**西安电子科技大学网信院**

**信息安全基础与密码学**

**综合实验**

**实 验 报 告（三）**

**基于中国剩余定理的秘密共享方案**

**班级：**

**姓名：**

**学号：**

**日期：2022年11月13日星期日**

一、实验目的（包括实验环境、实现目标等等）

实验环境

Windows11 , Python 3.9.12

实验目标：

1.通过算法编程，熟悉大整数的基本函数操作

2.通过编程实现基于中国剩余定理的秘密共享方案，加深对于中国剩余定理的理解与运用

二、方案设计

（包括背景、原理、必要的公式、图表、算法步骤等等）

实验背景

秘密共享是将秘密以适当的方式拆分，拆分后的每一个子秘密由不同的参与者管理，单个参与者无法恢复秘密信息，只有若干个参与者一同协作才能恢复秘密消息。并且，当其中某些参与者出问题时，秘密仍可以恢复。

实验原理

1.  门限,一个秘密 , 被分隔成个子秘密

对于某个秘密,计算 则子秘密为。

其中对  具有一定的要求：

要求一 ：选择个整数 , 满足

(1) 严格递增

(2) 两两互素

(3) ,有 

要求二: 

2. 个子秘密中任意选择个, , 恢复出秘密

计算  恢复出秘密。

3. 个子秘密中任意选择个,  ，恢复出秘密k

计算  恢复出秘密。

由于，因此模得出的结果并非k。

算法步骤

(1) 输入子秘密数，选择解密数秘密,以及秘密

(2) 随机生成个位到位的两两互素的大整数, 作为我们选择的放入表

(3) 将生成的按照升序排序

(4) 计算,判断是否满足，若满足则输出，否则回到(2)

