BÀI THỰC HÀNH SỐ 1

Môn: MẬT MÃ & AN NINH MẠNG

-o0o-

**Họ tên: Lê Bảo Khánh**

**MSSV: 1911363**

**Nhóm: L01**

**Phần 1. Các hệ mã đối xứng truyền thống**

**Câu 1:**

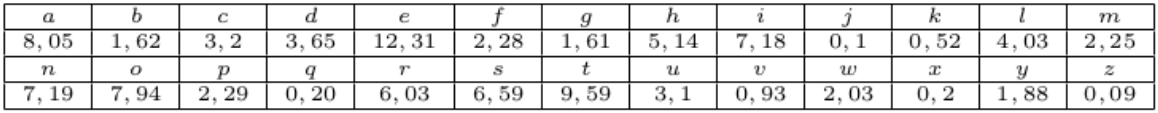
Ciphertext: **KNXMNSLKWJXMBFYJWGJSIXFIRNYXBTWIKNXMWFSITAJWMJQRNSLFSDIFD**

**\* Tìm kiếm plaintext:**

Ta nhận thấy, các ký tự xuất hiện nhiều nhất trong dãy trên là:

'J', 'X', 'S', 'I', 'F', 'W', 'N'

Sử dụng bảng tần suất xuất hiện của chữ cái trong văn bản tiếng Anh, ta thấy ký tự có tần suất xuất hiện lớn nhất là ‘E’ (12.31)



Lần lượt thay ‘E’ tương ứng với các ký tự phía trên để tìm kiếm plaintext:

+ Với ‘E’ = ‘J’ (k = 5), ta có plaintext:

*FISHINGFRESHWATERBENDSADMITSWORDFISHRANDOVERHELMINGANYDAY*

\* Điểm yếu của giải thuật Caesar:

+ Giải thuật Caesar không làm thay đổi phân phối tần suất xuất hiện của từ, nên dễ bị tấn công bằng việc phân tích tần suất xuất hiện của từ.

+ Giải thuật Caesar chỉ cần thay thế lần lượt giá trị k từ 1 đến 25 để tìm ra plaintext, nên dễ bị tấn công bằng brute force

**Câu 2:**

Chương trình brute-force:

import string  
  
  
def decrypt(cipher\_text, key):  
 plain\_text = ""  
  
 for c in cipher\_text:  
 if c in string.ascii\_letters:  
 temp = ord(c) + key  
 if temp > ord('z'):  
 temp = temp - 26  
  
 plain\_text = plain\_text + chr(temp)  
 else:  
 plain\_text = plain\_text  
  
 return plain\_text  
  
  
CIPHER\_TEXT = "asvphgyt"  
  
for key in range(1, 26):  
 print(f"Key = {key} ====> ", decrypt(CIPHER\_TEXT, key))

Kết quả:

Key = 1 ====> btwqihzu

Key = 2 ====> cuxrjiav

Key = 3 ====> dvyskjbw

Key = 4 ====> ewztlkcx

Key = 5 ====> fxaumldy

Key = 6 ====> gybvnmez

Key = 7 ====> hzcwonfa

Key = 8 ====> iadxpogb

Key = 9 ====> jbeyqphc

Key = 10 ====> kcfzrqid

Key = 11 ====> ldgasrje

Key = 12 ====> mehbtskf

Key = 13 ====> nficutlg

Key = 14 ====> ogjdvumh

Key = 15 ====> phkewvni

Key = 16 ====> qilfxwoj

Key = 17 ====> rjmgyxpk

Key = 18 ====> sknhzyql

Key = 19 ====> tloiazrm

Key = 20 ====> umpjbasn

Key = 21 ====> vnqkcbto

**Key = 22 ====> worldcup**

Key = 23 ====> xpsmedvq

Key = 24 ====> yqtnfewr

Key = 25 ====> zruogfxs

Vậy với k = 22, ta có plaintext là “worldcup”

**Câu 3:**

Công thức mã hóa:

C = (ap + b) mod 26

Không mất tính tổng quát, giả sử a, b < 26

Kí tự B (2) nhiều nhất => Thay E (5) -> B(2)

Kí tự U (20) nhiều thứ hai => Thay T(20) -> U(21)

Từ (1) và (2)

=>

hay

Ta có:

Với k = 1 => a = 3,

k = 16 => a = 29

* Chọn a = 3

Với =>

=>

* Chọn b = 13

Công thức hệ mã:

C = (3p + 13) mod 26

**Câu 4:**

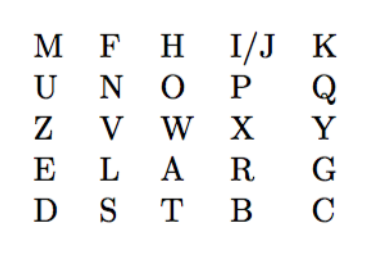
Hai vấn đề của mật mã One-time Pad:

+ Key phải có độ dài bằng message

+ Key phải được tạo ngẫu nhiên và chỉ dung được 1 lần

**Câu 5:**

Ma trận khóa:



Viết lại thành cặp 2 kí tự

Plaintext:

Must see you over Cadogan West. Coming at once

Viết lại thành

Mu

St

Se

Ey

Ou

Ov

Er

Ca

Do

Ga

nW

es

tC

om

in

ga

to

nc

ez (Do số lượng từ là lẻ nên thêm chữ “Z” vào từ cuối cùng)

