# python实现凯撒加密”实验报告

高一五班·吴宇杰

**一、实验目的**

加密技术是一种常用的安全保密手段，可以把重要的数据变成经过加密的乱码传送出去，到达目的地后再利用解密手段还原。通过运用python实现和改进“凯撒加密”，可以加深对加法密码的基本原理和算法的认识，提高信息安全的意识，学会使用程序设计语言实现简单算法，体验程序设计的基本流程。

**二、实验内容**

1、运用python实现“凯撒密码”加密。

2、运用python实现“移位密码”加密。

3、运用python实现“替换密码”加密。

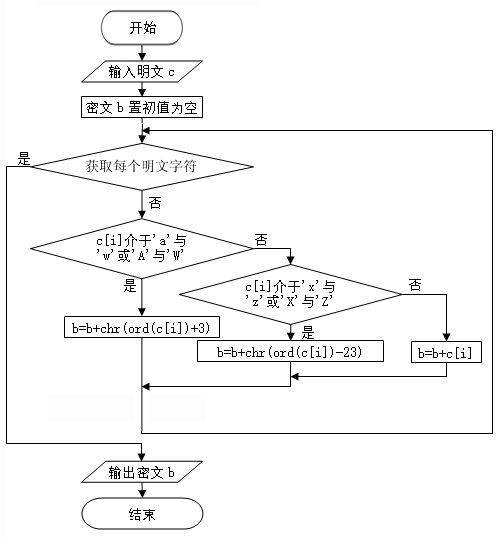
**三、实验步骤与结果**

**1、运用python实现“凯撒密码”加密**

（1）什么是“凯撒密码”？

密码的使用最早可以追溯到古罗马时期，苏托尼厄斯在公元2世纪写的《恺撒传》中对恺撒用过的一种加密技术进行了详细的介绍。“恺撒密码”只是简单地将明文中的每一个字母用字母表中该字母后的第3个字母替换。例如，将明文中的a用d替换，b用e替换，……，以此类推，X变成A，Y变成B，Z用C替换。这是一种很简单的加密方法，这种密码的密度是很低的。

（2）实现“凯撒密码”加密的算法流程图



（3）实现“凯撒密码”加密

源程序：

def caesar\_encode\_classical(word):

secret=''

for c in word:

if 'a'<= c <='w' or 'A'<= c <='W':

secret+=(chr(ord(c)+3))

elif 'x'<= c <='z' or 'X'<= c <='Z':

secret+=(chr(ord(c)-23))

elif c == ' ':

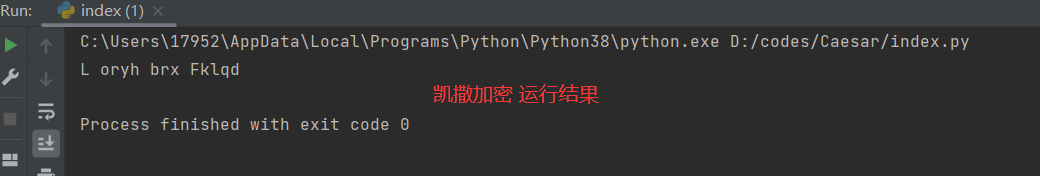
secret+=c

return secret

#使用例1

print(caesar\_encode\_classical('I love you China!'))

运行结果：



**2、运用python实现“移位密码”加密**

（1）什么是“加法密码”？

像凯撒密码这样，明文中的所有字母都在字母表向后（或向前）按照一个固定数目进行偏移后被替换成密文，这种密码称为加法密码，又称“移位密码”。这种加密方法可以依据移位的不同产生新的变化，字母表最多可以移动25位。移位密码的明文字母表向后或向前移动都是可以的，通常表述为向后移动，如果要向前移动1位，则等同于向后移动25位，位移选择为25即可。只是字母表移位置，保密性能仍然不好。将明文字符前移或后移一个固定的长度d（称为密钥），即使改变d的值，也最多只需25次尝试d的值，就能破解。

（2）实现“移位密码”加密

源程序：

def caesar\_encode\_plus(word,move):

secret=''

for c in word:

if 65<= ord(c) <=90 :#小写

i=ord(c)+move

while i > 90:

i-=26

secret+=chr(i)

elif 97<= ord(c) <=122:

i=ord(c)+move

while i >122:

i-=26

secret+=chr(i)

elif c == ' ':

