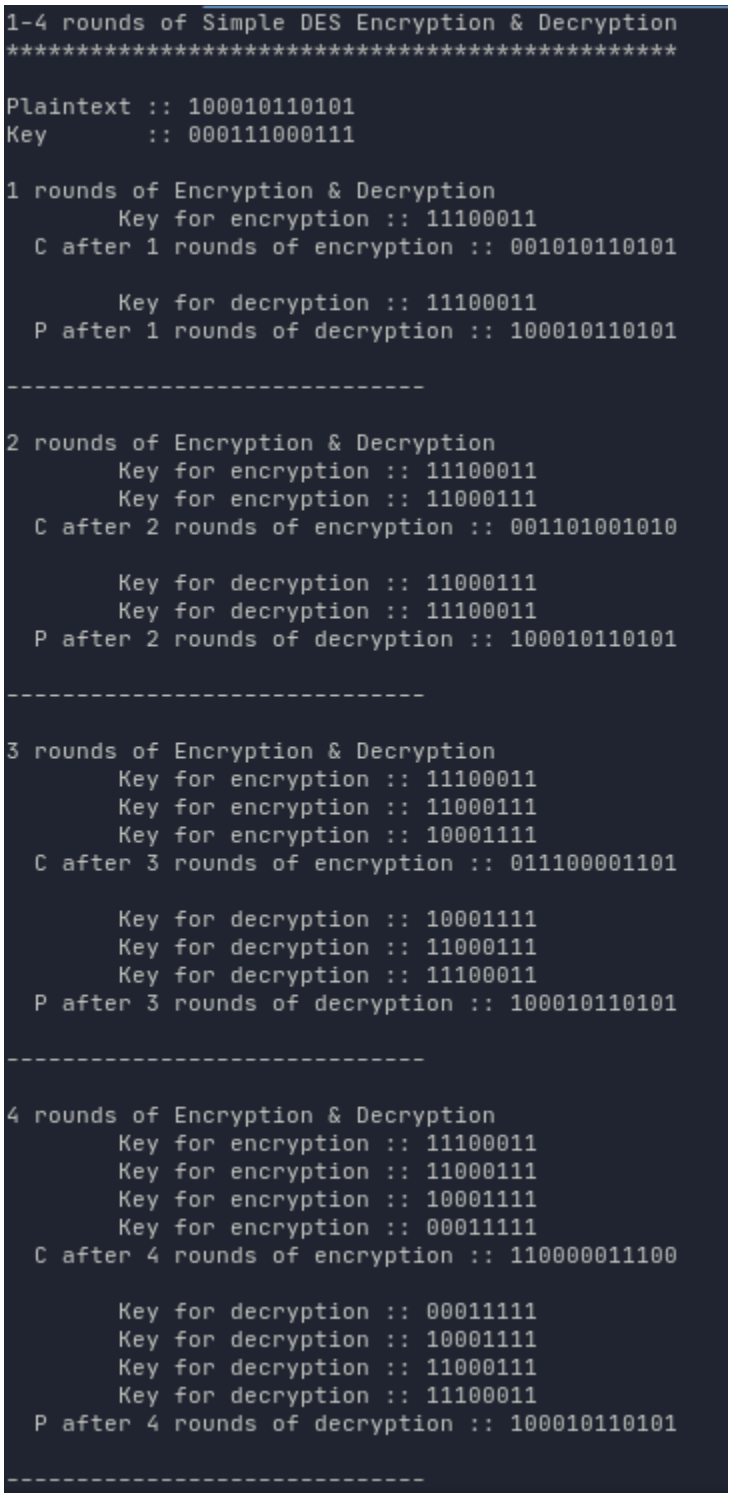
Tristan Erney
November 16th, 2021
Intro to Cryptology
Programming Assignment 2

