

天津大学本科生实验报告专用纸

学院 智能与计算学部 年级 2017 专业 软件工程 班级 2班 姓名 侯雨茜 学号 3017218092

课程名称 信息安全技术 实验日期 2019-10-22 成绩

同组实验者

实验 1: DES 加密算法实现

一、实验目的

实现 DES 算法并掌握其原理。

二、实验环境

- 操作系统: Mac OS
- 实现语言: Python

三、实验内容

- 实现 16 轮 DES 加密算法
- DES 算法正确性检验
- 指定任意 64 比特明文 x, 指定任意 64 比特密钥 k, 完成 3 个问题

四、实验步骤

4.1 实现 16 轮 DES 加密算法

4.1.1 算法代码如下:

```
DES.py
1  # S盒置换函数 48位->32位
2  # 函数说明: s为48位数据
3  # 返回值为32位
4  def SBoxes(x):
5      # S盒置换表
6      s = [
7          [
8              [14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7],
9              [0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8],
10             [4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0],
11             [15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13],
12         ],
13         [
14             [15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10],
15             [3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5],
16             [0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15],
17             [13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9],
18         ],
19     ]
```

天津大学本科生实验报告专用纸

```
DES.py
19  [
20      [10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8],
21      [13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1],
22      [13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7],
23      [1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12],
24  ],
25  [
26      [7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15],
27      [13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9],
28      [10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4],
29      [3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14],
30  ],
31  [
32      [2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9],
33      [14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6],
34      [4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14],
35      [11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3],
36  ],
37  [
38      [12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11],
39      [10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8],
40      [9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6],
41      [4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13],
42  ],
43  [
44      [4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1],
45      [13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6],
46      [1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2],
47      [6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12],
48  ],
49  [
50      [13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7],
51      [1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2],
52      [7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8],
53      [2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11],
54  ]
55  ]
56  x = str(bin(int(x, 2))[2:]).zfill(48)
57  y = ""
58  for i in range(8):
59      t = int(x[i * 6] + x[i * 6 + 5], 2)
60      u = int(x[i * 6 + 1: i * 6 + 5], 2)
61      y += str(bin(s[i][t][u])[2:]).zfill(4))
62
63  return y
64
65  # 密钥初始置换表
66  PC1 = [57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18, 10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36, 63,
67         55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22, 14, 6, 61, 53, 45, 37, 29, 21, 13, 5, 28, 20, 12, 4]
68
69  # 密钥压缩置换表
70  PC2 = [14, 17, 11, 24, 1, 5, 3, 28, 15, 6, 21, 10, 23, 19, 12, 4, 26, 8, 16, 7, 27, 20, 13, 2, 41, 52, 31, 37, 47, 55,
71         30, 40, 51, 45, 33, 48, 44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32]
72
73  # 数据初始置换表IP
74  IP = [58, 50, 42, 34, 26, 18, 10, 2, 60, 52, 44, 36, 28, 20, 12, 4, 62, 54, 46, 38, 30, 22, 14, 6, 64, 56, 48, 40, 32,
75        24, 16, 8, 57, 49, 41, 33, 25, 17, 9, 1, 59, 51, 43, 35, 27, 19, 11, 3, 61, 53, 45, 37, 29, 21, 13, 5, 63, 55, 47,
76        39, 31, 23, 15, 7]
77
78  # 数据扩展表
79  EP = [32, 1, 2, 3, 4, 5, 4, 5, 6, 7, 8, 9, 8, 9, 10, 11, 12, 13, 12, 13, 14, 15, 16, 17, 16, 17, 18, 19, 20, 21, 20, 21,
80        22, 23, 24, 25, 24, 25, 26, 27, 28, 29, 28, 29, 30, 31, 32, 1]
81
82  # P盒置换表
83  PT = [16, 7, 20, 21, 29, 12, 28, 17, 1, 15, 23, 26, 5, 18, 31, 10, 2, 8, 24, 14, 32, 27, 3, 9, 19, 13, 30, 6, 22, 11, 4,
84        25]
85
86  # 最终置换表
87  FP = [40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47, 15, 55, 23, 63, 31, 38, 6, 46, 14, 54, 22, 62, 30, 37, 5, 45, 13, 53,
88        21, 61, 29, 36, 4, 44, 12, 52, 20, 60, 28, 35, 3, 43, 11, 51, 19, 59, 27, 34, 2, 42, 10, 50, 18, 58, 26, 33, 1,
89        41, 9, 49, 17, 57, 25]
90
91  # P盒置换函数 32位->32位
92  def PBox(a, x):
93      s = len(a)
94      y = ""
95      for i in range(s):
96          y += x[a[i] - 1]
97      return y
98
99
100
101  # 密钥初始置换函数 64位->58位
102  # 函数说明: x为64位的初始密钥
103  # 返回值为58位
104  def PerChol(x):
105      return PBox(PC1, x)
```