Mã hóa

Mu -> UZ

St -> TB

Se -> DL

Ey -> GZ

Ou -> PN

Ov -> NW

Er -> LG

Ca -> TG

Do -> TU

Ga -> ER

nW -> OV

es -> LD

tC -> BD

om -> UH

in -> PF

ga -> ER

to -> HW

nc -> QS

ez -> DE

Ciphertext:

UZ TB DL GZ PN NW LG TG TU ER OV LD BD UH PF ER HW Qs DE

**Câu 6:**

Plaintext: spyarrivesonthursday

Key 1: BAO

|  |  |  |
| --- | --- | --- |
| 2 | 1 | 3 |
| **B** | **A** | **O** |
| S | p | y |
| A | r | r |
| I | v | e |
| S | o | n |
| T | h | u |
| R | s | d |
| A | y |  |

* prvohsy saistra yrenud
* prvohsysaistrayrenud

Key 2: KHANH

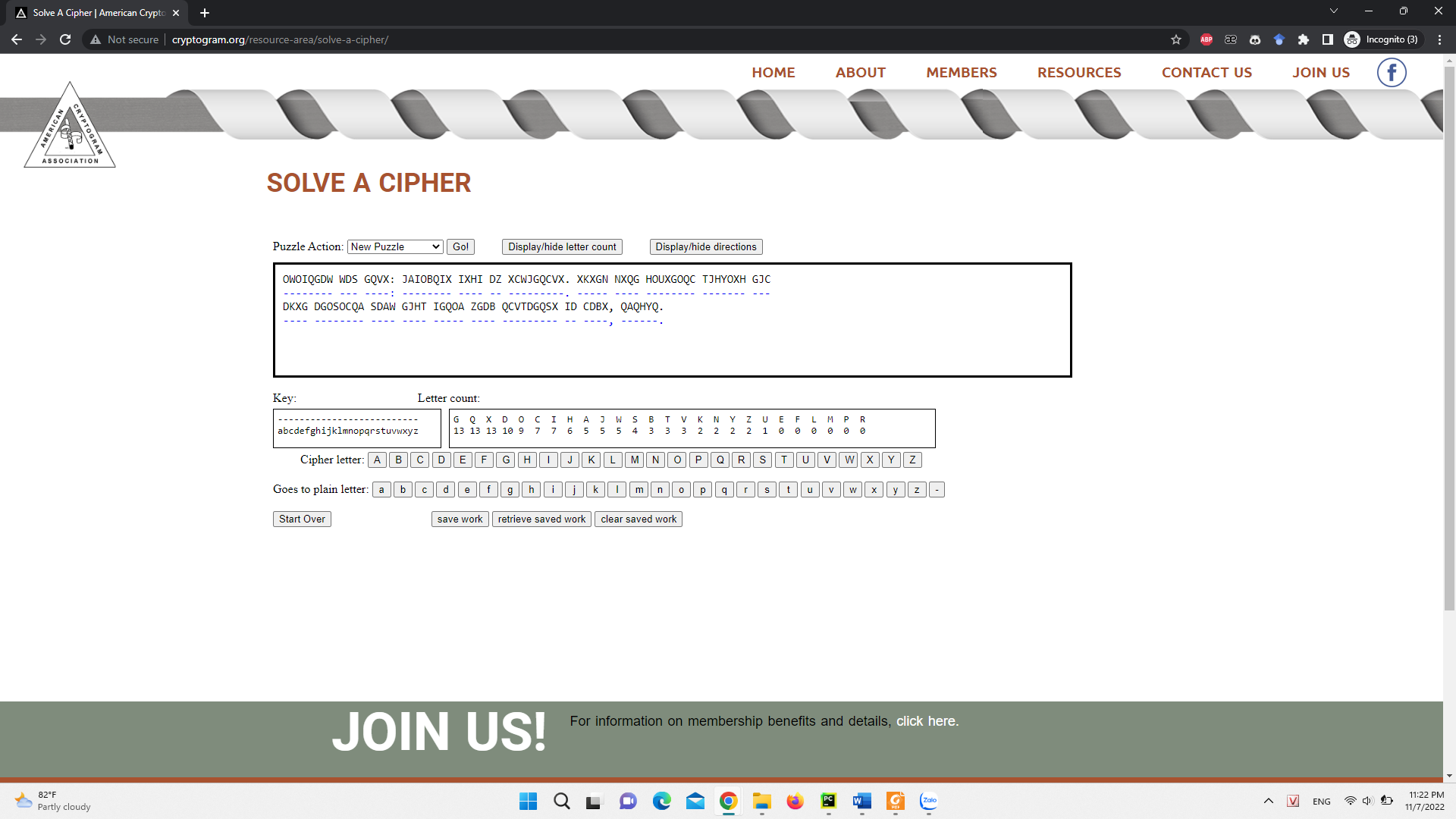
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 2 | 1 | 5 | 3 |
| **K** | **H** | **A** | **N** | **H** |
| p | r | v | O | h |
| s | y | s | A | i |
| s | t | r | A | y |
| r | e | n | U | d |

* vsrna ryte hiyd pssr oaau
* vsrnarytehiydpssroaau

**Câu 7:**

Cipher text:

