```
%       Post_Quantum_Cryptography_Project_1 Data Encryption Standard
%
% DES main program, Data Encryption Standard
%
% clear all;
clc;
%
% input message and key
%
plaintext = '0123456789abcdef';
key = '133457799bbcdff1';
%
% Key Schedule
%
K = KS(key);
%
% Encryption
%
ciphertext = DES_E(plaintext, K);
%
% Decryption
%
R_plaintext = DES_D(ciphertext, K);
%
% print data
%
fprintf('\n Original plaintext is: %s\n', plaintext);
fprintf('\n Encryption key is: %s \n', key);
fprintf('\n Encrypted ciphertext is: %s \n', ciphertext);
fprintf('\n Recovered plaintext is %s \n', R_plaintext);
%
Original plaintext is: 0123456789abcdef
Encryption key is: 133457799bbcdff1
Encrypted ciphertext is: 85E813540F0AB405
Recovered plaintext is 0123456789ABCDEF
>>
```

```
%
% DES Encryption function
%
function ciphertext = DES_E(plaintext, K)
%
% step: Initial Permutation(IP)
%
% round 0
[L0, R0] = IP(plaintext);
L(1, :) = L0;
R(1, :) = R0;
%
% step: 16 rounds
%
for ir = 1 : 16
    [Li, Ri] = Round(L(ir, :), R(ir, :), K(ir, :));
    L(ir + 1, :) = Li;
    R(ir + 1, :) = Ri;
end
%
% step: L <-> R (L and R Swap)
%
L16 = R(17, :);
R16 = L(17, :);
%
% step: Inverse Initial Permutation(IIP)
%
ciphertext = IIP(L16, R16);
return

%
% DES Decryption function
%
function R_plaintext = DES_D(ciphertext, K)
%
% step: Initial Permutation(IP)
%
% round 0
```

```
[L0, R0] = IP(ciphertext);
L(1, :) = L0;
R(1, :) = R0;
%
% step: 16 rounds
%
for ir = 1 : 16
    [Li, Ri] = Round(L(ir, :), R(ir, :), K(17 - ir, :));
    L(ir + 1, :) = Li;
    R(ir + 1, :) = Ri;
end
%
% step: L <-> R (L and R Swap)
%
L16 = R(17, :);
R16 = L(17, :);
%
% step: Inverse Initial Permutation(IIP)
%
R_plaintext = IIP(L16, R16);
Return

%
% Initial Permutation(IP) function
%
function [L, R] = IP(plaintext)
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 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 39 31 23 15  7];
L = plaintext(1:8); % L in hex char
R = plaintext(9:16);
Lb = uint32(hex2dec(L)); % Lb in 32 bits digit
Rb = uint32(hex2dec(R));
```

```
Lb_temp = Lb;
Rb_temp = Rb;
for i = 1 : 32
    if IP(i) <= 32
        Lb = bitset(Lb, 33 - i, bitget(Lb_temp, 33 - IP(i)));
    else
        Lb = bitset(Lb, 33 - i, bitget(Rb_temp, 65 - IP(i)));
    end
    if IP(i + 32) <= 32
        Rb = bitset(Rb, 33 - i, bitget(Lb_temp, 33 - IP(i + 32)));
    else
        Rb = bitset(Rb, 33 - i, bitget(Rb_temp, 65 - IP(i + 32)));
    end
end
L = dec2hex(Lb, 8);
R = dec2hex(Rb, 8);
return

%
% Inverse Initial Permutation(IIP) function
%
function out = IIP(L, R)
IIP = [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  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   41  9   49  17  57  25];
Lb = uint32(hex2dec(L)); % Lb in 32 bits digit
Rb = uint32(hex2dec(R));
Lb_temp = Lb;
Rb_temp = Rb;
for i = 1 : 32
    if IIP(i) <= 32
        Lb = bitset(Lb, 33 - i, bitget(Lb_temp, 33 - IIP(i)));
    else
```

```
        Lb = bitset(Lb, 33 - i, bitget(Rb_temp, 65 - IIP(i)));
    end
    if IIP(i + 32) <= 32
        Rb = bitset(Rb, 33 - i, bitget(Lb_temp, 33 - IIP(i + 32)));
    else
        Rb = bitset(Rb, 33 - i, bitget(Rb_temp, 65 - IIP(i + 32)));
    end
end
L = dec2hex(Lb, 8);
R = dec2hex(Rb, 8);
out = strcat(L, R); % str L + str R
return

%
% Key Schedule function for DES
%
function out = KS(key)
%
% permutation choice one
%
PC1L = [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];
PC1R = [63  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];
%
% shift number
%
Shift = [1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1];
%
% permutation choice two
%
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  30  40 ...