教师签字:

年 月 日

```

DES.py x
105     return PBox(PC1, x)
106
107
108     # 密钥压缩置换函数 56位->48位
109     # 函数说明: x为56位的密钥
110     # 输出为48位的子密钥
111     def PerCho2(x):
112         return PBox(PC2, x)
113
114
115     # 明文初始置换函数 64位->64位
116     # 函数说明: x为初始明文 64位
117     # 返回值为6位
118     def InitPer(x):
119         return PBox(IP, x)
120
121
122     # 最终置换函数 64位->64位
123     # 函数说明: x为完成最后一轮循环得到的64为数据
124     # 扩展成48位的输出
125     def FinalPer(x):
126         return PBox(FP, x)
127
128
129     # 数据扩展函数 32->48
130     # 函数说明: x为数据的右半部分 32位
131     # 扩展成48位的输出
132     def ExpPer(x):
133         return PBox(EP, x)
134
135
136     def PerTabl(x):
137         return PBox(PT, x)
138
139
140     # 异或运算函数
141     # 要求位数相同
142     def XOR(x, y):
143         x = int(x, 2)
144         y = int(y, 2)
145         z = x ^ y
146         return str(bin(z)[2:].zfill(32))
147
148
DES.py x
148
149     # 16进制转2进制函数
150     # 函数说明: x为16进制字符串
151     # 返回为2进制字符串
152     def HexToBin(x):
153         n = len(x) * 4
154         x = str(bin(int(x, 16))[2:].zfill(n))
155         return x
156
157
158     # 2进制转16进制函数
159     # y为2进制字符串
160     # 返回值为16进制字符串
161     def BinToHex(y):
162         return str(hex(int(y, 2))[2:].upper())
163
164
165     # 密钥循环左移函数 56位->56位
166     # 函数说明: x为密钥
167     # 返回值位数不变
168     def LeftCircularShift(x):
169         l = int(len(x) / 2)
170         y = ""
171         for i in range(l - 1):
172             y += x[i + 1]
173         y += x[0]
174         for i in range(l, 2 * l - 1):
175             y += x[i + 1]
176         y += x[l]
177         return y
178
179
180     def Round(x, k):
181         l0 = x[:32]
182         r0 = x[32:]
183         l1 = r0
184         ep = ExpPer(r0)
185         xo = XOR(ep, k)
186         sb = SBoxes(xo)
187         p = PerTabl(sb)
188         r1 = XOR(p, l0)
189         y = l1 + r1
190         return y
191
192

```

```

DES.py x
192
193 def swap32bit(x):
194     L0 = x[:32]
195     R0 = x[32:]
196     y = R0 + L0
197     return y
198
199
200 Rots = [1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 1]
201
202
203 # DES加密函数 64位->64位
204 # 函数说明: text为64位的给定明文, key为密钥
205 # 返回值为64位的密文
206 def Encrypt(key, text):
207     key = PerCho1(key)
208     text = InitPer(text)
209     for i in range(16):
210         for j in range(Rots[i]):
211             key = LeftCircularShift(key)
212             k = PerCho2(key)
213             text = Round(text, k)
214         text = swap32bit(text)
215     text = FinalPer(text)
216     return text
217
218
219 print('请输入16进制明文: ')
220 text = input()
221 print('请输入密钥: ')
222 key = input()
223 print('请输入加密次数: ')
224 itr = int(input())
225
226 text = HexToBin(text)
227 print('2进制明文为: ')
228 print(text)
229
230 key = HexToBin(key)
231 print('2进制密钥为: ')
232 print(key)
233
234 for i in range(itr):
235     text = Encrypt(key, text)
236
237
238 for i in range(itr):
239     text = Encrypt(key, text)
240     text = BinToHex(text).zfill(16)
241     print('加密后的密文为: ')
242     print(text)

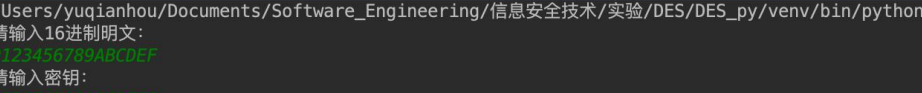
```

4.2 DES 算法正确性检验

4.2.1 请依据下面的实验结果检验程序的正确性:

1. 设 16 进制明文为 0123456789ABCDEF
2. 密钥 K 为: 123456789ABCDEF0
3. 加密后的密文为: 85E813540F0AB405