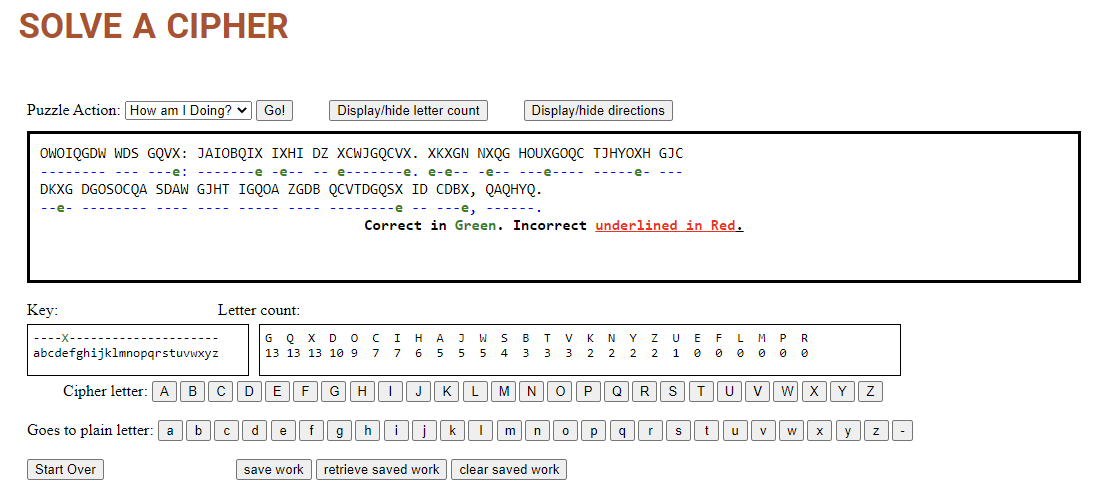
OWOIQGDW WDS GQVX: JAIOBQIX IXHI DZ XCWJGQCVX. XKXGN NXQG HOUXGOQC TJHYOXH GJC  
-------- --- ----: -------- ---- -- ---------. ----- ---- -------- ------- ---  
DKXG DGOSOCQA SDAW GJHT IGQOA ZGDB QCVTDGQSX ID CDBX, QAQHYQ.  
---- -------- ---- ---- ----- ---- --------- -- ----, ------.

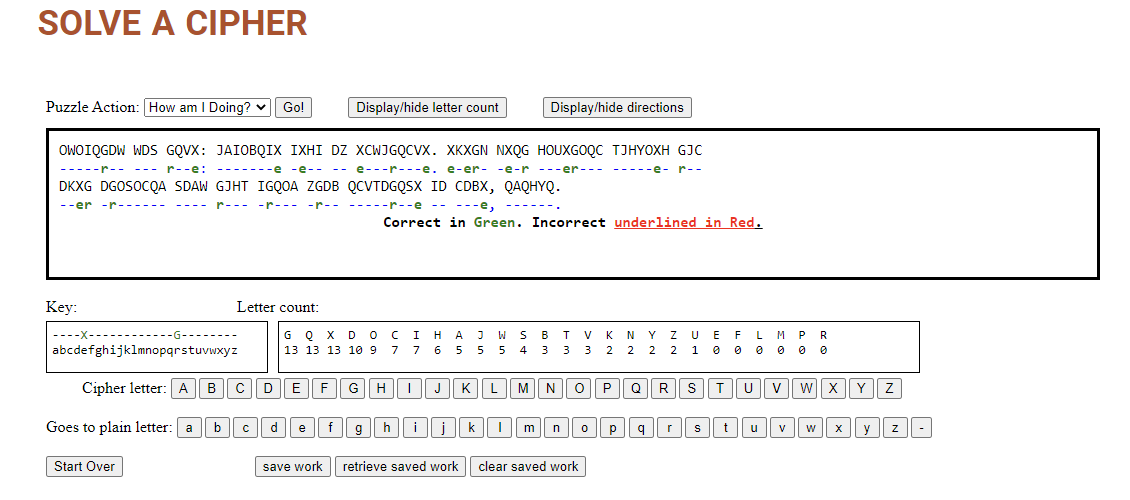


Thứ tự về độ phổ biến tần suất của các từ (giảm dần): E, T, A, O, N, I …

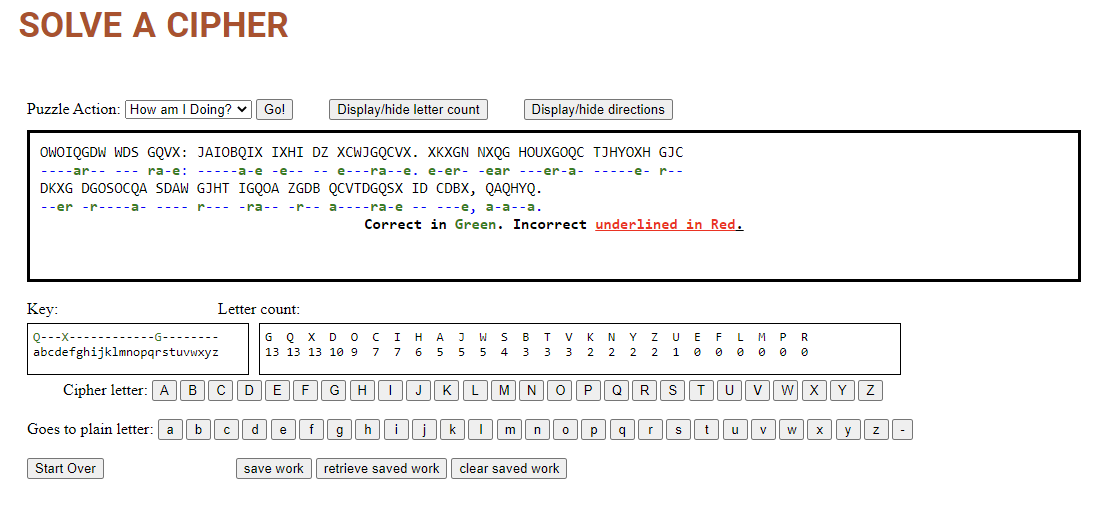
Phân tích tần suất xuất hiện, ta thấy G, Q, X xuất hiện nhiều nhất

