Program: AES

```
#include <iostream>
    #include <bitset>
    using namespace std;
    typedef bitset<8> byte;
    typedef bitset<32> word;

    const int Nr = 10;  //AES-128 requires 10 rounds of encryption
    const int Nk = 4;   //Nk Represents the number of word s that are input keys

    byte S_Box[16][16] = {
      {0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30, 0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB,
0x76},
      {0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72,
0xC0},
      {0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31,
0x15},
      {0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2,
0x75},
      {0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F,
0x84},
      {0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58,
0xCF},
      {0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F,
0xA8},
      {0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3,
0xD2},
      {0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19,
0x73},
      {0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B,
0xDB},
      {0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4,
0x79},
      {0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE,
0x08},
      {0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B,
0x8A},
```

```
        {0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D,
0x9E},
        {0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28,
0xDF},
        {0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41, 0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB,
0x16}
    };

    //Round constant, used in key expansion. (AES-128 only takes 10 rounds)
    word Rcon[10] = {0x01000000, 0x02000000, 0x04000000, 0x08000000, 0x10000000,
                0x20000000, 0x40000000, 0x80000000, 0x1b000000, 0x36000000};

    /**
     * Convert four byte s to one word.
     */
    word Word(byte& k1, byte& k2, byte& k3, byte& k4)
    {
        word result(0x00000000);
        word temp;
        temp = k1.to_ulong();  // K1
        temp <<= 24;
        result |= temp;
        temp = k2.to_ulong();  // K2
        temp <<= 16;
        result |= temp;
        temp = k3.to_ulong();  // K3
        temp <<= 8;
        result |= temp;
        temp = k4.to_ulong();  // K4
        result |= temp;
        return result;
    }

    /**
```

```
 *  Cyclic left shift by byte
 *  That is to say, [a0, a1, a2, a3] becomes [a1, a2, a3, a0]
 */
word RotWord(word& rw)
{
   word high = rw << 8;
   word low = rw >> 24;
   return high | low;
}

/**
 *  S-box transformation for each byte in input word
 */
word SubWord(const word& sw)
{
   word temp;
   for(int i=0; i<32; i+=8)
   {
      int row = sw[i+7]*8 + sw[i+6]*4 + sw[i+5]*2 + sw[i+4];
      int col = sw[i+3]*8 + sw[i+2]*4 + sw[i+1]*2 + sw[i];
      byte val = S_Box[row][col];
      for(int j=0; j<8; ++j)
         temp[i+j] = val[j];
   }
   return temp;
}

/**
 *  Key Extension Function - Extended 128-bit key to w[4*(Nr+1)]
 */
void KeyExpansion(byte key[4*Nk], word w[4*(Nr+1)])
{
   word temp;
```

```
    int i = 0;
    //The first four of w [] are input key s
    while(i < Nk)
    {
        w[i] = Word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3]);
        ++i;
    }

    i = Nk;

    while(i < 4*(Nr+1))
    {
        temp = w[i-1]; //Record the previous word
        if(i % Nk == 0)
            w[i] = w[i-Nk] ^ SubWord(RotWord(temp)) ^ Rcon[i/Nk-1];
        else
            w[i] = w[i-Nk] ^ temp;
        ++i;
    }
}

int main()
{
    byte key[16] = {0x24, 0x75, 0xA2, 0xB3,
                    0x34, 0x75, 0x56, 0x88,
                    0x31, 0xE2, 0x12, 0x00,
                    0x13, 0xAA, 0x54, 0x87};

    word w[4*(Nr+1)];

    cout << "KEY IS: ";
    for(int i=0; i<16; ++i)
        cout << hex << key[i].to_ulong() << " ";
```

```
    cout << endl;


    KeyExpansion(key, w);
    //Testing
    for(int i=0; i<4*(Nr+1); ++i)
       cout << "w[" << dec << i << "] = " << hex << w[i].to_ulong() << endl;


    return 0;
}
```

Output: