Network Security (CSE350)
Programming Assignment no. 2
(Project 1)

Programming language: C++
Platform: MS Windows

In this project we are implementing a AES (Advanced Encryption Standard) which uses a symmetric encryption technique in 128 bits of fixed-size blocks used to encrypt data
the encryption process uses 10 rounds of operations, where each round consist of four steps - SubBytes, ShiftRows, MixColumns, and AddRoundKey
Every byte in the block is changed in the SubBytes step by matching byte from a predetermined substitution table known as S-box. The block is moved one byte to the left in the second row, two bytes to the left in the third row, and three bytes to the left in the fourth row during the ShiftRows step. Every column in the block is multiplied in the MixColumns step.
to guarantee that the diffusion process has an impact on every byte in the block using a fixed matrix. Lastly, a key schedule is used to XOR the block with a subset of the key that is obtained from the main key in the AddRoundKey phase.
We defined the Original key as a 4x4 matrix of hexadecimal strings and constructed the Sbox which is used as the substitution tables in the encryption process, from the original key (which is of size 128 bits or 16 bytes) generating subkeys which are stored in the sub_key array during key expansion process to be used in each of the rounds(total of 10 rounds) of AES Algorithms, then for the AES encryption process by taking 4 plain texts as input and similarly AES Decryption using Inverse operations of the cipher text that is produced, both the plain text of the first encryption and decryption of the 9th cipher text matches and vice-versa

<u>Functionalities implemented in our system</u>:

- Construct_s_box: Constructs the s_box in the form of a C++ map.
- Construct_inverse_s_box: Constructs the inverse_s_box in the form of a C++ map.
- Copy_data: Copy data of one matrix into another.
- Hex_string_to_binary: Converts a string of 2 hexadecimal digits into the binary representation stored inside a string.
- Binary_to_hex: Converts a string of 8 size representing 8 bits into a corresponding string of 2 hexadecimal digits.
- Xor_operation: perform XOR operation on 4 bytes
  .

- GF_multiplication: perform Galois Field multiplication of 2 strings representing 2 hexadecimal digits each.
- Substitute_bytes: perform substitute bytes transformation of AES
- Inverse_sub_bytes: perform Inverse substitution byte transformation of AES
- Shift_rows: perform shift rows transformation of AES
- Inverse_shift_rows: perform inverse transformation of AES
- Mix_column: perform mix columns transformation of AES
- Inverse_mix_column: perform inverse mix columns transformation of AES
- Add_round_key: perform XOR operation with the respective sub-key of that round.
- G_function: perform shift_rows, then substitute byte, then xor with a special value to obtain a value.
- Generate_sub_key: function used to generate 10 subkeys for each round.
- Matrix_to_string: converts a matrix with every element being 2 hexadecimal digits stored as a string to a string of hexadecimals.
- String_to_matrix: converts a string of hexadecimals into a matrix with 2 hexadecimals each in every element.
- AES_Encryption: perform AES encryption for a given plaintext.
- AES_Decryption: perform AES decryption for a given ciphertext.
- Generate_plaintext: convert a string of characters(1 byte) into a string of corresponding hexadecimals(4 bit)
- Generate_originaltext: convert a string of hexadecimals into a string of characters(1 byte).

Sample Input:

```
PS C:\Users\vansh\Desktop\NSC-A2> cd "c:\Users\vansh\Desktop\NSC-A2\" ; if ($?) { g++ code.cpp -o code } ; if ($?) { .\code }
Key(128 bit) in hexadecimal: 54686174732O6D79204B756E67204675

First Pair:
576861747361707O2A596F7574756265, 3052D8148F3D687F773A1172E1C64BD6

Orignal text(in characters): Whatsapp*Youtube
Plaintext(in hexadeimals): 576861747361707O2A596F7574756265
Ciphertext(in hexadecimals): 3052D8148F3D687F773A1172E1C64BD6
Cihertext(in characters): 0R╪¶Å=hw:◄rß╞K╓
Decypted Text(in characters): Whatsapp*Youtube
Decrypted Text(in hexadecimals): 576861747361707O2A596F7574756265
First Encryption(in hexadecimals): E2C053F5DCE98E8F480841B7FC50ABE6
Ninth Decryption(in hexadecimals): E2C053F5DCE98E8F480841B7FC50ABE6
Ninth Encryption(in hexadecimals): 34F0D2B61E25D2DEB7F99B178EE96F4E
First Decryption(in hexadecimals): 34F0D2B61E25D2DEB7F99B178EE96F4E

Second Pair:
6D796D61726B73696E4E534331303078, 8BAD2050CC68F0191D0FF4583C527633

Orignal text(in characters): mymarksinNSC100x
Plaintext(in hexadecimals): 6D796D61726B73696E4E534331303078
Ciphertext(in hexadecimals): 8BAD2050CC68F0191D0FF4583C527633
Cihertext(in characters): ï¡ P╠h≡↓↔┌X<Rv3
Decrypted Text(in hexadecimals): 6D796D61726B73696E4E534331303078
Decypted Text(in characters): mymarksinNSC100x
First Encryption(in hexadecimals): 282B8711307BDC2485943996E84A5C97
Ninth Decryption(in hexadecimals): 282B8711307BDC2485943996E84A5C97
Ninth Encryption(in hexadecimals): 71346CEE2E1F97B23327832F880B0805
First Decryption(in hexadecimals): 71346CEE2E1F97B23327832F880B0805

Thrid Pair:
656B6B686F6B61646F40646976696E65, 5B73306A1E3CE98DCCEFA557FFAE77A7

Orignal text(in characters): ekkhokado@divine
Plaintext(in hexadecimals): 656B6B686F6B61646F40646976696E65
Ciphertext(in hexadecimals): 5B73306A1E3CE98DCCEFA557FFAE77A7
Cihertext(in characters): [s0j▲<0ì╠nÑW «wº
Decrypted Text(in hexadecimals): 656B6B686F6B61646F40646976696E65
Decypted Text(in characters): ekkhokado@divine
First Encryption(in hexadecimals): F17C9F3AC5AF187A5C60D50B039FD268
Ninth Decryption(in hexadecimals): F17C9F3AC5AF187A5C60D50B039FD268
Ninth Encryption(in hexadecimals): 8FAE09784DE9F27639E2559138594C70
First Decryption(in hexadecimals): 8FAE09784DE9F27639E2559138594C70
```

```
Fourth Pair:
3142686172617468416E6456616E7368, AC91965752A3D1AC86B016FF1D1E4427

Orignal text(in characters): 1BharathAndVansh
Plaintext(in hexadecimals): 3142686172617468416E6456616E7368
Ciphertext(in hexadecimals): AC91965752A3D1AC86B016FF1D1E4427
Cihertext(in characters): ¼æûWRú╤¼å▒▀ ↔▲D'
Decrypted Text(in hexadecimals): 3142686172617468416E6456616E7368
Decypted Text(in characters): 1BharathAndVansh
First Encryption(in hexadecimals): C0F897532BDE62858279F166AE11D745
Ninth Decryption(in hexadecimals): C0F897532BDE62858279F166AE11D745
Ninth Encryption(in hexadecimals): 4FFD3E751B55C0965E3846092AE7E3B8
First Decryption(in hexadecimals): 4FFD3E751B55C0965E3846092AE7E3B8
PS C:\Users\vansh\Desktop\NSC-A2>
```