Thử lần lượt, kiểm tra thấy X tương ứng với E

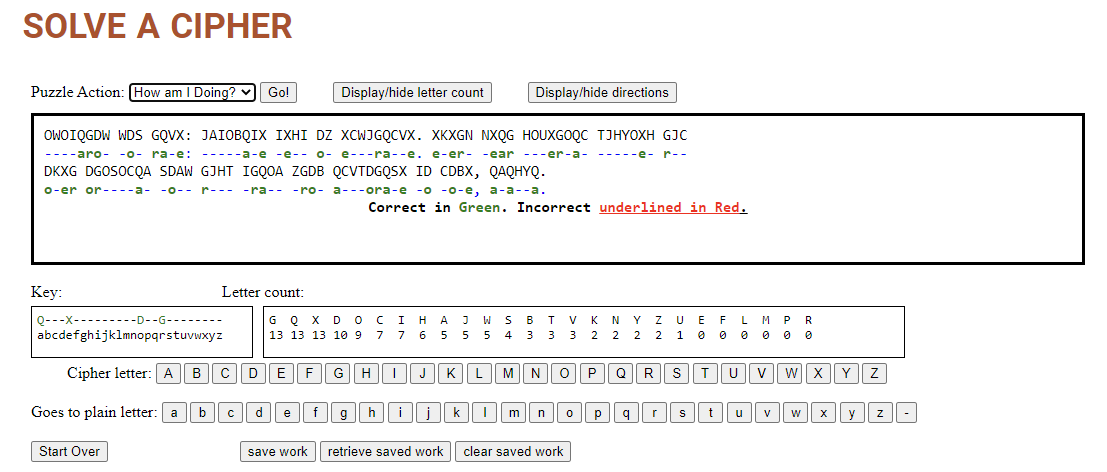


Theo tần suất xuất hiện, tiếp tục thử thay G bằng T, A, O, R thì thấy G tương ứng với R 

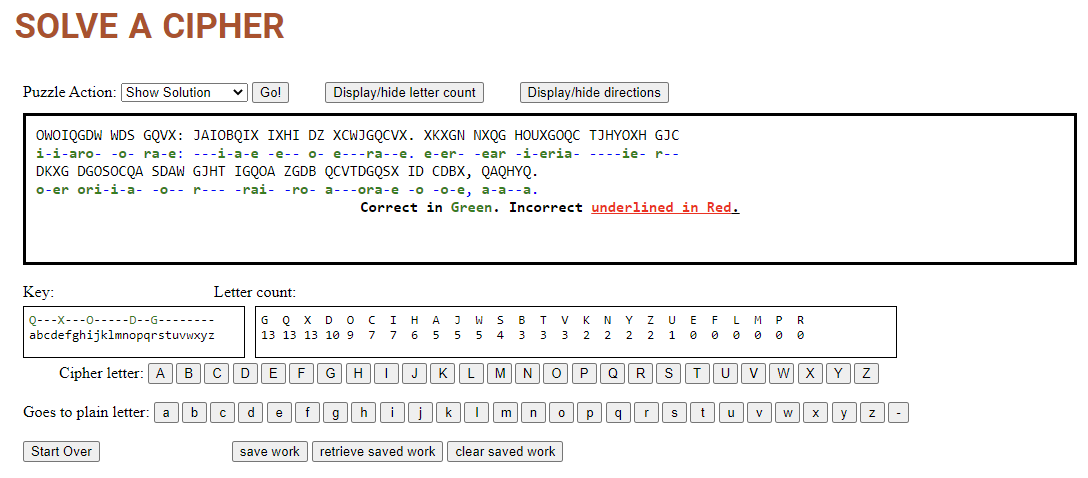
Tiếp tục thử thay Q bằng T, A, O thì thấy Q tương ứng với A