4.1 节程序运行结果如下:



```
Run: DES
/Users/yuqianhou/Documents/Software_Engineering/信息安全技术/实验/DES/DES_py/venv/bin/python /Users/yuqianhou/Documents/Software_Engineering/信息安全技术/实验/DES/DES_py/venv/bin/python DES.py
请输入16进制明文:
0123456789ABCDEF
请输入密钥:
123456789ABCDEF0
请输入加密次数:
1
2进制明文为:
00000001001000110100010101100111110001001101010111100110111101111
2进制密钥为:
00010010001101000101011001111000100110101111001101111011110000
加密后的密文为:
85E813540F0AB405

Process finished with exit code 0
```

程序正确。

4.3 指定任意 64 比特明文 x，指定任意 64 比特密钥 k，完成 3 个问题

4.3.1 随机改变明文 x 的一个比特位，即 1 变 0，0 变 1，得到 x'。

测试 $y = \text{DES}_k(x)$ 和 $y' = \text{DES}_k(x')$ 中有多少位不同，即 $y \oplus y'$ 中“1”的个数。

修改代码，使程序可以自动计算所需要的值：

```
203 # DES加密函数 64位->64位
204 # 函数说明: text为64位的给定明文, key为密钥
205 # 返回值为64位的密文
206 def Encrypt(key, text):
207     key = PerCho1(key)
208     text = InitPer(text)
209     print('请输入需要加密的轮数 (1 <= i <= 16) : ')
210     num = int(input())
211     for i in range(num):
212         for j in range(Rots[i]):
213             key = LeftCircularShift(key)
214             k = PerCho2(key)
215             text = Round(text, k)
216             text = swap32bit(text)
217             text = FinalPer(text)
218     return text
219
220
221 # print('请输入16进制明文x: ')
222 # text = input()
223 text = 'EB3B4B887DF349EF'
224 print('16进制明文x = ' + text)
225 # print('请输入密钥k: ')
226 # key = input()
227 key = '4F6556C45580DAD9'
228 print('密钥k = ' + key)
229 # print('请输入加密次数: ')
230 # itr = int(input())
231 itr = 1
232
233 text = HexToBin(text)
234 textBin = text
235 print('2进制明文为: ')
236 print(text)
237
238 key = HexToBin(key)
239 # print('2进制密钥为: ')
240 # print(key)
241
242 for i in range(itr):
243     text = Encrypt(key, text)
244     textHex = BinToHex(text).zfill(16)
245     print('加密后的密文z为: ')
```

```
246 print(textHex)
247 print('2进制密文z为: ')
248 # text0 = int(text)
249 text0 = int(text, 2)
250 print(text)
251
252
253 # print('请输入修改后的16进制明文x\': ')
254 # text = input()
255 text = 'EB3B4B987DF349EF'
256 print('修改后的16进制明文x\' = ' + text)
257 # print('请输入加密次数: ')
258 # itr = int(input())
259
260 text = HexToBin(text)
261 print('2进制明文为: ')
262 print(text)
263
264 # key = HexToBin(key)
265 # # print('2进制密钥为: ')
266 # # print(key)
267
268 # for i in range(16):
269 #     for j in range(64):
270
271     for k in range(itr):
272         text = Encrypt(key, text)
273         textHex = BinToHex(text).zfill(16)
274         print('加密后的密文z\'为: ')
275         print(textHex)
276         print('2进制密文z\'为: ')
277         # text1 = int(text)
278         text1 = int(text, 2)
279         print(text)
280
281     # 异或运算
282     xorz = text0 ^ text1
283     # 十进制转换为二进制
284     xorzBin = str(bin(xorz))
285     print('z^z\'为: ')
286     print(xorzBin)
287     tmp = xorzBin.count('1')
288     print('其中, z^z\'中1的个数为: ' + str(tmp))
289
```

设任意 64 比特明文为:

$x = \text{EB3B4B887DF349EF}_{(16)}$
 $= 1110\ 1011\ 0011\ 1011\ 0100\ 1011\ 1000\ 1000\ 0111\ 1101\ 1111\ 0011\ 0100\ 1001\ 1110\ 1111_{(2)}$

任意 64 比特密钥为:

$k = \text{4F6556C45580DAD9}_{(16)}$
 $= 0100\ 1111\ 0110\ 0101\ 0101\ 0110\ 1100\ 0100\ 0101\ 0101\ 1000\ 0000\ 1101\ 1010\ 1101\ 1001_{(2)}$

生成的密文为:

