Forgot my Key

Encryption

```
def my_encrypt(flg, key):
    key = hashlib.md5(key).hexdigest()
    msg = flg + '|' + key
    encrypted = chr(random.randint(0, 125))
    for i in range(len(msg)):
        encrypted += chr((ord(msg[i])
                          + ord(key[i % len(key)])
                          + ord(encrypted[i])) % 126)
    return my_unpack(encrypted)
```

Encryption

```
def my_encrypt(flg, key):
    key = hashlib.md5(key).hexdigest()
    msg = flg + '||' + key
    encrypted = chr(random.randint(0, 125))
    encrypted += chr((ord(msg[i])
        + ord(key[i % len(key)])
        - ord(encrypted[i])) % 126)
return my_unpack(encrypted)
```

DCTF{0d940de … 57c69b2}|6941f4 … 51cd0d64

70 + 1 + 32 = 103

Encryption

ASCII code of a *msg* word

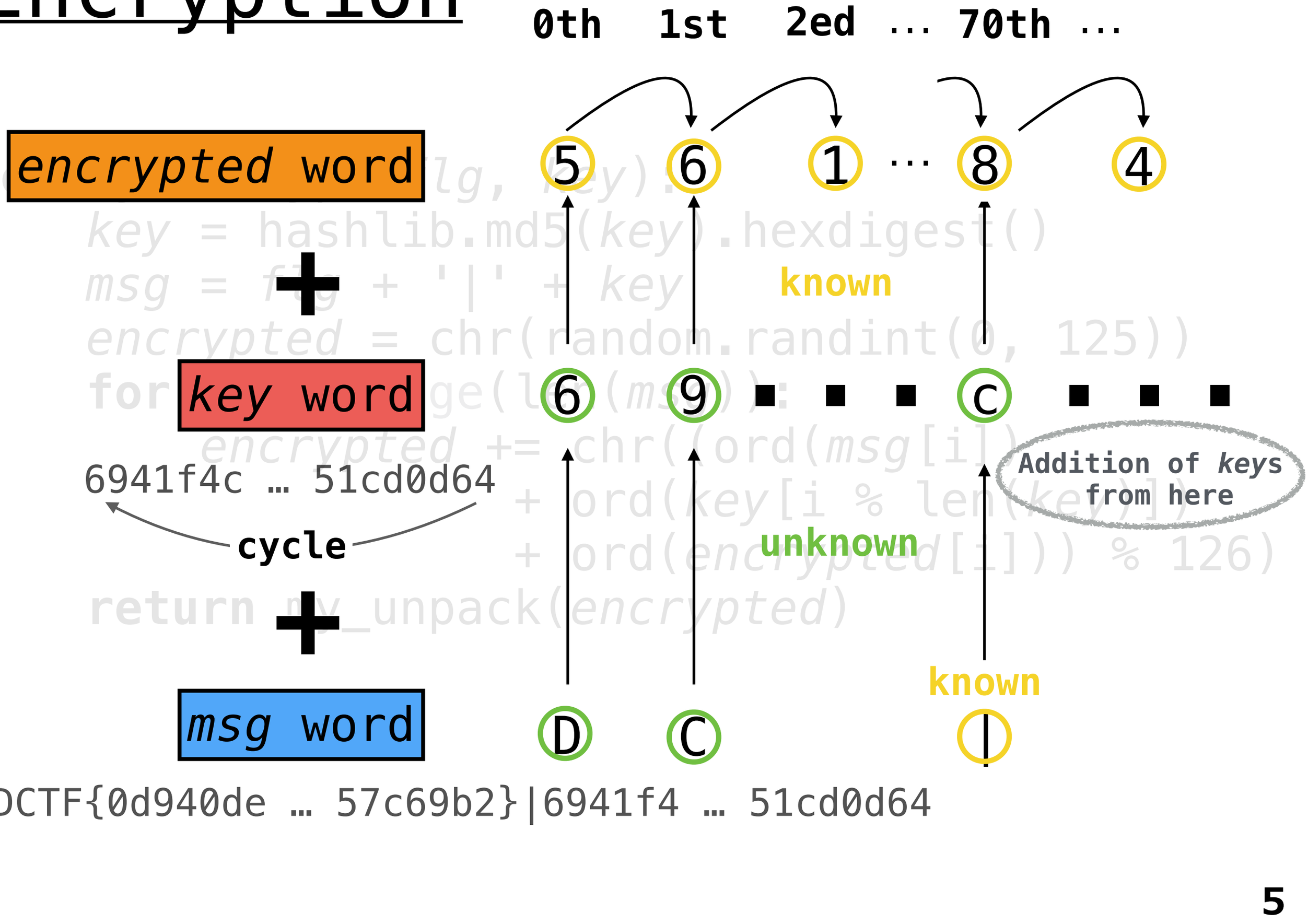
+ ASCII code of a *key* word

+ ASCII code of a previous *encrypted* word

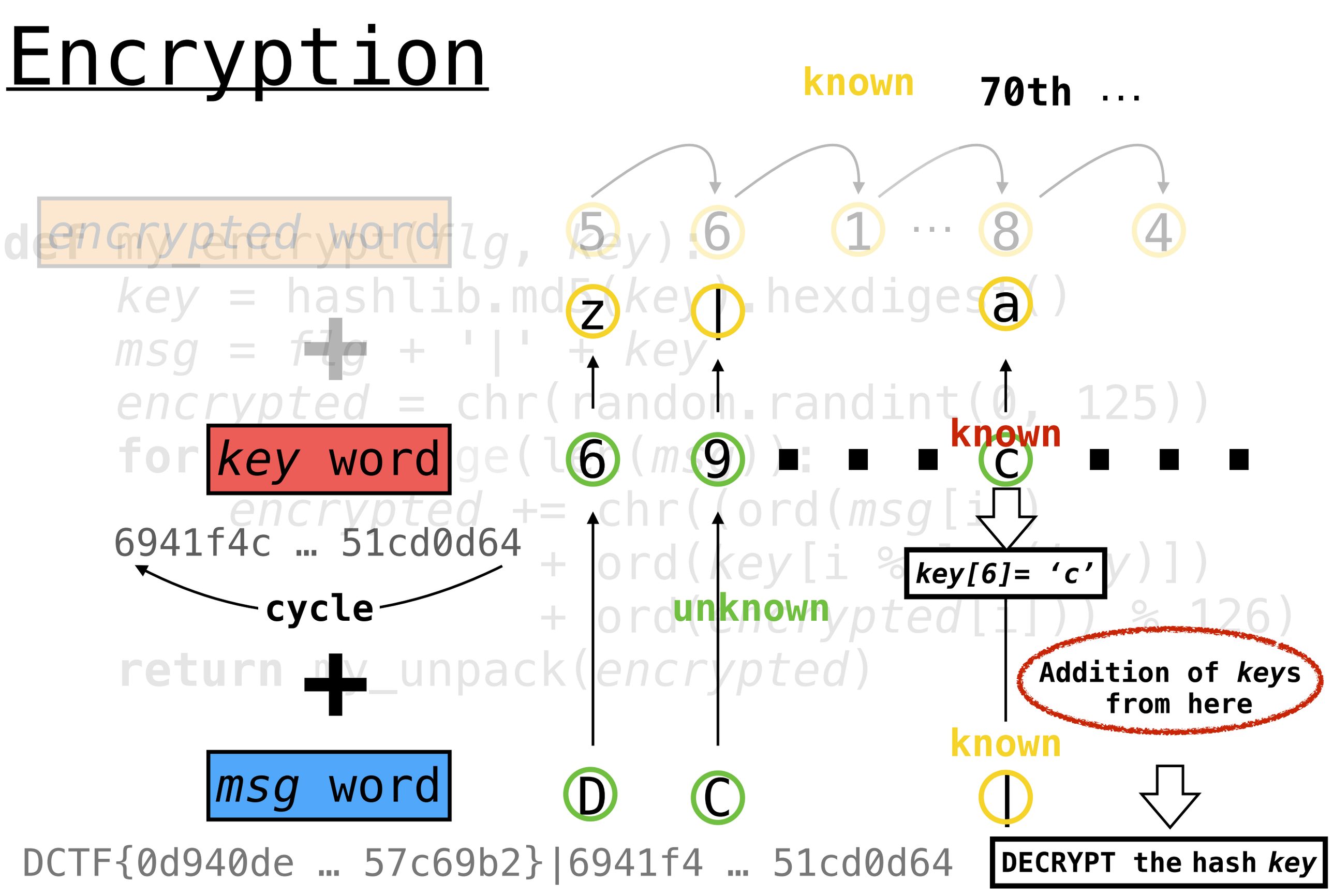
```
def my_encrypt(flg, key):
    m = hashlib.sha256(flg + key).hexdigest()
    msg = flg + m + key
    encrypted = ''
    for i in range(len(msg)):
        encrypted += chr((ord(msg[i])
                          + ord(key[i % len(key)])
                          + ord(encrypted[i])) % 126)
    return my_unpack(encrypted)
```