       51   45  33  48  44  49  39  56 ...
       34   53  46  42  50  36  29  32];
%
% key from input key string
%
K_L = key(1:8);
K_R = key(9:16);
K_Lb = uint32(hex2dec(K_L));
K_Rb = uint32(hex2dec(K_R));
%
% key schedule
%
C = uint32(0);
D = uint32(0);
for i = 1 : 28
    if PC1L(i) <= 32
        C = bitset(C, 33-i, bitget(K_Lb, 33-PC1L(i)));
    else
        C = bitset(C, 33-i, bitget(K_Rb, 65-PC1L(i)));
    end
    if PC1R(i) <= 32
        D = bitset(D, 33-i, bitget(K_Lb, 33-PC1R(i)));
    else
        D = bitset(D, 33-i, bitget(K_Rb, 65-PC1R(i)));
    end
end
%
K = uint64(zeros(16, 1));
for ir = 1 : 16
    if Shift(ir) == 1
        C = bitset(C, 4, bitget(C, 32));
        C = bitshift(C, 1);
        D = bitset(D, 4, bitget(D, 32));
        D = bitshift(D, 1);
    else
        C = bitset(C, 4, bitget(C, 32));
        C = bitshift(C, 1);
```

```
        D = bitset(D, 4, bitget(D, 32));
        D = bitshift(D, 1);
        C = bitset(C, 4, bitget(C, 32));
        C = bitshift(C, 1);
        D = bitset(D, 4, bitget(D, 32));
        D = bitshift(D, 1);
    end
    for i = 1 : 48
        if PC2(i) <= 28
            K(ir) = bitset(K(ir), 49-i, bitget(C, 33-PC2(i)));
        else
            K(ir) = bitset(K(ir), 49-i, bitget(D, 61-PC2(i)));
        end
    end
end
out = dec2hex(double(K), 16);
return

%
% Round function for DES
%
function [Lo, Ro] = Round(L, R, Ki)
%
% Expansion Matrix for Expansion
%
E = [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  22  23  24  25 ...
     24  25  26  27  28  29 ...
     28  29  30  31  32  1];
%
% S-box for Substitution
%
S = zeros(4, 16, 8);
S(:,:,1) = [14  4   13  1   2   15  11  8   3   10  6   12  5   9   0   7;
```

```
            0  15   7   4   14  2   13  1   10  6   12  11  9   5   3   8;
            4   1   14  8   13  6   2   11  15  12  9   7   3   10  5   0;
            15  12  8   2   4   9   1   7   5   11  3   14  10  0   6   13];
S(:,:,2) = [15  1   8   14  6   11  3   4   9   7   2   13  12  0   5   10;
            3   13  4   7   15  2   8   14  12  0   1   10  6   9   11  5;
            0   14  7   11  10  4   13  1   5   8   12  6   9   3   2   15;
            13  8   10  1   3   15  4   2   11  6   7   12  0   5   14  9];
S(:,:,3) = [10  0   9   14  6   3   15  5   1   13  12  7   11  4   2   8;
            13  7   0   9   3   4   6   10  2   8   5   14  12  11  15  1;
            13  6   4   9   8   15  3   0   11  1   2   12  5   10  14  7;
            1   10  13  0   6   9   8   7   4   15  14  3   11  5   2   12];
S(:,:,4) = [7   13  14  3   0   6   9   10  1   2   8   5   11  12  4   15;
            13  8   11  5   6   15  0   3   4   7   2   12  1   10  14  9;
            10  6   9   0   12  11  7   13  15  1   3   14  5   2   8   4;
            3   15  0   6   10  1   13  8   9   4   5   11  12  7   2   14];
S(:,:,5) = [2   12  4   1   7   10  11  6   8   5   3   15  13  0   14  9;
            14  11  2   12  4   7   13  1   5   0   15  10  3   9   8   6;
            4   2   1   11  10  13  7   8   15  9   12  5   6   3   0   14;
            11  8   12  7   1   14  2   13  6   15  0   9   10  4   5   3];
S(:,:,6) = [12  1   10  15  9   2   6   8   0   13  3   4   14  7   5   11;
            10  15  4   2   7   12  9   5   6   1   13  14  0   11  3   8;
