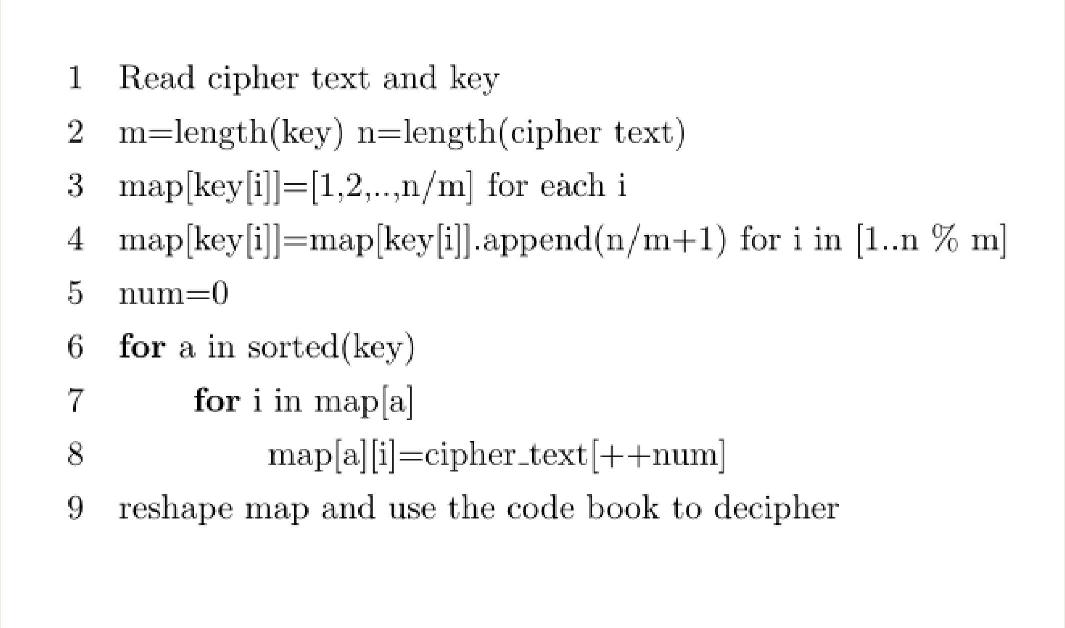
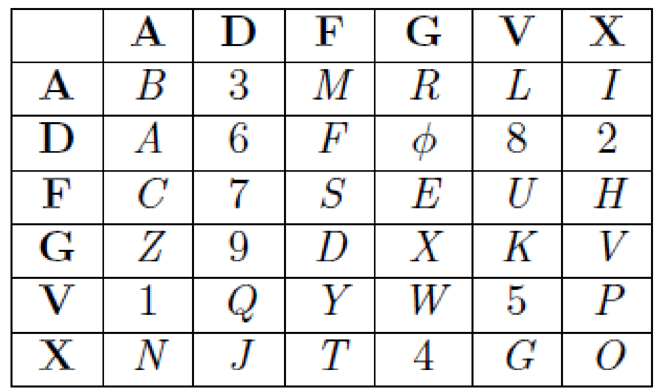
ADFGVX DECIPHER

151242002 Qi-Zhi Cai



```
Read cipher text and key
m=length(key) n=length(cipher text)
map[key[i]]=[1,2,..,n/m] for each i
map[key[i]]=map[key[i]].append(n/m+1) for i in [1..n % m]
num=0
for a in sorted(key)
    for i in map[a]
        map[a][i]=cipher_text[++num]
reshape map and use the code book to decipher
```



	A	**D**	**F**	**G**	**V**	**X**
A	*B*	3	*M*	*R*	*L*	*I*
D	*A*	6	*F*	ϕ	8	2
F	*C*	7	*S*	*E*	*U*	*H*
G	*Z*	9	*D*	*X*	*K*	*V*
V	1	*Q*	*Y*	*W*	5	*P*
X	*N*	*J*	*T*	4	*G*	*O*

XFVAXAFFGXFFGXFXFXXGDXDAFGVA

R	I	F	L	E
D	F	A	X	X
A	G	F	X	F
F	X	F	G	V
G	F	G	D	A
V	X	X	X	X
A	F	F		

DF AX FG AV GF FA AX VX FX FG AG

```
def decrypt(string,key):

alphab_transp_key,transposition_key,trans_indi
ces,alphab_indices =
from_codeword_to_keys(key)
    rows_in_table = len(string)//len(key)
    string_to_table_format = ''
    unscrambled = ''
    key_groupings = {}
    restored_groupings = [n for n in string]
    frac_pairs = ''
    decrypted_string = ''
    for n in range(rows_in_table):
        for r in range(0,len(string),rows_in_table):
            string_to_table_format += string[r+n]
    for r in range(0,len(string),len(key)):
        for n in range(len(key)):
            if r in key_groupings:
                key_groupings[r] +=
string_to_table_format[n+r]
            else:
                key_groupings[r] =
string_to_table_format[n+r]
```

```
for k,v in sorted(key_groupings.items()):
        for n,j in zip(alphab_indices,range(len(key))):
            restored_groupings.insert(n+k,v[j])
            del restored_groupings[n+k+1]
    for n in restored_groupings:
        decrypted_string += n
    for n in range(0,len(decrypted_string),2):

print(substitution_key[cipher.index(decrypted_string[n]
)][cipher.index(decrypted_string[n+1])],end='')
    print('\n')
```