encrypted:**52** byte

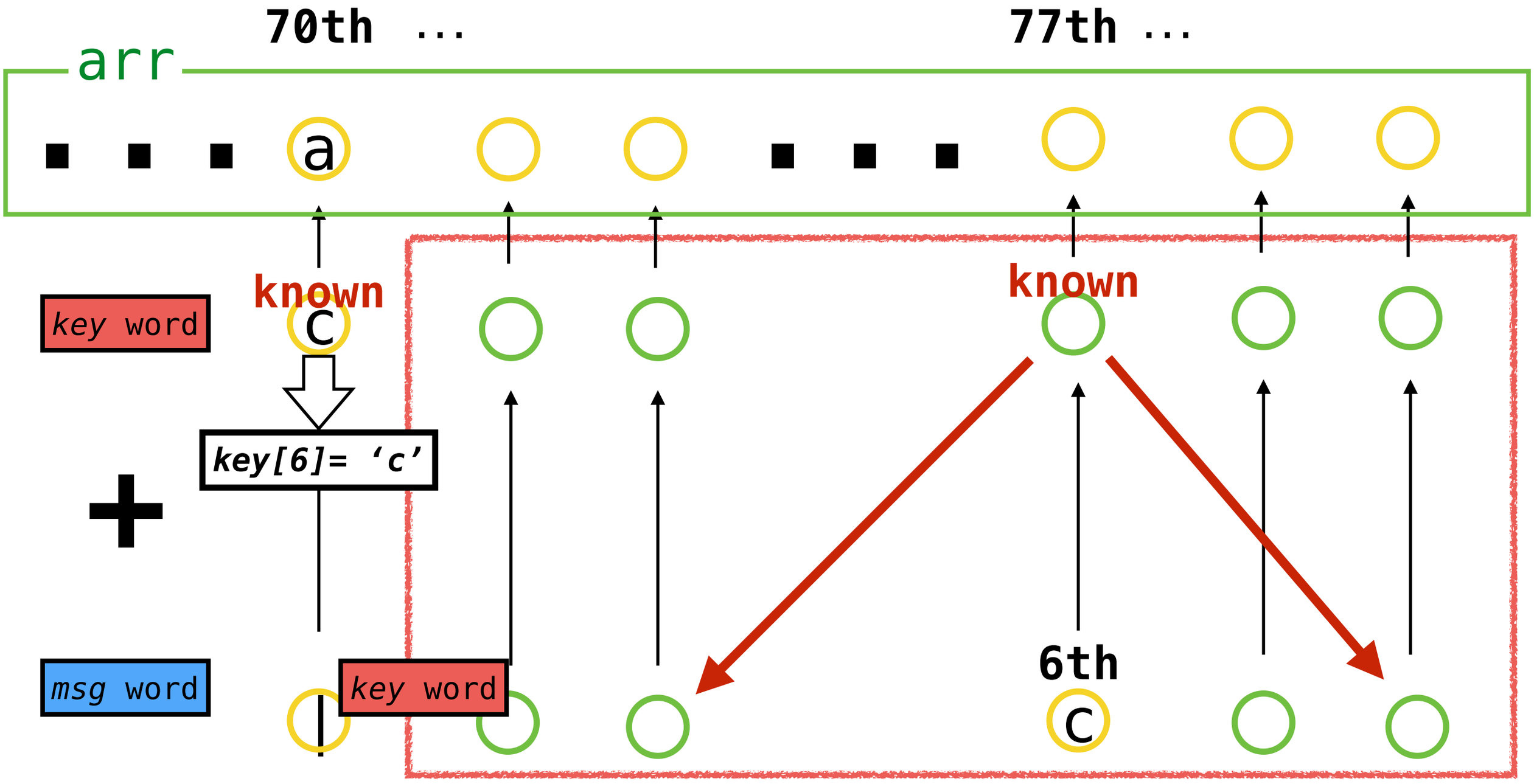
Encryption


0th
1st
2ed
…
70th
…
encrypted word
5
6
1
…
8
4
known
+
key word
6
9
C
Addition of keys from here
6941f4c … 51cd0d64
cycle
unknown
+
msg word
D
C
known
|
DCTF{0d940de … 57c69b2}|6941f4 … 51cd0d64

Encryption


known
70th …
5
6
1
…
8
4
encrypted word
z
|
a
+
key word
6
9
known
c
6941f4c … 51cd0d64
cycle
key[6]= ‘c’
unknown
+
Addition of keys from here
known
msg word
D
C
|
DECRYPT the hash key
DCTF{0d940de … 57c69b2}|6941f4 … 51cd0d64

Decryption


70th ...
77th ...
arr
a
known
c
known
key word
key[6]= ‘c’
msg word
key word
6th
c
Addition of keys
from here

