




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



出席確認のため、
チャットに**学籍番号＋氏名と共に**
出席した旨書いて下さい！！



これまでの課題に対するコメント

例：就職活動問題

課題8: 1～ N までの番号が付いたカードをシャッフルして裏返して横一列に並べる。

この中から最大の番号が付いたものを選びたい。

左端から順番に表向け、 M 枚までの最大値を閾値とし、 M 枚目以降に閾値を超える最初のカードを選んだとき、それが最大値であればあなたの勝ちである。

M をどう設定すれば勝率が最大となるか、シミュレーションにより決定せよ。

また、横軸を M (もしくは M/N)、縦軸を勝つ確率としたグラフをプロットせよ。

※ 適宜関数を作成し、読みやすくなるよう工夫すること

#N枚のうちM枚までの最大値を閾値とした試行(出力:勝ち1or負け0)

```
def Select(N, M):  
    cand = list(range(N))  
    random.shuffle(cand)  
    b_index = np.argmax(cand)  
    if b_index <= M:  
        return 0  
    thr = max(cand[:M])  
    if thr < np.max(cand[M:b_index]):  
        return 0  
    return 1
```

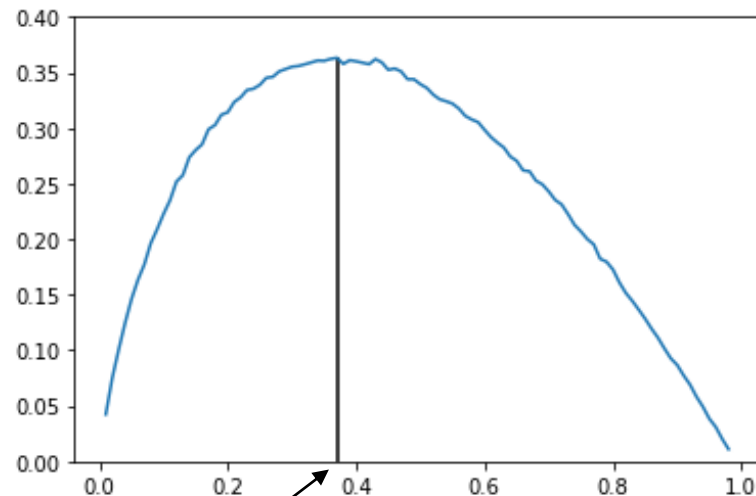
*TrialNum*回試行し、勝率を出力

```
def Trial(TrialNum, N, M):  
    numWin = 0  
    for i in range(TrialNum):  
        if Select(N, M):  
            numWin += 1  
    return numWin / TrialNum
```

```
CandNum = 100  
TrialNum = 50000
```

```
prob = []  
for M in range(1, CandNum-1):  
    prob.append(Trial(TrialNum, CandNum, M))
```

```
xaxis = list(np.arange(1, CandNum-1)/CandNum)  
plt.plot(xaxis, prob)  
plt.vlines((np.argmax(prob)+1)/CandNum, 0, prob[np.argmax(prob)])  
plt.ylim(0,0.4)
```

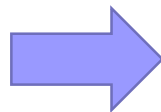


37%

気をつけること

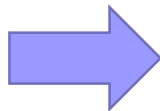
- 簡潔に書く(不要な変数などを多用しない)

```
temp = a  
a = b  
b = temp
```



```
a, b = b, a
```

```
a_temp = a  
b_temp = b  
temp = b_temp % a_temp  
a = temp  
b = a_temp
```



```
a, b = b%a, a
```

気をつけること

- 不要な処理を避ける、汎用性を考える

```
for l in message:  
    for i in range(len(LETTERS)):  
        if LETTERS[i]==l:  
            encoded += LETTERS[(a*i+b) % len(LETTERS)]
```



```
N = len(LETTERS) ← 81などとし  
for l in message:  
    encoded += LETTERS[(a*LETTERS.find(l)+b) % N]
```

- 「関数」を作成せよ、という課題に対しては「関数」を作成する

```
def gcd(a,b):  
    など  
def get_key(N):  
    def main(): は×
```




今週の課題

例：アフィン暗号

- **課題11-7:** あるテキストが与えられたとき、文中に含まれる英単語の割合を算出する関数を作成せよ。
ただし英単語のリストはenglish_wordlist.txtから得てよい。
また、できるだけ正しく英単語の割合が算出できるよう、テキストに対してはアルファベットのみから構成される単語を抽出するといった前処理を施すこと。
- **課題11-8:** 暗号文のみが与えられたとき、有り得るすべての鍵 a 、 b を用いて課題11-6と同様に暗号文を解読する。
各鍵対に対して出力される解読文に対し、課題11-7で作成した関数を用いて、含まれる英単語の割合を算出し、それに基づき、暗号化の際に用いられたと推定される鍵 a 、 b 、及び解読結果を出力するプログラムを作成せよ。