            9   14  15  5   2   8   12  3   7   0   4   10  1   13  11  6;
            4   3   2   12  9   5   15  10  11  14  1   7   6   0   8   13];
S(:,:,7) = [4   11  2   14  15  0   8   13  3   12  9   7   5   10  6   1;
            13  0   11  7   4   9   1   10  14  3   5   12  2   15  8   6;
            1   4   11  13  12  3   7   14  10  15  6   8   0   5   9   2;
            6   11  13  8   1   4   10  7   9   5   0   15  14  2   3   12];
S(:,:,8) = [13  2   8   4   6   15  11  1   10  9   3   14  5   0   12  7;
            1   15  13  8   10  3   7   4   12  5   6   11  0   14  9   2;
            7   11  4   1   9   12  14  2   0   6   10  13  15  3   5   8;
            2   1   14  7   4   10  8   13  15  12  9   0   3   5   6   11];
%
% P Matrix for Permutation
%
PF = [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  25];
```

```
%
%
L_temp = L;
L = R;
%
% first, expansion
%
Rb = uint32(hex2dec(R));
% default number is in double(64 bits), but we're using the first 48 bits only
Rb_E = 0; % E for expansion
for i = 1 : 48
    Rb_E = bitset(Rb_E, 49 - i, bitget(Rb, 33 - E(i)));
end
%
% XOR with key Ki, 64 bit key
%
Rb_E_K = bitxor(Rb_E, hex2dec(Ki));
%
% S-Box
%
S_8 = uint32(0);
for i = 1 : 8
    % row index
    x = bitget(Rb_E_K, (9-i)*6) * 2 + bitget(Rb_E_K, (9-i)*6 - 5) + 1;
    % col index
    y = bitget(Rb_E_K, (9-i)*6 - 1) * 2^3 + bitget(Rb_E_K, (9-i)*6 - 2) * 2^2
  + ... bitget(Rb_E_K, (9-i)*6 - 3) * 2   + bitget(Rb_E_K, (9-i)*6 - 4) + 1;
    Si = S(x, y, i);
    S_8 = S_8 + Si * 16^(8-i);
end
%
% 32-bit permutation function
%
temp = S_8;
for i = 1 : 32
    S_8 = bitset(S_8, 33 - i, bitget(temp, 33 - PF(i)));
end
%
```

```
% XOR with L
%
R = dec2hex(bitxor(uint32(hex2dec(L_temp)), S_8), 8);
% R = dec2hex(bitxor(S_8, uint32(hex2dec(L_temp))), 8);
%
%
Lo = L;
Ro = R;
return

%
% TDEA main program, Triple Data Encryption Algorithm
%
% clear all;
clc;
%
% input message and key
%
% plaintext = '0123456789abcdef';
% plain = '0123456789abcdeffecdab1645237980fecdab16452379800123456789abcdef33';
%
% plain_text = '-n°µª°¨Æ±¡Á`§ä±o¥X®É¶¡©M¾÷·|¡A¤£-n°µª°¨Æ±¡Á`§ä ª°¥XÂÇ¤f¡C--
±i·R¬Â';
text_mode = 'Chinese';
%
word = actxserver('Word.Application');
wdoc = word.Documents.Open('D:\Beginner MATLAB
Projects\Post_Quantum_Cryptography_MATLAB\DES_verify\§E¥ú¤¤_ªü¨½¤sÆg.docx');
plain_text = wdoc.Content.Text;
Quit(word);
delete(word);
%
% plain_text = 'No man is an island, Entire of itself, Every man is a piece of
the continent';
% text_mode = 'English';
%
% Input key has to be 0-9, a-f, or A-F.
key1 = '133457799bbcdff1';
```

```
key2 = 'aabcdbcdeffe6497';
key3 = '974acfe58d1b32f6';
%
% Key Schedule
%
K1 = KS(key1);
K2 = KS(key2);
K3 = KS(key3);
%
% check point
%
if strcmp(text_mode, 'English') == 1
    plain_text_len = length(plain_text);
    plain = char();
    for i = 1 : plain_text_len
        % turn the char string into a double,
        % then turn into a 2 bits hexdecimal
        pp = dec2hex(double(plain_text(i)), 2);
        plain = strcat(plain, pp);
    end
elseif strcmp(text_mode, 'Chinese') == 1
    plain_text_len = length(plain_text);
    plain = char();
    for i = 1 : plain_text_len
        % turn the char string into a double,
        % then turn into a 2 bits hexdecimal
        pp = dec2hex(double(plain_text(i)), 4);
        plain = strcat(plain, pp);
    end