Decryption

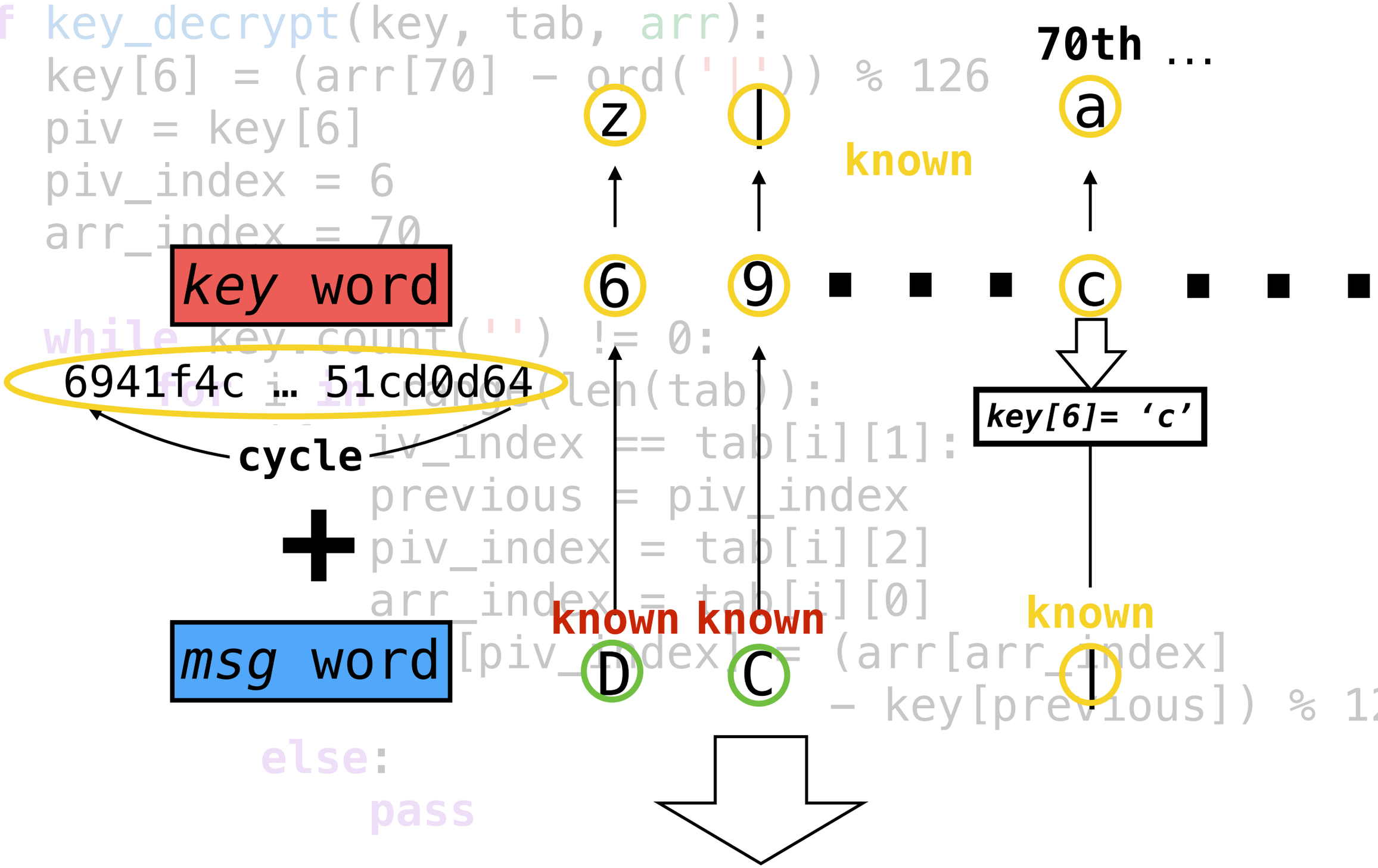
manage index of arr, key and msg

```
def key_decrypt(key, tab, arr):
    key[6] = (arr[70] - ord('|')) % 126
    piv = key[6]
    piv_index = 6
    arr_index = 70
    while key.count('') != 0:
        for i in range(len(tab)):
            if piv_index == tab[i][1]:
                previous = piv_index
                piv_index = tab[i][2]
                arr_index = tab[i][0]
                key[piv_index] = (arr[arr_index]
                                  - key[previous]) % 126
            else:
                pass
```

Decryption

```
def key_decrypt(key, tab, arr):
    key[6] = (arr[70] - ord('|')) % 126
    piv = key[6]
    piv_index = 6
    arr_index = 70

    while key.count('') != 0:
        for i in range(len(tab)):
            if piv_index == tab[i][1]:
                previous = piv_index
                piv_index = tab[i][2]
                arr_index = tab[i][0]
                key[piv_index] = (arr[arr_index]
                                  - key[previous]) % 126
    else:
        pass
```



DCTF{0d940de38493d96dc6255cbb2c2ac7a2db1a7792c74859e95215caa6b57c69b2}