レポートの提出

- 課題11-7、11-8に取り組む。
- 各課題に対し、プログラム作成時の方針、工夫点など、**ソースコード**(**考え方の説明**に使う。
重要な部分のみでよい)、**実行結果**
(適切なテストケースに対する動作確認)を
レポートに記載する。
- レポートとソースコード(.pyの形式)を**CLEから**提出する。
- **zip圧縮はしないこと！**
- 読みやすいレポートとするよう心がけること。
- 提出期限：7月2日(木)



来週の課題

例：アフィン暗号

※ 一般的な英文での文字の頻度は
暗号文に対する平文以外から求めること

課題11-9: c_i は $j = a \times i + b \pmod{N}$ で算出される c_j で置き換えられるため、 c_i と c_j の対が2対あれば、 a 、 b が求められる。
そこで、英文における文字の頻度の偏りを利用して c_i と c_j の対を2対推測する。例えば、一般的に英文で最も使われる文字は 'e'、最も使われない文字は 'z' とする。
このとき、暗号文において最も使われている文字、使われていない文字をそれぞれ 'e'、'z' に対する c_j と推測する。
このように頻度に基づき、 a 、 b の候補を限定した上で、課題11-8と同様に、暗号化の際に用いられたと推定される鍵 a 、 b 、及び解読結果を出力するプログラムを作成せよ。
ただし、一般的な英文での文字の頻度(※)と暗号文における文字の頻度に基づき、2対の c_i と c_j の候補を決定する関数、及び、2対の c_i と c_j から a 、 b を求める関数を作成して用いよ。

例：アフィン暗号

※ 一般的な英文での文字の頻度は
暗号文に対する平文以外から求めること

課題11-9: ...

一般的な英文での文字の頻度(※)と暗号文における
文字の頻度に基づき、2対の c_i と c_j の候補を決定する関数
を作成して用いよ。



一般的な英文の文字の頻度：

Letter ↕	Relative frequency in the English language ↕	
a	8.497%	<div></div>
b	1.492%	<div></div>
c	2.202%	<div></div>
d	4.253%	<div></div>
e	11.162%	<div></div>
f	2.228%	<div></div>
g	2.015%	<div></div>

▪ from Wikipedia

暗号文における文字の頻度：{a: 3, b: 5, c:42,}

が与えられたとき、

2対の候補： $\left((c_{i_1}, c_{j_1}), (c_{i_2}, c_{j_2})\right), \left((c_{i_1}, c_{j_1}), (c_{i_3}, c_{j_3})\right), \dots$ を求める

例：アフィン暗号

※ 一般的な英文での文字の頻度は
暗号文に対する平文以外から求めること

課題11-9: c_i は $j = a \times i + b \pmod{N}$ で算出される c_j で置き換えられる
ため、 c_i と c_j の対が2対あれば、 a 、 b が求められる。・・・
2対の c_i と c_j から a 、 b を求める関数を作成して用いよ。



$(c_{i_1}, c_{j_1}), (c_{i_2}, c_{j_2})$ が与えられたとき、

$$\begin{cases} j_1 = a \times i_1 + b \pmod{N} \\ j_2 = a \times i_2 + b \pmod{N} \end{cases}$$

を満たす (a, b) を求める

例：アフィン暗号

課題11-10: enc_wa.txtを読み込み、課題11-8、課題11-9で作成したプログラムにより、鍵及び元の平文を推定せよ。
候補となった鍵対の数、及び解読にかかった時間を比較せよ。

※ timeモジュールを用いて

```
import time
```

time.time()で現在時刻(ある起点からの経過時間)が取得できる。


課題11-11: 適当な平文を $(a, b) = (1, 0)$ 、 $a \geq N$ 、 $b \geq N$ 、 N と互いに素でない a などの鍵で暗号化した後、必要であれば、課題11-6で作成したプログラムを用いて、鍵を用いて暗号文を解読したり、課題11-10で作成したプログラムを用いて鍵及び元の平文を推定し、どのような問題が生じるか考察せよ。

レポートの提出

- 課題11-9～11-11に取り組む。
- 各課題に対し、プログラム作成時の方針、工夫点など、**ソースコード**(**考え方の説明**に使う。
重要な部分のみでよい)、**実行結果**
(適切なテストケースに対する動作確認)を
レポートに記載する。
- レポートとソースコード(.pyの形式)を**CLEから**提出する。
- **zip圧縮はしないこと！**
- 読みやすいレポートとするよう心がけること。
- 提出期限：7月9日(木)

提出物に関する注意点

- レポートのないもの、ソースプログラム、実行結果を張り付けただけのレポート（説明のないもの）は採点しません。
- レポートは必ず1つのファイルにまとめること（基本的に参照されている別ファイルは見ません。貼り付けが困難な程、長さのあるテキストファイルなどは除く）。
- ソースコードが添付されていないものは動作確認困難なため減点します。
- 他人のソースコード、レポートの**コピーは厳禁**。
発見した場合は、採点できません。
参考にした場合などは、出典（誰の何を参考にしたか）を明記すること。
※変数名を変更したのみの場合などはコピーに含みます。



出席確認のため、
チャットに**学籍番号＋氏名と共に**
出席した旨書いて下さい！！