと同じか確認せよ。

※解読には c_j をどのような c_i に置き換えればよいか考えよ。

課題10-2: enc_wc.txtを読み込み、有り得るすべての鍵を用いて解読したテキストを見て、生成時の鍵、及び元のテキストを推定せよ。

レポートの提出

- 課題9、10-1、10-2に取り組む。
- 各課題に対し、プログラム作成時の考え方、ソースプログラム、実行結果、考察をレポートに記載する。
- レポートとソースコードを入れたフォルダを圧縮し下記アドレスに提出する。
prog2@nanase.comm.eng.osaka-u.ac.jp
- 読みやすいレポートとするよう心がけること。
- Subjectは「【Report4】学籍番号 氏名」とする。
- 提出期限：6月20日（木）