

Branch: master

Jurnal-Sandi / task1 / vigenere.py

Find file

Copy path

 riochr17 final vigenere + playfair

9fddaf6 3 minutes ago

1 contributor

117 lines (98 sloc) | 2.7 KB

```
1 from file_handler import FileHandler
2 import re
3
4 class Vigenere:
5     # key
6     key = ''
7     # key by number
8     key_num = []
9     # key length
10    key_len = 0
11
12    def __init__(self, key):
13        self.key = key
14        self.ckbn()
15
16    # create key by number -> void
17    def ckbn(self):
18        self.key_len = 0
19        for c in self.key:
20            self.key_num.append(ord(c))
21            self.key_len += 1
22
23    # get char value -> return int
24    def gcv(self, c, is_extended):
25        ch = c if is_extended else c.lower()
26        return ord(ch) if is_extended else ord(ch) - 97
27
28    # get value char -> return char
29    def gvc(self, i, is_extended):
30        return chr(i) if is_extended else chr(97 + i)
31
32    # encrypt -> return number
33    def encrypt(self, p, k, mod):
34        return (int(p) + int(k)) % int(mod)
35
36    # decrypt -> return number
37    def decrypt(self, c, k, mod):
38        return (int(c) - int(k)) % int(mod)
39
40    # encrypt file & write encrypted file -> void
41    def encFile(self, filename, filename_out):
42        # file handler
43        fh = FileHandler()
44        # bytes data
45        bd = fh.readFileReturnBytes(filename)
46        # bytes out
47        bo = bytearray()
48        # key number iterator
49        kni = 0
50        for b in bd:
51            bo.append(self.encrypt(b, self.key_num[kni], 256))
52            kni += 1
53            kni = kni % self.key_len
54        fh.writeFileByBytes(filename_out, bo)
55
56    # decrypt file & write decrypted file -> void
57    def decFile(self, filename, filename_out):
58        # file handler
```

```

59         fh = FileHandler()
60         # bytes data
61         bd = fh.readFileReturnBytes(filename)
62         # bytes out
63         bo = bytearray()
64         # key number iterator
65         kni = 0
66         for b in bd:
67             bo.append(self.decrypt(b, self.key_num[kni], 256))
68             kni += 1
69             kni = kni % self.key_len
70         fh.writeFileByBytes(filename_out, bo)
71
72     # generate key with len -> return string
73     def gk(self, len_exp):
74         # len key
75         lk = len(self.key)
76         # out key
77         ok = ''
78         # iterate char in string self.key
79         for i in range(len_exp):
80             ok += self.key[i % lk]
81
82         return ok
83
84     # encrypt vigenere -> return string
85     def ev(self, pt, is_extended = False):
86         # remove non-w character
87         if not is_extended:
88             pt = re.sub(r'\W', '', pt)
89         # len plain text
90         lpt = len(pt)
91         # new key
92         nk = self.gk(lpt)
93         # chiper text
94         ct = ''
95         # iterate char in string plain text and new key
96         for i in range(lpt):
97             ct += self.gvc(self.encrypt(self.gcv(pt[i], is_extended), self.gcv(nk[i], is_extended), (256 if is_exte
98
99         return ct if is_extended else ct.upper()
100
101     # decrypt vigenere -> return string
102     def dv(self, ct, is_extended = False):
103         # len chiper text
104         lct = len(ct)
105         # new key
106         nk = self.gk(lct)
107         # plain text
108         pt = ''
109         # iterate char in string plain text and new key
110         for i in range(lct):
111             pt += self.gvc(self.decrypt(self.gcv(ct[i], is_extended), self.gcv(nk[i], is_extended), (256 if is_exte
112
113         return pt if is_extended else pt.upper()
114
115
116

```