1)



```
1-4 rounds of Simple DES Encryption & Decryption
************************************************

Plaintext :: 100010110101
Key       :: 000111000111

1 rounds of Encryption & Decryption
        Key for encryption :: 11100011
  C after 1 rounds of encryption :: 001010110101

        Key for decryption :: 11100011
  P after 1 rounds of decryption :: 100010110101

------------------------------

2 rounds of Encryption & Decryption
        Key for encryption :: 11100011
        Key for encryption :: 11000111
  C after 2 rounds of encryption :: 001101001010

        Key for decryption :: 11000111
        Key for decryption :: 11100011
  P after 2 rounds of decryption :: 100010110101

------------------------------

3 rounds of Encryption & Decryption
        Key for encryption :: 11100011
        Key for encryption :: 11000111
        Key for encryption :: 10001111
  C after 3 rounds of encryption :: 011100001101

        Key for decryption :: 10001111
        Key for decryption :: 11000111
        Key for decryption :: 11100011
  P after 3 rounds of decryption :: 100010110101

------------------------------

4 rounds of Encryption & Decryption
        Key for encryption :: 11100011
        Key for encryption :: 11000111
        Key for encryption :: 10001111
        Key for encryption :: 00011111
  C after 4 rounds of encryption :: 110000011100

        Key for decryption :: 00011111
        Key for decryption :: 10001111
        Key for decryption :: 11000111
        Key for decryption :: 11100011
  P after 4 rounds of decryption :: 100010110101

------------------------------
```

2)

```
Item #2 ::
Plaintext  :: 11110000111101010101010110101010101000011110000
Key        :: 010011001

Ciphertext :: 10111110101001110101011111010010111010001110100 0

Plaintext` :: 11110000111101010101010110101010101000011110000
```

3)

```
Item #3 ::
P1 :: 11001100110011001100110011001100110011001100110011001100
P2 :: 11001100110010001100110011001100110011001100110011001100

C1 :: 00010001000000011000001000101111110111111101011 1
C2 :: 00010001000001011111001111110100111011101100111 0
```

s_des.cpp

```
#include <iostream>
#include <stdlib.h>
#include <stdint.h>
#include <bitset>

template <int I>
void printb(uint64_t n) {
  std::bitset<I> x(n);
  std::cout << x;
}

uint8_t sbox1[2][8] = {
                    { 5, 2, 1, 6, 3, 4, 7, 0 },
                    { 1, 4, 6, 2, 0, 7, 5, 3 }
};

uint8_t sbox2[2][8] = {
                    { 4, 0, 6, 5, 7, 1, 3, 2 },
                    { 5, 3, 0, 7, 6, 2, 1, 4 }
};

uint8_t E(uint8_t X) {
  uint8_t e = 0;

  e |= ((0x20 & X) >> 5) << 7;
  e |= ((0x10 & X) >> 4) << 6;
  e |= ((0x04 & X) >> 2) << 5;
  e |= ((0x08 & X) >> 3) << 4;
  e |= ((0x04 & X) >> 2) << 3;
  e |= ((0x08 & X) >> 3) << 2;
  e |= ((0x02 & X) >> 1) << 1;
  e |= ((0x01 & X) >> 0) << 0;

  return e;
}

uint8_t F(uint8_t X, uint8_t K) {
  uint8_t e = E(X);
  uint8_t e_xor_k = e ^ K;

  uint8_t sbox_1 = sbox1[e_xor_k >> 7][0x07 & (e_xor_k >> 4)];
  uint8_t sbox_2 = sbox2[(0x0f & e_xor_k) >> 3][0x07 & e_xor_k];

  return (sbox_1 << 3) + sbox_2;
}

uint16_t sdes_encrypt(uint16_t P, uint16_t K, uint8_t max_rounds) {
```

```
  uint8_t round = 0;

  uint8_t L = (uint8_t)(P >> 6);
  uint8_t R = (uint8_t)(0x3f & P);

  for (round = 0; round < max_rounds; round += 1) {
    uint8_t K_ = ((K << round) | (K >> (9 - round))) >> 1;

    std::cout << "\tKey for encryption :: ";
    printb<8>(K_);
    std::cout << "\n";

    uint8_t L_ = R;
    uint8_t R_ = L ^ F(R, K_);

    L = L_;
    R = R_;
  }

  return (R << 6) + L;
}

uint16_t sdes_decrypt(uint16_t C, uint16_t K, int8_t max_rounds) {
  int8_t round = max_rounds - 1;

  uint8_t R = (uint8_t)(C >> 6);
  uint8_t L = (uint8_t)(0x3f & C);

  for (; round >= 0; round -= 1) {
    uint8_t K_ = ((K << round) | (K >> (9 - round))) >> 1;

    std::cout << "\tKey for decryption :: ";
    printb<8>(K_);
    std::cout << "\n";

    uint8_t R_ = L;
    uint8_t L_ = R ^ F(R_, K_);

    L = L_;
    R = R_;

  }

  return (L << 6) + R;
}



int main() {
```

```
  std::cout << "1-4 rounds of Simple DES Encryption & Decryption\n";
  std::cout << "**********************************************\n\n";

  uint16_t P = 0x08b5;
  uint16_t K = 0x01c7;

  std::cout << "Plaintext :: ";
  printb<12>(P);
  std::cout << "\n";

  std::cout << "Key       :: ";
  printb<12>(K);
  std::cout << "\n\n";

  for (int i = 1; i <= 4; i += 1) {
    std::cout << i << " rounds of Encryption & Decryption\n";

    uint16_t C = sdes_encrypt(P, K, i);
    std::cout << "  C after " << i << " rounds of encryption :: ";
    printb<12>(C);
    std::cout << "\n\n";

    uint16_t P_ = sdes_decrypt(C, K, i);
    std::cout << "  P after " << i << " rounds of decryption :: ";
    printb<12>(P_);
    std::cout << "\n\n" << "------------------------------\n\n";
  }

  return 0;
}
```

