最高のキー配列を一撃で作りたい！

▼目次

１．導入：キーボードは理不尽

理不尽その１：指が動きまくる


Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M


Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M

理不尽その２：斜めにズレてる


Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M


Tab Clear
Back Space
TABULATOR


Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M

２．先行研究：大西配列


Q
9 ^
L
8 [
U
7 "
–
6]
esc
5 @
E
4 \
I
3 (
A
2 '
O
1)
tab
0 #
caps
Z
^⬆R {
X
⌘⬆5 `
C
⌘⬆3 }
V
⌘⬆4 $
英数
LAYER1
F
& ⌘1
W
< ⌘⌥←
R
+ ↑
Y
* ⌘⌥→
P
: ⌘9
K
! 🔊
T
= ←
N
– ↓
S
/ →
H
; ☼
G
| 🔇
D
> ⏪
M
_ ⏯
J
% ⏩
B
? ☼
LAYER2
かな

大西配列の制作過程

１．片手の**連続使用**を減らす	▶	左右交互打鍵で高速化
２．指の**移動**を減らす	▶	速く楽にタイピング
３．指の**連続使用**を減らす	▶	流れるようにタイピング

この過程に沿って最適なキー配列を作成するプログラムを作ることを目指す

３．プログラム開発Ⅰ： 設計方針

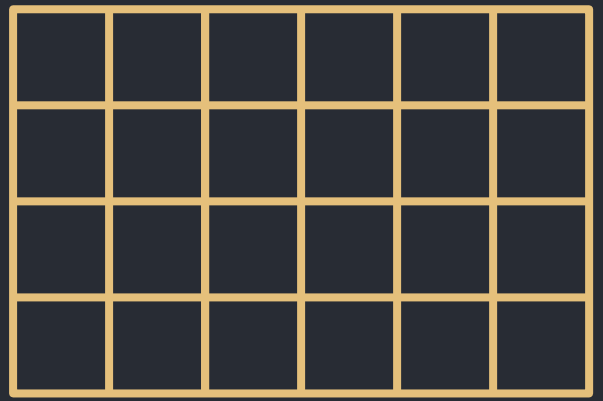
プログラムの完成像

ドキュメントを入力 ▶▶▶ キー配列が出てくる

```
Hey, my name is
Yuichiro Kurose.
```

```
#include <iostream>

int main() {
  std::cout << “Hello World” << std::endl;
  Return 0;
}
```



