**人工智能程序设计**

**python**

**实验报告**

**实验题目： 设计凯撒密码的加密和解密**

**专 业： 人工智能**

**姓 名： 叶薛华**

**学 号： 22920192204117**

**实验日期**： 2020.03.25

<正文内容字体为黑色宋体小四>

1. **实验目的**

1.熟悉python操作

2.熟悉凯撒密码的工作模式

**二. 实验内容**

编写一个程序，利用凯撒密码的工作模式对中文密码进行加密，并给出相应的解密程序。（在[密码学](https://baike.baidu.com/item/%E5%AF%86%E7%A0%81%E5%AD%A6" \t "https://baike.baidu.com/item/%E6%81%BA%E6%92%92%E5%AF%86%E7%A0%81/_blank)中，恺撒密码，或称恺撒加密、恺撒变换、变换加密，是一种最简单且最广为人知的加密技术。）

1. **实验思路**

加密过程：

1.在加密开始时，系统首先生成一个随机数，例如999

2.将系统生成的随机数转变成汉字，如999->九百九十九

3.用设定好的固定的数字对b对其加密，比如b=10000，那么他们就是chr(ord(i)+10000)

4.将转换好的文本连接在输入的密文之后

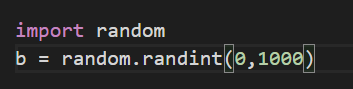
5.用该随机数对连接起来的密文进行加密

代码如下：

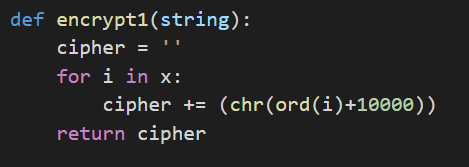
输入密文

{Q(JLH$)LUT]KSHKDWQ$`ZA

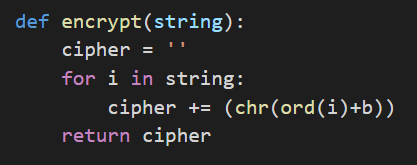
生成随机数用于加密



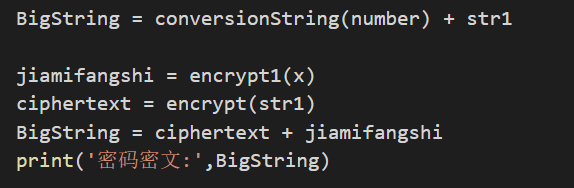
加密生成的随机数



加密密文



将密文和随机数连接，以及函数主体部分



解密过程：

1. 首先从密码中去掉固定位数的密文，只留下原先随机数变化来的部分
2. 将剩余部分利用固定的b进行转换
3. 将转换后的随机数换成阿拉伯数字形式
4. 把最初去掉的部分用转换后的数字进行解密