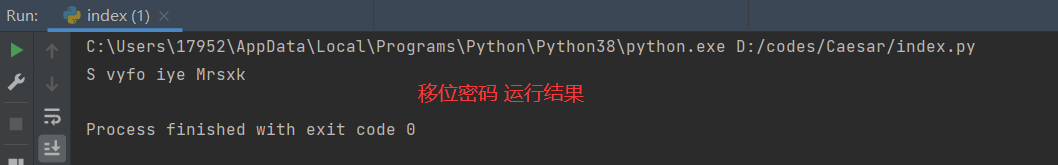
secret+=c

return secret

#使用例2

print(caesar\_encode\_plus('I love you China!',1024))

运行结果：



**3、运用python实现“替换密码”加密**

（1）什么是“替换密码”？

密码术可以大致别分为两种，即移位和替换。考虑到恺撒密码的安全性极差，应对算法进行改进。为了使密码有更高的安全性，一种可行的改进方式是，使用单字母替换密码，建立一个明文字符与密文字符之间的一一映射表，即“密表”，如：

明文：ABCDEFGHIJKLMNOPQRSTUVWXYZ

密文：QWERTYUIOPASDFGHJKLZXCVBNM

加密时，A→Q,B→W，…,解密时Q→A,W→B，…，小写字母对应法则也相同，即a→q,b→w,…。原先恺撒密码只有一个“密钥”，现在好比有25个“密钥”，这样保密性能大大提升，破解难度大大增加。另外，这个“密表”也是可以按需要改变的。

（2）实现“替换密码”加密

源程序：

功能代码①:实现随机生成一个”密表”

def mkkey():

import random

key={}

upkey=[]

lowkey = []

low=[]

for i in range(65,91):

upkey.append(chr(i))

for j in range(97,123):

lowkey.append(chr(j))

up=random.sample(upkey,26)

for u in up:

low.append(u.lower())

Upperkey=dict(zip(upkey,up))

Lowerkey = dict(zip(lowkey, low))

return [Upperkey,Lowerkey]

功能代码②:移位加密本体

def caesar\_encode\_proplus(word,dic):

secret=''

for w in word:

if 65<= ord(w) <=90:

secret+=dic[0][w]

elif 97<= ord(w) <=122:

secret+=dic[1][w]

elif w == ' ':

secret+=w

return secret

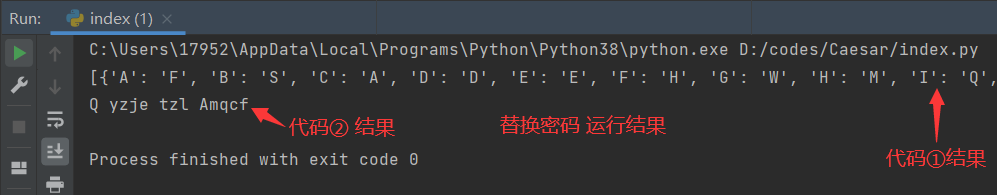
#使用例3

key=mkkey()

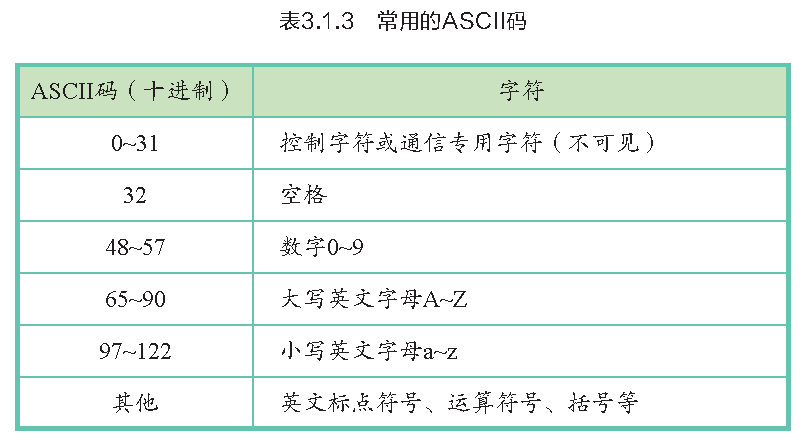
print(key)

print(caesar\_encode\_proplus('I love you China!',key))

运行结果：



**附1：**

****

**附2:源代码的GitHub链接(所有文件遵循mit开源协议)**

**源代码**

[jacksBlog/code.py at main · Jackwu945/jacksBlog (github.com)](https://github.com/Jackwu945/jacksBlog/blob/main/2021%E7%A0%94%E7%A9%B6%E6%80%A7%E5%AD%A6%E4%B9%A0/code.py)

本文档

[jacksBlog/report.md at main · Jackwu945/jacksBlog (github.com)](https://github.com/Jackwu945/jacksBlog/blob/main/2021%E7%A0%94%E7%A9%B6%E6%80%A7%E5%AD%A6%E4%B9%A0/report.docx)