Thử với D, ta thấy D tương ứng với O

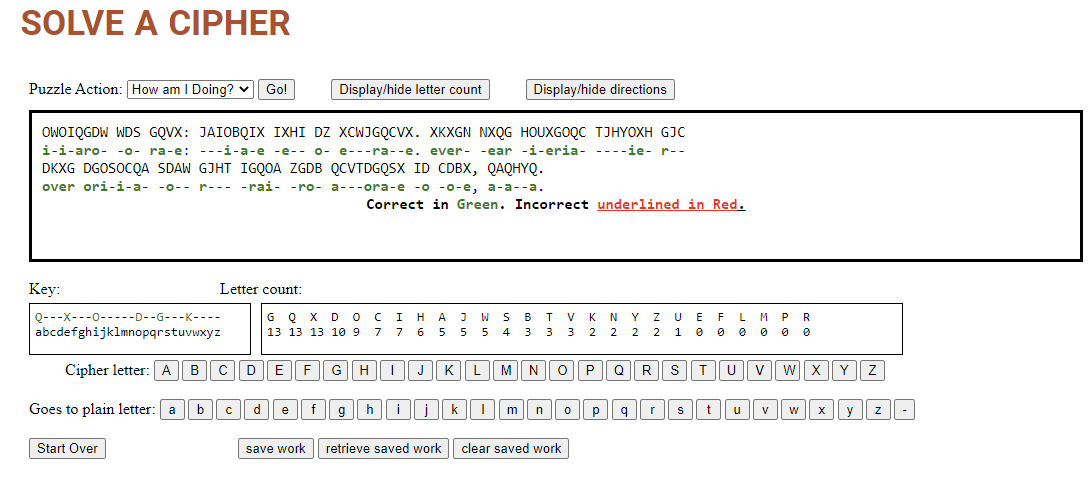


Tiếp tục thử với O, ta thấy O tương ứng với I



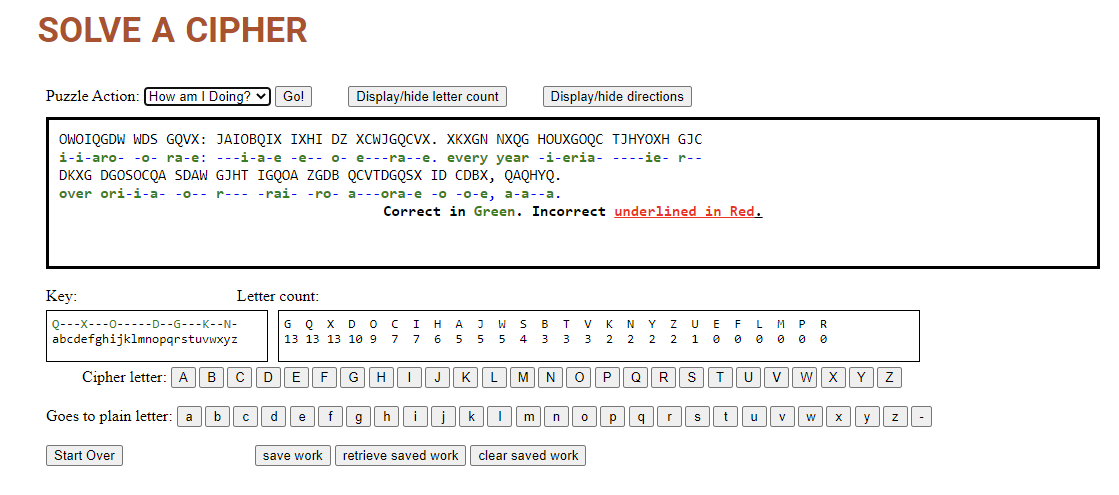
Ta tiếp tục đoán trên một số cụm từ như “o-er” => “over”

Thử K tương ứng V ta có kết quả chính xác:



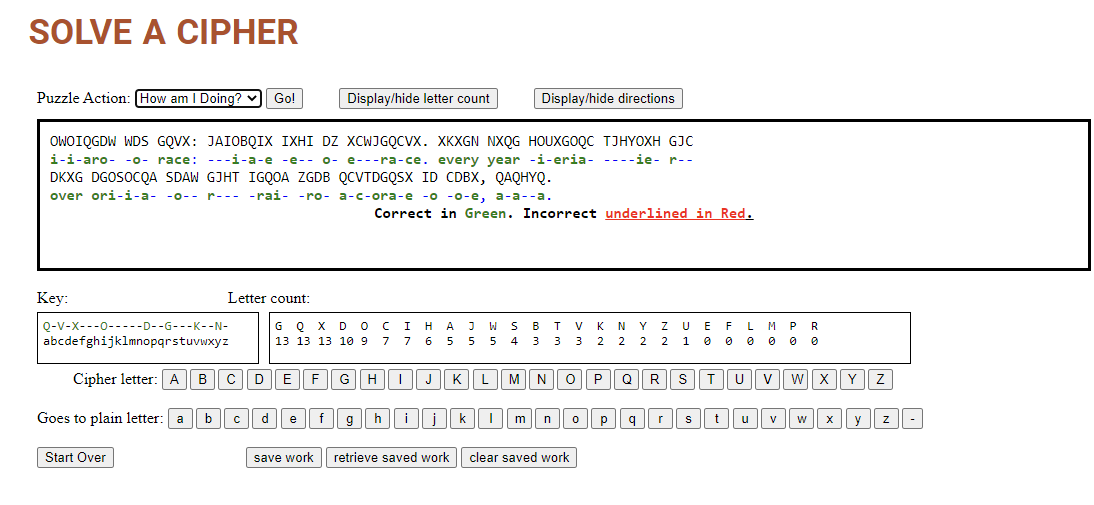
Ta đoán cụm “ever- -ear” => “every year”