代码分块内容如下：

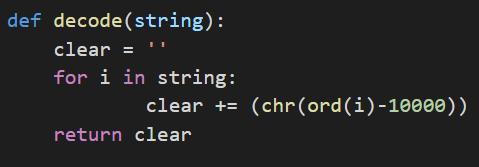
输入待解密内容

4LY(R{~5}}OO9W5ZSZ`L622

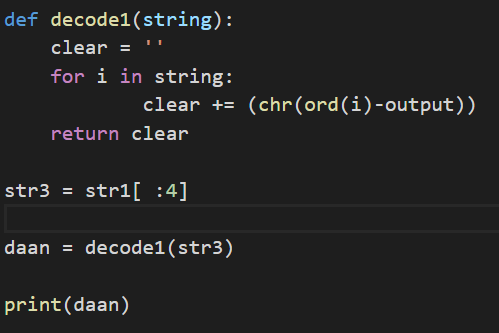
去掉密文，保留转化后的随机数

__XZKT`H_TB%ZF96DS%01%C

将随机数按确定规则还原

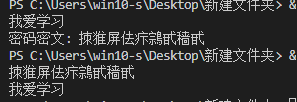


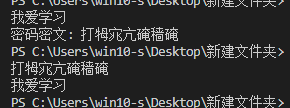
使用转换出来的随机数还原密文



两部分数字和汉字之间的转化内容过多，不放截图，源代码附在最后

1. **实验结果**

****

****

**五．实验总结**

1.掌握了凯撒密码的加密方式，而且在这次实验中体会了多种加密方式，也跟同学们交流了不同加密方式优缺点，最终选定了一种。对密码第一次有了直观的体会。

2.这次的代码是我用python完成的第一个非基础代码，对自己来说难度很大，不过也在同时巩固了上课讲过的各种操作。

附：源码如下：

加密——

str1 = input('')

import random

b = random.randint(0,1000)

number = b

def conversionString(intnumber):

    numberList = ['零','一','二','三','四','五','六','七','八','九']

    unitList = ["","十","百","千","万",'十万','百万','千万','亿','十亿','百亿','千亿','万亿','兆']

    # 转为字符串 获取传入字符串长度

    strnumber = str(intnumber)

    lennumber = len(strnumber)

    # 如果长度等于1 则直接返回对应的各位数字

    if lennumber == 1:

        return numberList[intnumber]

    # 如果不为一 也就不是个位数 则需要获取相关单位

    string = ''

    for i in range(lennumber):

        # print('第{}次,string值为:{}'.format(i,string))

        if int(strnumber[i]) != 0:

            # 判断万出现的次数 如果多次删除现有的 万 字 防止出现 五十万二万 重复

            for unit in ['万','亿']:

                if string.count(unit) > 1:

                    print(string.count(unit))

                    string = string.replace(unit, '',1)

            # 获取当前数字对应的汉字 + 单位

            string = string + numberList[int(strnumber[i])]+unitList[lennumber - i - 1]

        # 如果前一位也是零 那么直接跳出循环重新执行 //防止

        elif int(strnumber[i - 1]) == 0:

                continue

        else:

            # 如果都不是 也就是为 那么则直接加一个零

            string = string+numberList[int(strnumber[i])]

    # 返回值 // rstrip 删除结尾的所有零

    return string.rstrip('零')

x = conversionString(number)

def encrypt1(string):

    cipher = ''

    for i in x:

        cipher += (chr(ord(i)+10000))

    return cipher

def encrypt(string):

    cipher = ''

    for i in string:

        cipher += (chr(ord(i)+b))

    return cipher

BigString = conversionString(number) + str1

jiamifangshi = encrypt1(x)

ciphertext = encrypt(str1)

BigString = ciphertext + jiamifangshi

print('密码密文:',BigString)

解密——

str1 = input('')

str2 = str1[4: ]

def decode(string):

    clear = ''

    for i in string:

            clear += (chr(ord(i)-10000))

    return clear

jiamifangshi = decode(str2)

number\_map = {

    "零": 0,

    "一": 1,

    "二": 2,

    "三": 3,

    "四": 4,

    "五": 5,

    "六": 6,

    "七": 7,

    "八": 8,

    "九": 9

}

unit\_map = {

    "十": 10,

    "百": 100,

    "千": 1000,

    "万": 10000,

    "亿": 100000000

}

def forward\_cn2an\_one(inputs):

    output = 0

    unit = 1

    num = 0

    for index, cn\_num in enumerate(inputs):

        if cn\_num in number\_map:

            # 数字

            num = number\_map[cn\_num]

            # 最后的个位数字

            if index == len(inputs) - 1:

                output = output + num

        elif cn\_num in unit\_map:

            # 单位

            unit = unit\_map[cn\_num]

            # 累加

            output = output + num \* unit

            num = 0

        else:

            raise ValueError(f"{cn\_num} 不在转化范围内")

    return output

output = forward\_cn2an\_one(jiamifangshi)

def decode1(string):

    clear = ''

    for i in string:

            clear += (chr(ord(i)-output))

    return clear

str3 = str1[ :4]

daan = decode1(str3)

print(daan)