$y = \text{94581A6CF1D89B98}_{(16)}$
 $= 1001\ 0100\ 0101\ 1000\ 0001\ 1010\ 0110\ 1100\ 1111\ 0001\ 1101\ 1000\ 1001\ 1011\ 1001\ 1000_{(2)}$

任意改变明文为:

$x' = 1110\ 1011\ 0011\ 1011\ 0100\ 1011\ 1001\ 1000\ 0111\ 1101\ 1111\ 0011\ 0100\ 1001\ 1110\ 1111_{(2)}$
 $= \text{EB3B4B987DF349EF}_{(16)}$

生成的密文为:

$y' = \text{AA7844FAE405A113}_{(16)}$
 $= 1010\ 1010\ 0111\ 1000\ 0100\ 0100\ 1111\ 1010\ 1110\ 0100\ 0000\ 0101\ 1010\ 0001\ 0001\ 0011_{(2)}$
因此 $y \oplus y' = 0011\ 1110\ 0010\ 0000\ 0101\ 1110\ 1001\ 0110\ 0001\ 0101\ 1101\ 1101\ 0011\ 1010\ 1000\ 1011$,
其中, “1”的个数为 32 个。
程序运行结果如下:

```
DES x
/Users/youqianhou/Documents/Software_Engineering/信息安全技术/实验/DES
请输入16进制明文x:
EB3B4B887DF349EF
请输入密钥k:
4F6556C45580DAD9
请输入加密次数:
1
2进制明文为:
1110101100111011010010111000100001111101111100110100100111101111
请输入需要加密的轮数 (i <= i <= 16) :
16
加密后的密文z为:
94581A6CF1D89B98
2进制密文z为:
100101000101100000011010011011001111000110110001001101110011000
请输入修改后的16进制明文x':
EB3B4B987DF349EF
请输入加密次数:
1
2进制明文为:
1110101100111011010010111001100001111101111100110100100111101111
请输入需要加密的轮数 (i <= i <= 16) :
16
加密后的密文z'为:
AA7844FAE405A113
2进制密文z'为:
101010100111100001000100111110101110010000001011010000100010011
z^z'为:
0b11111000100000010111101001011000010101110111010011101010001011
其中, z^z'中1的个数为: 32
```

4.3.2 设 $\text{DES}_k^i(x)$ 是 DES 经过 i 轮加密后的结果, 其中 $1 \leq i \leq 16$.
对 $i = 1, \dots, 16$ 分别测试 $z = \text{DES}_k^i(x)$ 与 $z' = \text{DES}_k^i(x)$ 有多少位不同。找到最小的 i 值, 使得 $z \oplus z'$ 中“1”的个数约等于 32。
仍然以上一题的 x, x' 和 key 值为例:
当 $i = 1$ 时, $z \oplus z'$ 中“1”的个数为 1;
当 $i = 2$ 时, $z \oplus z'$ 中“1”的个数为 7;
当 $i = 3$ 时, $z \oplus z'$ 中“1”的个数为 20;
当 $i = 4$ 时, $z \oplus z'$ 中“1”的个数为 30;
当 $i = 5$ 时, $z \oplus z'$ 中“1”的个数为 36;
当 $i = 6$ 时, $z \oplus z'$ 中“1”的个数为 36;

当 $i = 7$ 时, $z \oplus z'$ 中“1”的个数为 31;
当 $i = 8$ 时, $z \oplus z'$ 中“1”的个数为 34;
当 $i = 9$ 时, $z \oplus z'$ 中“1”的个数为 36;
当 $i = 10$ 时, $z \oplus z'$ 中“1”的个数为 35;
当 $i = 11$ 时, $z \oplus z'$ 中“1”的个数为 30;
当 $i = 12$ 时, $z \oplus z'$ 中“1”的个数为 26;
当 $i = 13$ 时, $z \oplus z'$ 中“1”的个数为 31;
当 $i = 14$ 时, $z \oplus z'$ 中“1”的个数为 29;
当 $i = 15$ 时, $z \oplus z'$ 中“1”的个数为 28;
当 $i = 16$ 时, $z \oplus z'$ 中“1”的个数为 32;
因此, 当 i 最小为 4 时, 1 的个数已经接近于 32。
运行结果如下:

```
DES x
/Users/youqianhou/Documents/Software_Engineering/信息安全技术/实验/DES
16进制明文x = EB3B4B887DF349EF
密钥k = 4F6556C45580DAD9
2进制明文为:
1110101100111011010010111000100001111101111100110100100111101111
请输入需要加密的轮数 (1 <= i <= 16) :
4
加密后的密文z为:
E8D44E6896C23CD3
2进制密文z为:
1110100011010100010011100110100010010110110000100011110011010011
修改后的16进制明文x' = EB3B4B987DF349EF
2进制明文为:
1110101100111011010010111001100001111101111100110100100111101111
请输入需要加密的轮数 (1 <= i <= 16) :
4
加密后的密文z'为:
0C7F8B73F55B7DB7
2进制密文z'为:
0000110001111111100010110111001111110101010110110111110110110111
z^z'为:
0b1110010010101011110001010001101101100011100110010100000101100100
其中, z^z'中1的个数为: 30
```