(5) 计算组成密钥对

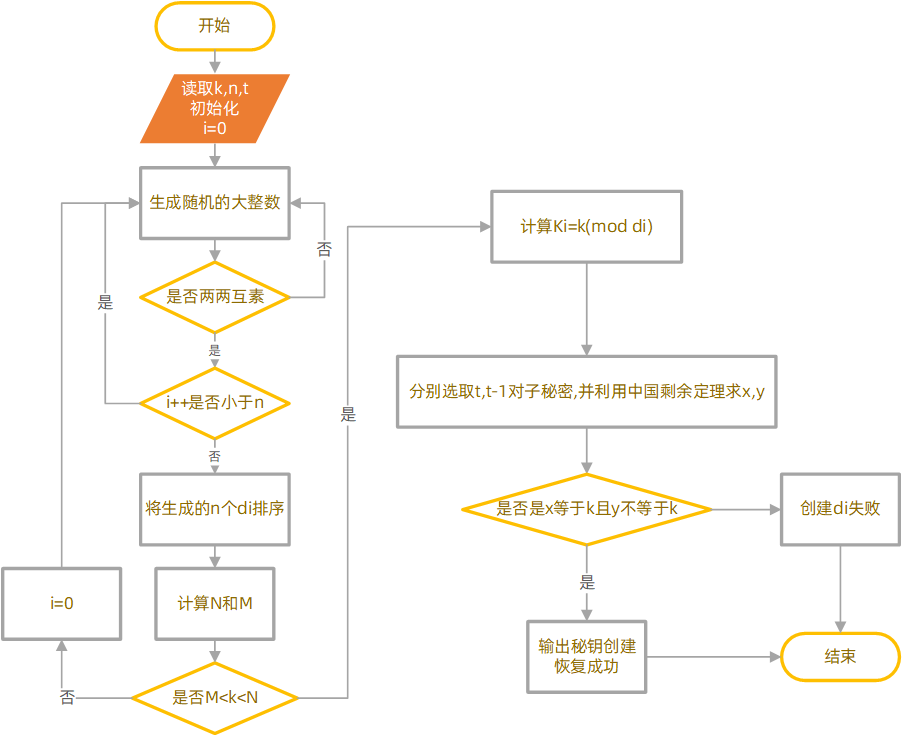
(6) 分别任选和个子秘密,利用中国剩余定理进行恢复出

(7) 比较恢复的秘密是否与原秘密相同,且恢复出的秘密与原秘密不同

三、方案实现

（包括算法流程图、主要函数的介绍、算法实现的主要代码等等）

算法流程图



主要函数

is\_coprime() 判断列表是否互素

find\_d()寻找di子秘密列表

generate\_key() 判断是否生成合适的di,并输出M,N

encrypt()加密函数，生成子秘密对

decrypt()解密函数，生成原消息

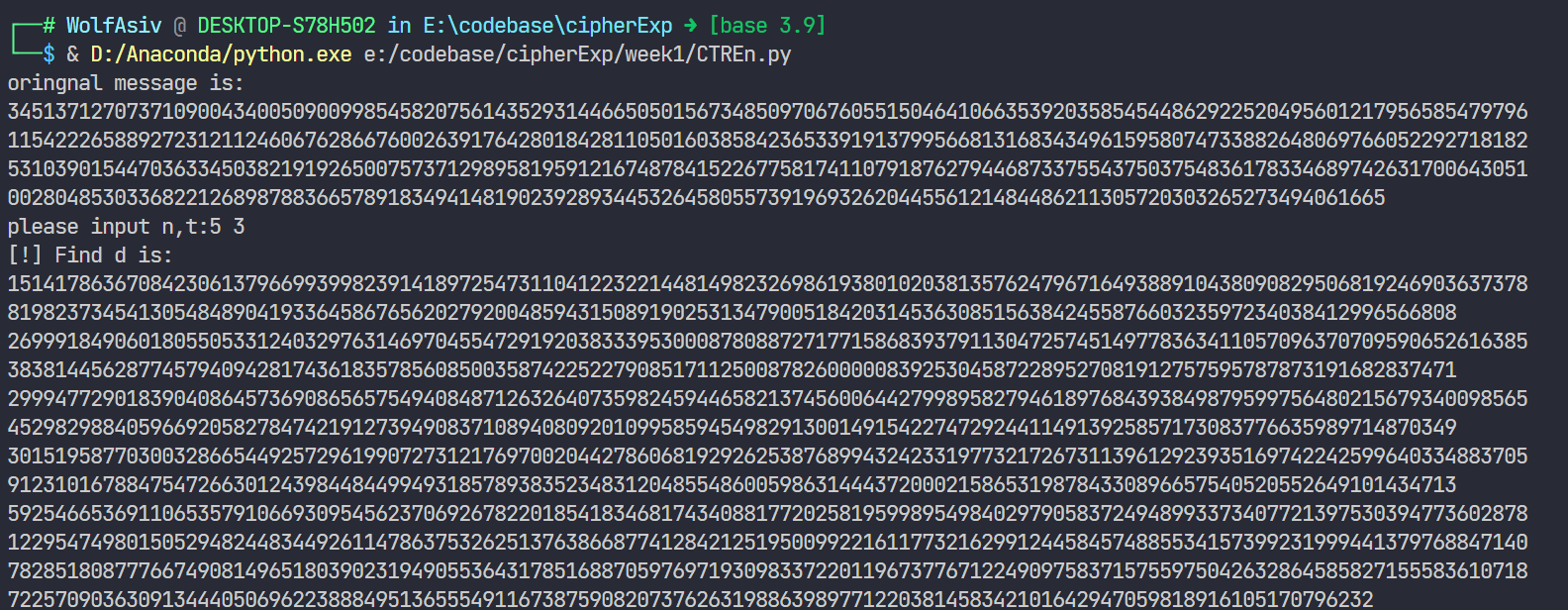
chinese\_reminder() 中国剩余定理

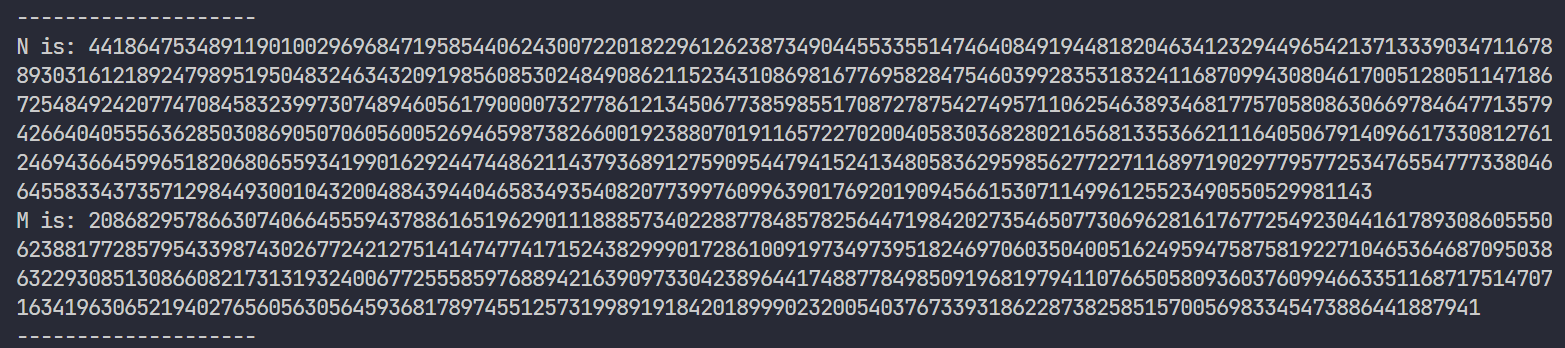
实现代码

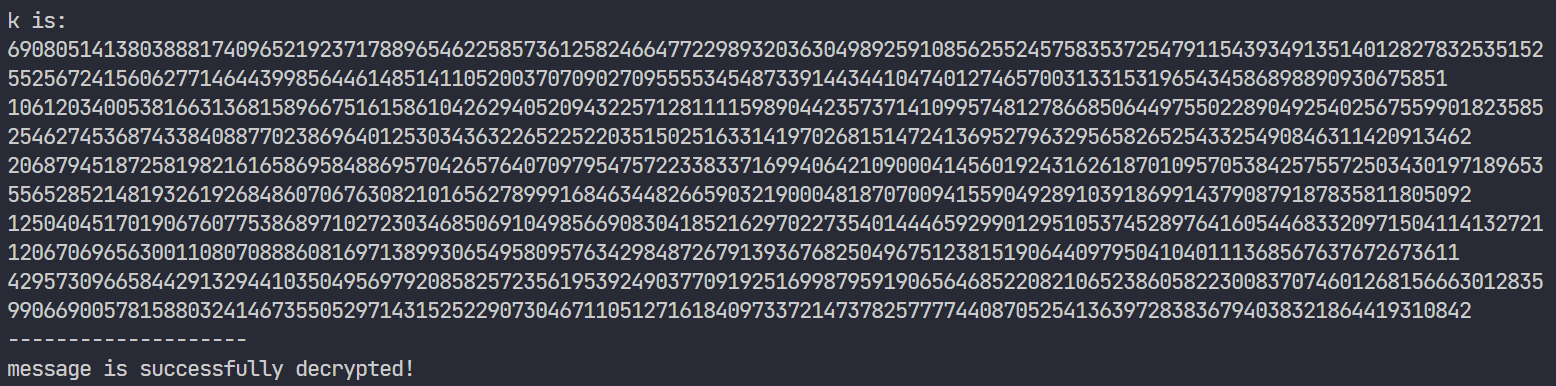
1. *#基于中国剩余定理的(t,n)门限秘钥共享方案*
2. *#挑选五个素数，满足严格递增*
3. import random
4. import gmpy2
5. def is\_coprime(list,bignum):
6. for i in list:
7. if gmpy2.gcd(i,bignum)!=1:
8. return False
9. return True
10. def find\_d(n):
11. d=[]*#初始化d*
12. *#向下取整*
13. minlen=int(500/t)
14. maxlen=int(500/(t-1))
15. while len(d)<n:
16. temp=random.randint(pow(10,minlen),pow(10,maxlen))
17. if is\_coprime(d,temp):
18. d.append(temp)
19. d.sort()
20. return d*#返回一个递增的素数列表*
21. def generate\_key(message,n,t):
22. while True:
23. d=find\_d(n)
24. M=1
25. N=1
26. for i in range(t):
27. N\*=d[i]
28. for i in range(len(d)-t+1,len(d)):
29. M\*=d[i]
30. if M<message and message<N:
31. break
32. print("[!] Find d is:")