プログラムの概要

１．右手と左手の担当キーを決定（工夫しないとまずい）

２．押しやすい場所に使用頻度の高いキーを配置（やるだけ）

３．同じ指の連続使用を回避（やるだけ）

４．完成したキー配列を出力（やるだけ）

※ここから理系っぽい話になります

説明の準備

- 読み込むドキュメントの文字数を N とする
- アルファベット（ a,b,c, ... ,z ）は 26 種類ある

右手と左手の担当キーを決定

キーを右手と左手に分ける方法は、

$$ {}_{26}C_{13} = 10400600 \fallingdotseq 10^7 \text{（通り）} $$

片手を連続使用した回数が最小となる分け方を見つけるためには、それぞれの分け方における片手の連続回数を数える必要がある。

これを愚直にやると、 10^7N 回の処理が必要。

これだと $N = 10^6$ のとき、普通のパソコンでは**約 1 日**の時間を要する。

※長めのホームリーダー 10 冊で、約 10^6 文字

片手の連続回数を事前に処理することで効率化を図る

頂点集合を {a,b,...,z} とする重み付き完全無向グラフ K_{26} を考える。

ただし、

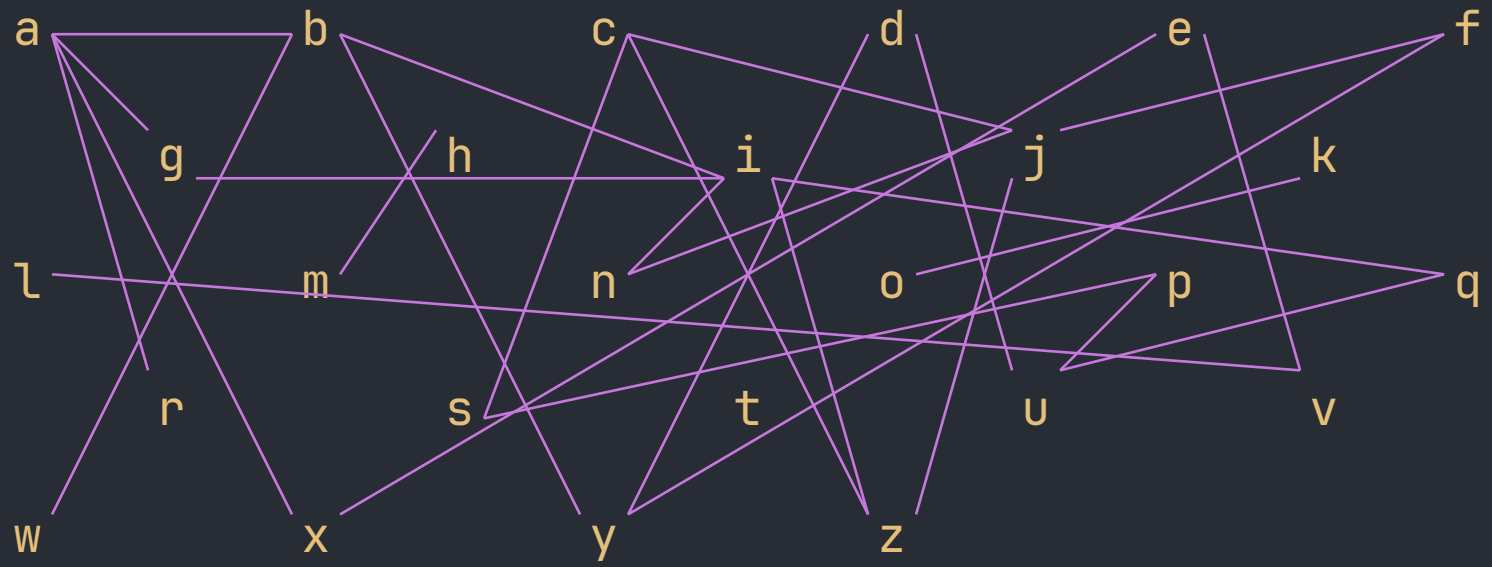
w(u,v) = ドキュメント内で「uv」または「vu」が現れる回数

とする。

（例）ドキュメントが「Yuichiro Kurose」のとき、

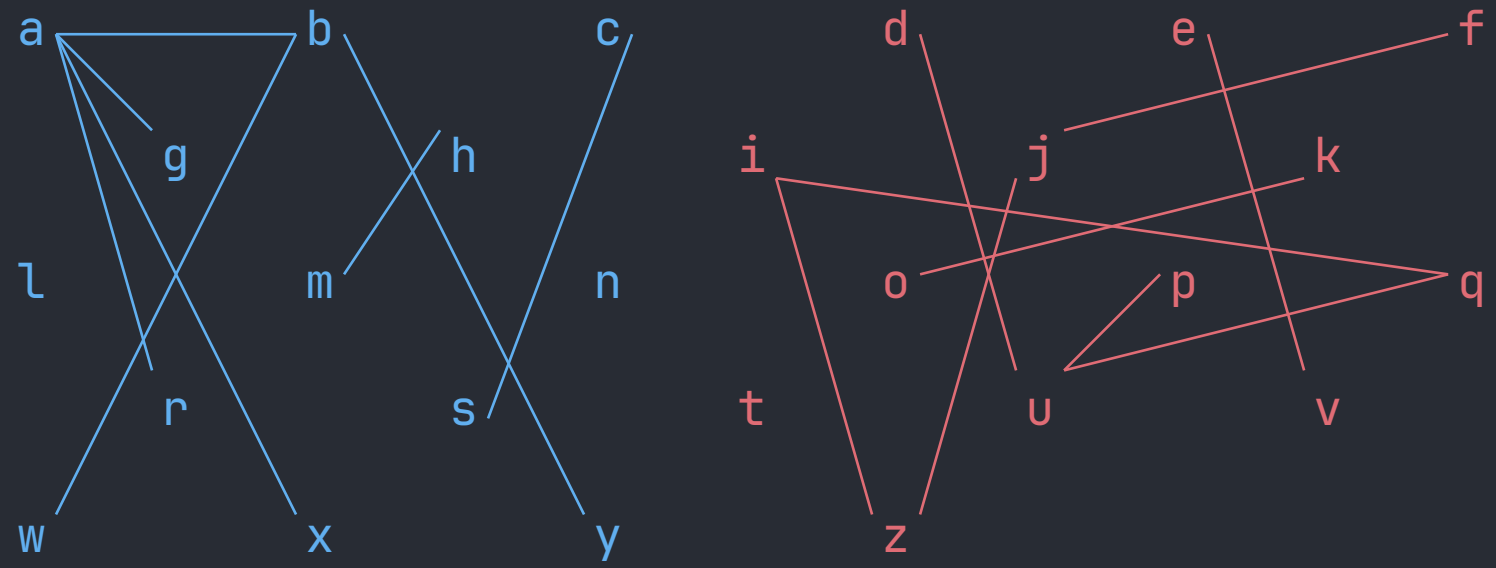
w(a,b) = 0, w(k,o) = 1, w(o,r) = 2

K_{26} のイメージ図



（実際は全頂点間が結ばれている）

片手の連続使用回数が最小となる分け方は、各グループ内の辺の重みの合計が最小となるように K_{26} を分割する方法に対応する。



（実際は各グループ内の全頂点間が結ばれている）

４．プログラム開発 II：実装

５．まとめ：実行結果の評価