Thử N tương ứng Y ta có kết quả chính xác

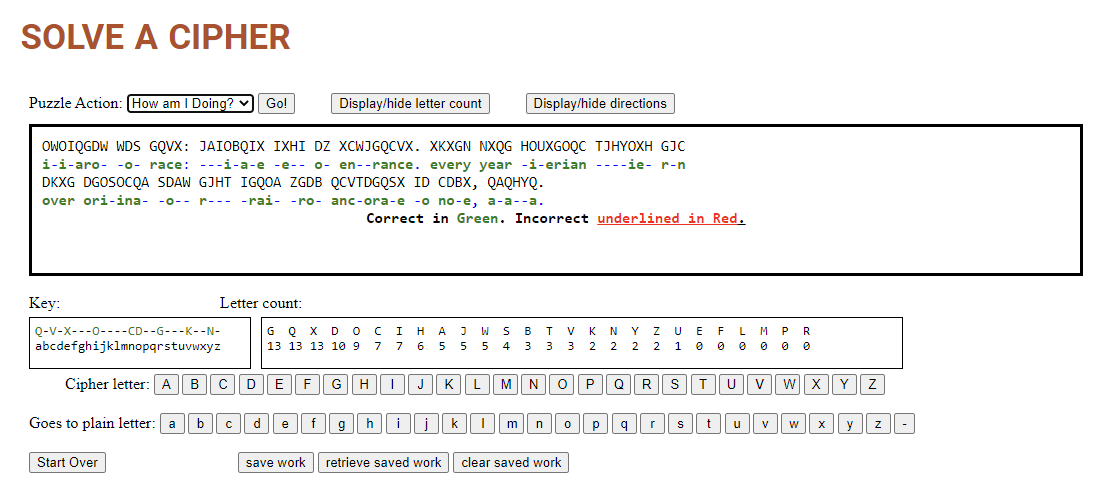


Ta đoán cụm “ra-e” => “race”

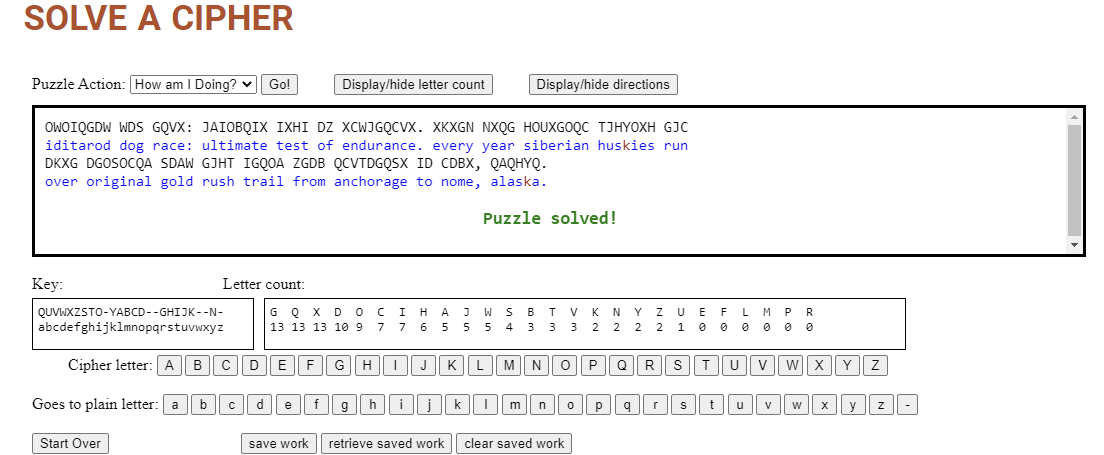
Thử V tương ứng C ta có kết quả chính xác



Tiếp tục thử tần suất với C, ta thấy C tương ứng với N



Tiếp tục thử và sai, giải tương tự, ta giải được đáp án:



Plaintext: iditarod dog race: ultimate test of endurance. every year siberian huskies run over original gold rush trail from anchorage to nome, alaska.

Phần 2. Chuẩn mã hoá dữ liệu DES

**Câu 1:**

|  |  |
| --- | --- |
| Mã hóa khối | Mã hóa dòng |
| Chuyển từ plaintext thành ciphertext theo từng block của plaintext | Chuyển từ plaintext thành ciphertext theo từng byte của plaintext |
| Đơn giản hơn | Phức tạp hơn |
| Sử dụng cả confusion + diffusion | Chỉ sử dụng confusion |
| Chậm hơn | Nhanh hơn |

Tham khảo từ: Difference between Block Cipher and Stream Cipher

[*https://www.geeksforgeeks.org/difference-between-block-cipher-and-stream-cipher/*](https://www.geeksforgeeks.org/difference-between-block-cipher-and-stream-cipher/)

**Câu 2:**

**\* Message (hex): 0 1 2 3 4 5 6 7 8 9 A B C D E F**

M (64 bit) = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

**\* Key (hex): 0 1 2 3 4 5 6 7 8 9 A B 1 3 6 3**

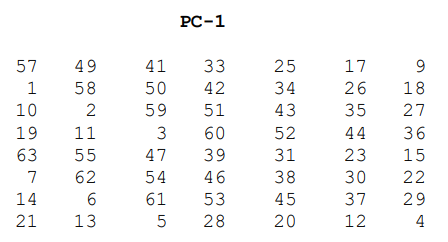
K (64 bit) = 00000001 00100011 01000101 01100111 10001001 10101011 00010011 01100011

**a. Tính khoá con K1:**

K = 00000001(8) 00100011(16) 01000101(24) 01100111(32) 10001001(40) 10101011(48) 00010011(56) 01100011(64)

(chú thích: những chữ số được highlight vàng là để chú thích về vị trí)

Thực hiện 56-bit permutation với PC-1:



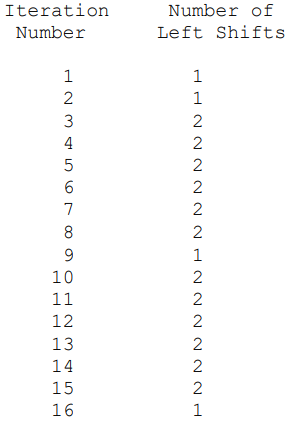
K+ = 0011000 0100011 0010101 0100100 1110101 0000011 0000110 0000000

Chia K+ thành nửa trái (28-bit) và nửa phải (28-bit)

C0 = 0011000 0100011 0010101 0100100

D0 = 1110101 0000011 0000110 0000000

Thực hiện shift trái 1 số (round 1)

****

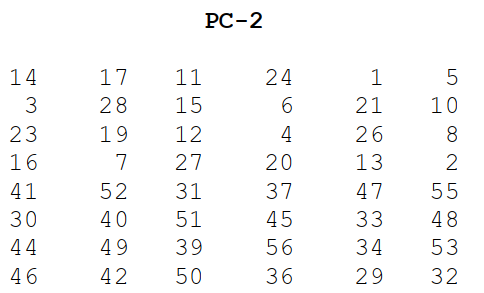
C1 = 0011000 0100011 0010101 0100100

D1 = 1101010 0000110 0001100 0000001

Kết hợp lại:

C1D1 = 0011000(7) 0100011(14) 0010101(21) 0100100(28) 1101010(35) 0000110(42) 0001100(49) 0000001 (56)

Thực hiện 56-bit permutation với PC-2:

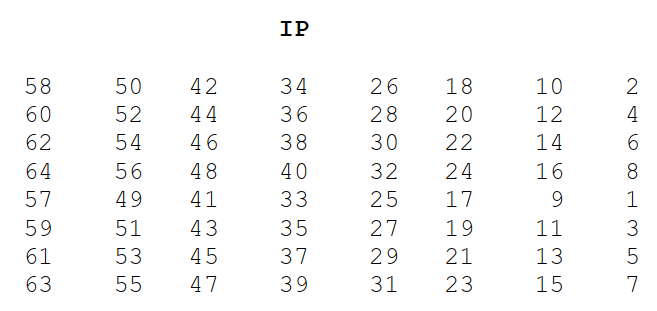


-> K1 = 110000 100010 110110 000010 100010 110000 000110 100011

**b.**

M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

Thực hiện initial permutation với M:



IP **=** 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000 1010 1010 1111 0000 1010 1010

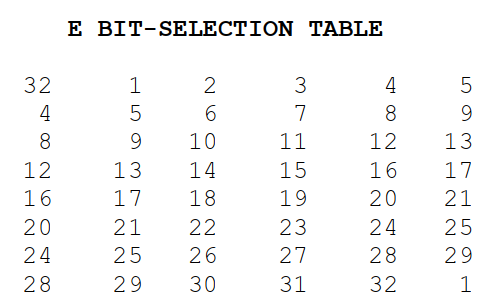
Chia IP thành nửa trái L0 (32 bits) và nửa phải R0 (32 bits)

L0 = 1100 1100 0000 0000 1100 1100 1111 1111  
R0 = 1111 0000 1010 1010 1111 0000 1010 1010

**c.**

Tính kết quả mở rộng R0

R0 = 1111 0000 1010 1010 1111 0000 1010 1010

  
E(R0) = 011110 100001 010101 010101 011110 100001 010101 010101

**d.**

E(R0) = 011110 100001 010101 010101 011110 100001 010101 010101

K1 = 110000 100010 110110 000010 100010 110000 000110 100011

A = E[R0] ⊕ K1

= 101110 000011 100011 010111 111100 010001 010011 110110

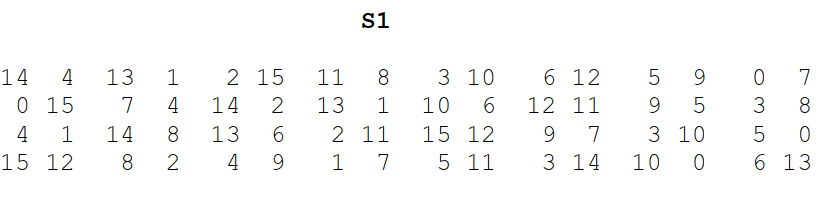
e.

A = 101110 000011 100011 010111 111100 010001 010011 110110

= B1 B2 B3 B4 B5 B6  B7 B8

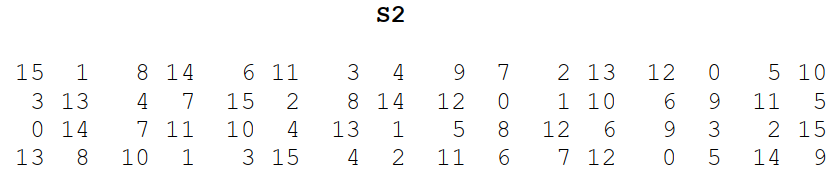
Tính toán S1(B1), S2(B2), S3(B3), S4(B4), S5(B5), S6(B6), S7(B7), S8(B8):

