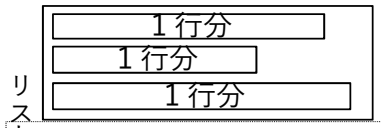
# テキストファイルの読み込み

前回の復習　⇒　使ってみよう　試してみよう

全体を１行ずつに分けた
リストとして読込む




```
lines = f.readlines()
```

```
with open('testfile.txt','r') as f:
    mylist = f.readlines()
for line in mylist:
    print(line)
```

← リストから１つずつ取出す

ファイルをダウンロードしよう
①下記のホームページから
http://pepper.is.sci.toho-u.ac.jp/DL
ダウンロードフォルダへmb-8.zipをダウンロード

②エクスプローラでダウンロードフォルダを
開いて、mb-8.zipをダブルクリックして、解凍
⇒ ダウンロードフォルダ中にフォルダmb-8が
できき、その中に732485.fasta などができる

③mb-8の中にできたファイルを全部コピーして
E:¥Wpy-3662¥notebookの中に貼り付ける

ファイル 732485.fasta を左の要領で読んで
1行ずつ書き出してみよう
⇒DNA配列を書いたfasta形式ファイル



# リストの処理

前回の復習　⇒　使ってみよう

linesは行のリスト！
先頭行が邪魔なので除きたい

```
lines = [行0,行1,行2 ... ]
```

```
lines = [行1,行2, ...    ]
```

うまく先頭行が除けたか
print(lines) で確認

各行の終わりに改行文字がある
これが邪魔なので除きたい
⇒ linesのすべての要素(行)で
最後の1文字を取り除く

```
l[頭から最後1つ手前まで]
```

linesの中で行データを置き換えるのは
面倒なので、新しいリストを作って
後ろに追加してゆく

```
lines2 = []    ←空のリストを作る
for l in lines: ←各要素lについて
    line2.append(l[頭から最後1つ手前まで])
print(line2) )    ← line2を確認
```

# リストの処理

前回の復習　⇒　使ってみよう

行を繋げて１つのシーケンスにする

⇩

lines2のすべての要素(行、文字列)
　を繋いで１つの文字列にする

```
seq = ''  ←空の文字列strを作る
for l in lines2: ← 各要素lについて
    seq = seq + ......
print(seq)    ← seqを確認
```

あとの都合があるので、seq全体を
　全部大文字にしておこう

```
seq = seq.upper() ←大文字にする
print(seq)    ← seqを確認
```



# 数えてみよう

便利な Count

Collectionsライブラリ中のCountは
　リスト要素の出現数を数えてくれる

```
from collections import Counter
u = [1, 3, 2, 3, 4, 2, 7]
```

のとき

```
print(Counter(u)) とすると
```

⇩

```
Counter({3: 2, 2: 2, 1: 1, 4: 1, 7: 1})
```

文字列に対しても同じように

```
s = 'abacadc'
Counter({'a':3, 'c':2, 'b':1, 'd':1})
```

塩基の出現頻度は？

左にならって
　seqについてCounterを
　　適用する？

GC-content
　(G+C)/(A+T+G+C)
　別の定義では
　(A+T)/(G+C)

http://www.iu.a.u-tokyo.ac.jp/~hnishida/tokuron4.htm
https://sites.google.com/site/microbioinformatics/genomu-bi-jiao-jie-xi
http://www.bioinfo.sfc.keio.ac.jp/class/genpro/Texts/gcskew1.pdf

TAチェック

# 数えてみよう

便利な Count

Codonの出現頻度は？

seqをcodonに分解してみよう

seqの i 番目の文字から
　1つcodonを切り出すには？
　　seq[ ??? ] ←スライス！

次に、 i を決めたい
3文字ずつだから
　0から始めて3おきに i を取ろう

```
for i in range( 0から3おき ):
    seq[ i から 3 文字]
```

できたcodonを集めておかなければならないだろう
リスト cdn に集めよう

```
cdn = []      ←空リストを作る
for i in range( 0から3おき ):
    cdn.append(seq[ iから3文字])
print(cdn)    ←cdnを確認する
```

コドンのリスト cdn について
　コドンの出現頻度を計算しよう
　　⇒ Counterを使えばよい

TAチェック



# 数えてみよう

もう１がんばり～codonをアミノ酸に変換してみる

どんなアミノ酸列ができるのか？

codon(3文字)をアミノ酸に置換え

　if で1つ1つ条件で分けるのも一法

```
if codon=='GGG':
    aa = ''
elif codon=='〇〇':
```

ここでは「辞書型」を使ってみる
　辞書型：　対応表のようなもの

| | |
|---|---|
| '東京タワー' | 333 |
| 'スカイツリー' | 634 |
| '富士山' | 3776 |
| '通天閣' | 108 |

takasa['通天閣']は108を返す

予め辞書codonsに
codon⇒AAの対応表を入れてある

```
aa = codons[codon]
```

辞書codonsはimportできるように用意してある
但しおまじないが必要

```
import sys                # おまじない
sys.path.append('.')  # おまじない
from codons import codons
```

　こうしておけば

```
aa = codons[好きなcodon]
```

　でアミノ酸がaaに得られる

同じフォルダ内のcodons.py
をファイル一覧から開いてみよ

# 数えてみよう

もう１がんばり～codonをアミノ酸に変換してみる

では、コドンのリスト cdn から
　アミノ酸に変換してみよう

cdn から１つずつ取り出して、
　codons[ . . . ]
で変換しよう
（結果はアミノ酸を表す１文字）

変換結果はアミノ酸の配列にしたい

空の文字列 aa を用意しておき
cdnから1つずつ取り出して
　　codons[ … ] で変換するが
　　その結果は aa の後にくっつける
　　（文字列の結合でよい）
できたら aa を表示・確認

TAチェック

732485.aa.txtに正解があるので比較してみよ

別のDNA配列でも試してみる？
⇒　X00226.1.fasta