33. for i in d:
34. print(i)
35. print("--------------------")
36. print("N is:",N)
37. print("M is:",M)
38. print("--------------------")
39. return M,N,d
40. def encrypt(message,d):
41. k=[]
42. for i in d:
43. k.append(((message%i),i))
44. print("k is:")
45. for i,j in k:
46. print(i)
47. print("--------------------")
48. return k
49. def chinese\_remainder(data):
50. *#判断是否互质*
51. for i in range(len(data)):
52. for j in range(i+1,len(data)):
53. if gmpy2.gcd(data[i][1],data[j][1])!=1:
54. return -1,-1
55. *#计算n的乘积*
56. N=1
57. for a,m in data:
58. N\*=m
59. *#计算Ni*
60. Ni=[]
61. for a,m in data:
62. Ni.append(N//m)
63. *#计算Ni的逆元*
64. Ni\_inv=[]
65. for i in range(len(Ni)):
66. Ni\_inv.append(gmpy2.invert(Ni[i],data[i][1]))
67. *#计算x*
68. x=0
69. for i in range(len(Ni)):
70. x+=data[i][0]\*Ni[i]\*Ni\_inv[i]
71. x=x%N
72. return x
73. def decrypt(k,t):
74. *#任选t个k中的元素*
75. key\_x=random.sample(k,t)
76. x=chinese\_remainder(key\_x)
77. *# print("the decrypted message is:")*
78. *# print(x)*
79. key\_y=random.sample(k,t-1)
80. y=chinese\_remainder(key\_y)
81. return x,y
82. if \_\_name\_\_ == '\_\_main\_\_':
83. message=34513712707371090043400509009985458207561435293144665050156734850970676055150464106635392035854544862922520495601217956585479796115422265889272312112460676286676002639176428018428110501603858423653391913799566813168343496159580747338826480697660522927181825310390154470363345038219192650075737129895819591216748784152267758174110791876279446873375543750375483617833468974263170064305100280485303368221268987883665789183494148190239289344532645805573919693262044556121484486211305720303265273494061665
84. print("oringnal message is:")
85. print(message)
86. *#输入n,t*
87. n,t=map(int,input("please input n,t:").split())
88. *# print(n)*
89. *# print(t)*
90. M,N,d=generate\_key(message,n,t)
91. k=encrypt(message,d)
92. x,y=decrypt(k,t)
93. if(x==message and y!=message):
94. print("message is successfully decrypted!")
95. else:
96. print("encryption failed!")

四、数据分析(包括算法测试数据的分析，运行结果截图等等)

运行结果







五、思考与总结

1. 在基于中国剩余定理的(*t*, *n*)秘密共享方案中，少于*t*个子秘密，是否能够正确恢复出秘密？请简述原因。

答:不能，例如:

个子秘密中任意选择个,  ，恢复出秘密k

计算  恢复出秘密。

由于，因此模得出的结果并非k,更小的t个也是如此。

1. 实验过程中还遇到了什么问题，如何解决的？通过该实验有何收获？

答:遇到无法确定di的范围，算法效率可能很低，如果范围跨度很大的话，因此我们大致计算一下，因为密文是500的大数，假设子秘钥有位，di的位数选择为：