4.3.3 对明文 x 的 64 个不同的位置分别进行试验 ii.
假设 x' 是 x 第 j 个位置取反后的结果, 即 1 变 0, 0 变 1. 对固定的 i , 记 $z = \text{DES}_k^i(x)$, $z' = \text{DES}_k^i(x)$.
我们记 $z \oplus z'$ 中“1”的个数为 $e(i, j)$. 计算 $e_i = (1/64) \sum_{j=1}^{64} e(i, j)$, $1, 2, \dots, 16$. 找到最小的 i 值, 使得 e_i 约等于 32。
仍然以上一题的 x 和 key 值为例, 修改代码, 使程序可以自动计算所需要的值:

```
DES.py
221 # 转换明文的第num个比特位
222 def BitTrans(text, num):
223     tmp = ''
224     for i in range(0, num):
225         tmp += text[i]
226     if text[num] == '0':
227         tmp += '1'
228     else:
229         tmp += '0'
230     for i in range(num+1, len(text) - 1):
231         tmp += text[i]
232     return tmp

DES.py
280 ## # print(key)
281
282 sum = 0
283 for i in range(1, 16):
284     for j in range(64):
285         textModified = BitTrans(text, j)
286         textModified = HexToBin(textModified)
287         textModified = Encrypt(key, textModified, i)
288         text1 = int(textModified, 2)
289         xorz = text0 ^ text1
290         xorzBin = str(bin(xorz))
291         tmp = xorzBin.count('1')
292         sum += tmp
293         print('改变明文第 ' + str(j+1) + ' 比特位后, 经过 ' + str(i) + ' 轮迭代加密后, 密文有 ' + str(tmp) + ' 位不同')
294     sum = sum / 64.0
295     print('经过 ' + str(i) + ' 轮迭代加密后 ' + 'e(' + str(i) + ') = ' + str(sum))
296     if(sum >= 30 and sum <= 34):
297         minNum = i
298         break
299
300 print('满足题意的最小的i值为 ' + str(minNum))
301
```

程序运行结果如下:

```
DES
/Users/yuqianhou/Documents/Software_Engineering/信息安全技术/实验/DES/D
16进制明文x = EB3B4B887DF349EF
密钥k = 4F6556C45580DAD9
2进制明文为:
1110101100111011010010111000100001111101111100110100100111101111
加密后的密文z为:
94581A6CF1D89B98
2进制密文z为:
1001010001011000000110100110110011110001110110001001101110011000
改变明文第 1 比特位后, 经过 1 轮迭代加密后, 密文有 27 位不同
改变明文第 2 比特位后, 经过 1 轮迭代加密后, 密文有 29 位不同
改变明文第 3 比特位后, 经过 1 轮迭代加密后, 密文有 27 位不同
改变明文第 4 比特位后, 经过 1 轮迭代加密后, 密文有 29 位不同
改变明文第 5 比特位后, 经过 1 轮迭代加密后, 密文有 29 位不同
改变明文第 6 比特位后, 经过 1 轮迭代加密后, 密文有 29 位不同
改变明文第 7 比特位后, 经过 1 轮迭代加密后, 密文有 27 位不同
改变明文第 8 比特位后, 经过 1 轮迭代加密后, 密文有 29 位不同
改变明文第 9 比特位后, 经过 1 轮迭代加密后, 密文有 27 位不同
改变明文第 10 比特位后, 经过 1 轮迭代加密后, 密文有 27 位不同
```

.....

```
改变明文第 55 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 56 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 57 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 58 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 59 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 60 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 61 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 62 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 63 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
改变明文第 64 比特位后, 经过 7 轮迭代加密后, 密文有 31 位不同
经过 7 轮迭代加密后 e(7) = 31.790659613238404
满足题意的最小的i值为 7
```

使得 e_i 约等于 32 的最小的 i 值为 7。

五、结论

通过本次实验的实践和学习，我掌握了使用 python 语言编写 DES 加密算法的方法。同时，我还在对某一比特位的改变的测试过程中发现，仅需经过几轮的迭代加密，就可以使改变后的密文与原始的密文有 30-34 位的不同。

源代码详见 DES.py 文件。

--	--	--