end
%
plain_len = length(plain);
q = floor(plain_len / 16); % integer division
r = mod(plain_len, 16);
if r ~= 0
    q = q + 1;
    % pending
    plain = strcat(plain, '8');
```

```
    for i = 1 : 16-r-1
        plain = strcat(plain, '0');
    end
end
%
% Encryption
%
ciphertext = char();
for iq = 1 : q
    P = plain((iq-1)*16 + 1 : iq*16);
    C = DES_E(DES_D(DES_E(P, K1), K2), K3);
    ciphertext = strcat(ciphertext, C);
end
%
% Decryption
%
cipher_len = length(ciphertext);
q = cipher_len / 16;
R_plaintext = char();
for iq = 1 : q
    C = ciphertext((iq-1)*16 + 1 : iq*16);
    P = DES_D(DES_E(DES_D(C, K3), K2), K1);
    R_plaintext = strcat(R_plaintext, P);
end
%
if strcmp(text_mode, 'English') == 1
    R_plaintext_len = length(R_plaintext);
    % to preserve the spaces in the original char string, first
    R_plain_text = zeros(1, R_plaintext_len/2);
    for i = 1 : R_plaintext_len/2
        R_plain_text(i) = hex2dec(R_plaintext((i-1)*2 + 1 : i*2));
    end
    R_plain_text = char(R_plain_text);
elseif strcmp(text_mode, 'Chinese') == 1
    R_plaintext_len = length(R_plaintext);
    % to preserve the spaces in the original char string, first
    R_plain_text = zeros(1, R_plaintext_len/4);
    for i = 1 : R_plaintext_len/4
```

```
        R_plain_text(i) = hex2dec(R_plaintext((i-1)*4 + 1:i*4));
    end
    R_plain_text = char(R_plain_text);
end
%
fid = fopen('R_plaintext.doc', 'wt');
fprintf(fid, '%s', R_plain_text);
fclose(fid);
%
% print data
%
% fprintf('\n Original plaintext is: %s\n', plain);
fprintf('\n Original plaintext is: %s\n', plain_text);
fprintf('\n Encryption key1 is: %s \n', key1);
fprintf('\n Encryption key2 is: %s \n', key2);
fprintf('\n Encryption key3 is: %s \n', key3);
fprintf('\n Encrypted ciphertext is: %s \n', ciphertext);
% fprintf('\n Recovered plaintext is: %s \n', R_plaintext);
fprintf('\n Recovered plaintext is: %s \n', R_plain_text);
%
```

```
Original plaintext is: 余光中 阿里山讚

春季為何總如此年輕
山雀和蜜蜂究竟
對櫻花說了些什麼
秋季為何總如此清醒
銀杏和青楓究竟
對風霜說了些什麼
神木為何總如此沈靜
古老的回憶究竟
內心轉多少層年輪
高山為何總如此鎮定
斜坡和絕壁究竟
是怎樣的去脈來龍
這一切，只有太陽知道
這一切造化之功
連史前的造山運動
```

只有牠，如此年輕
每天把台灣喚醒
為阿里山加上金冠
一頂金冠尊貴而燦爛
用霞火煉丹而成
全世界共仰的壯觀

Encryption key1 is: 133457799bbcdff1
Encryption key2 is: aabcdbcdeffe6497
Encryption key3 is: 974acfe58d1b32f6

Encrypted ciphertext is:
BE8E135886C07263FC9B0EED984C561D65F48616805B9BF5FBAEB9B7A9405878A6EB173BBE552B3
03A53583D79EF83C81429B2F9F757C3D0B7792272368DC673D3E3F1E52AAE185B121BBF51CE67F4
F7D71D7D3A5FE69E5B97B0122EF1FC9E86A92BE208A8C1F19B58C815F74F2845742226684C45F7B
9EFB0D26FA23A11BFDFEEA7724A43830B88F16E389818E1677695568638DFF445AFDEAE07C7118B
DAFB56ED75C0A0EA7B35D860CEDF2A71ADCD057D376BF426A3C3048EAB89E580D9F697B77131EDB
8B3852039F7D6CC22141BBA012C240D23614FF6D9E9C68D15824FA57759FF798601DDA758808DDF
D04F4EB47F63FA236B3B99250155E0F0667903A25E846036CB4CF4207B176422A0C90AB5E779FBE
2F1DBB88460A51A27D530A73CF98F2D5B79B8198D50BA414CC6E4CA8EDD5DADFBE16C1116C92F93
961ABC92109983B48E9DEB48028378B61A664CB4EF59AEBBA882B05969E440D059365176EEA9625
BFED88834C2588CD9128A4D041AFE4BF7B623E1E976674DA5872AFEABFCEFBFC5E2E2223487BA60
F795D2171D

Recovered plaintext is: 余光中 阿里山讚

春季為何總如此年輕
山雀和蜜蜂究竟
對櫻花說了些什麼
秋季為何總如此清醒
銀杏和青楓究竟
對風霜說了些什麼
神木為何總如此沈靜
古老的回憶究竟
內心轉多少層年輪
高山為何總如此鎮定
斜坡和絕壁究竟

是怎樣的去脈來龍
這一切，只有太陽知道
這一切造化之功
連史前的造山運動
只有牠，如此年輕
每天把台灣喚醒
為阿里山加上金冠
一頂金冠尊貴而燦爛
用霞火煉丹而成
全世界共仰的壯觀
耀
>>