Với Si(Bi) là output thứ i của S-box



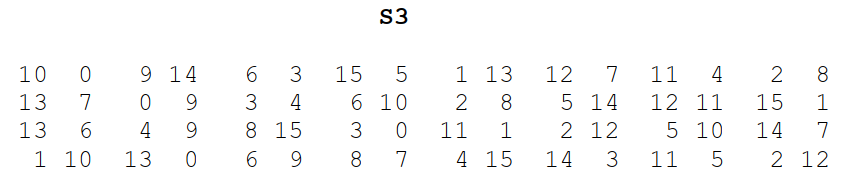
B1 = 101110 (hàng 10 = 2, cột 0111 = 7)

=> S1(101110) = 11 = 1011



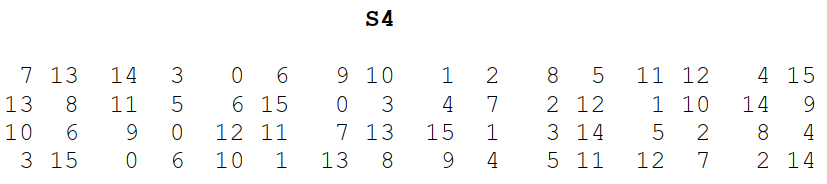
B2 = 000011 (hàng 01 = 1, cột 0001 = 1)

=> S2(000011) = 13 = 1101



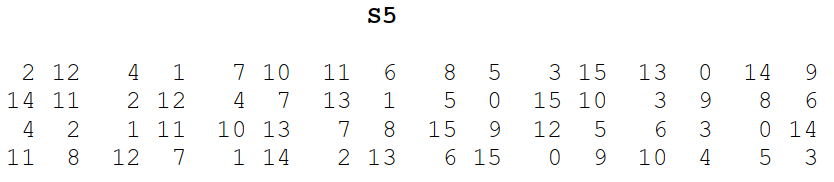
B3 = 100011 (hàng 11 = 3, cột 0001= 7)

=> S3(100011) = 7 = 0111



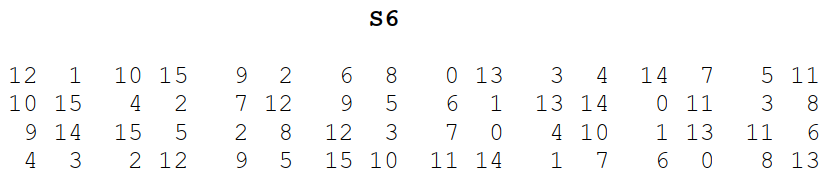
B4 = 010111 (hàng 01 = 1, cột 1011= 11)

=> S4(010111) = 12 = 1100



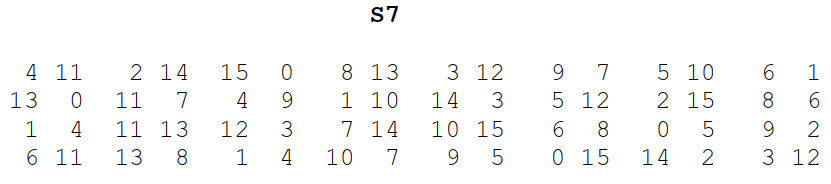
B5 = 111100 (hàng 10 = 2, cột 1110 = 14)

=> S5(111100) = 0 = 0000



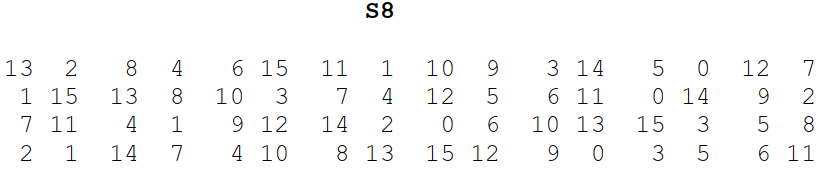
B6 = 010001 (hàng 01 = 1, cột 1000 = 8)

=> S6(010001) = 6 = 0110



B7 = 010011 (hàng 01 = 1, cột 1001= 9)

=> S7(010011) = 3 = 0011



B8 = 110110 (hàng 10 = 2, cột 1011= 11)

=> S8(110110) = 13 = 1101

**f.**

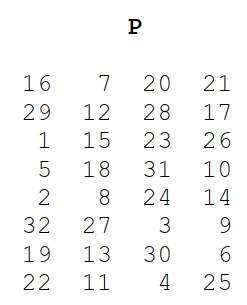
B

= S1(B1)S2(B2)S3(B3)S4(B4)S5(B5)S6(B6)S7(B7)S8(B8)

= 1011 1101 0111 1100 0000 0110 0011 1101

**g.**

Hoán vị B = 1011(4) 1101(8) 0111(12) 1100(16) 0000(20) 0110(24) 0011(28) 1101(32)



f = P(B) = 0000 1110 1010 1001 0101 1110 0111 1110

**h.**

P(B) = 0000 1110 1010 1001 0101 1110 0111 1110

L0 = 1100 1100 0000 0000 1100 1100 1111 1111

R1 = P (B) ⊕ L0

= 1100 0010 1010 1001 1001 0010 1000 0001

**i.**

L1 = R0 = 1111 0000 1010 1010 1111 0000 1010 1010

R1 = 1100 0010 1010 1001 1001 0010 1000 0001

Tính ciphertext cho round 1:

C1 = L1R1

= 1111 0000 1010 1010 1111 0000 1010 1010 1100 0010 1010 1001 1001 0010 1000 0001

= F0AAF0AAC2A99281