s_des_cbc.cpp

```
#include <iostream>
#include <stdlib.h>
#include <stdint.h>
#include <bitset>

template <int I>
void printb(uint64_t n) {
  std::bitset<I> x(n);
  std::cout << x;
}

uint8_t sbox1[2][8] = {
                    { 5, 2, 1, 6, 3, 4, 7, 0 },
                    { 1, 4, 6, 2, 0, 7, 5, 3 }
};

uint8_t sbox2[2][8] = {
                    { 4, 0, 6, 5, 7, 1, 3, 2 },
                    { 5, 3, 0, 7, 6, 2, 1, 4 }
};

uint8_t E(uint8_t X) {
  uint8_t e = 0;

  e |= ((0x20 & X) >> 5) << 7;
  e |= ((0x10 & X) >> 4) << 6;
  e |= ((0x04 & X) >> 2) << 5;
  e |= ((0x08 & X) >> 3) << 4;
  e |= ((0x04 & X) >> 2) << 3;
  e |= ((0x08 & X) >> 3) << 2;
  e |= ((0x02 & X) >> 1) << 1;
  e |= ((0x01 & X) >> 0) << 0;

  return e;
}

uint8_t F(uint8_t X, uint8_t K) {
  uint8_t e = E(X);
  uint8_t e_xor_k = e ^ K;

  uint8_t sbox_1 = sbox1[e_xor_k >> 7][0x07 & (e_xor_k >> 4)];
  uint8_t sbox_2 = sbox2[(0x0f & e_xor_k) >> 3][0x07 & e_xor_k];

  return (sbox_1 << 3) + sbox_2;
}

uint16_t sdes_encrypt(uint16_t P, uint16_t K, uint8_t max_rounds) {
```

```
  uint8_t round = 0;

  uint8_t L = (uint8_t)(P >> 6);
  uint8_t R = (uint8_t)(0x3f & P);

  for (round = 0; round < max_rounds; round += 1) {
    uint8_t K_ = ((K << round) | (K >> (9 - round))) >> 1;

    uint8_t L_ = R;
    uint8_t R_ = L ^ F(R, K_);

    L = L_;
    R = R_;
  }

  return (R << 6) + L;
}

uint16_t sdes_decrypt(uint16_t C, uint16_t K, int8_t max_rounds) {
  int8_t round = max_rounds - 1;

  uint8_t R = (uint8_t)(C >> 6);
  uint8_t L = (uint8_t)(0x3f & C);

  for (; round >= 0; round -= 1) {
    uint8_t K_ = ((K << round) | (K >> (9 - round))) >> 1;

    uint8_t R_ = L;
    uint8_t L_ = R ^ F(R_, K_);

    L = L_;
    R = R_;

  }

  return (L << 6) + R;
}

uint64_t sdes_cbc_encrypt(uint64_t P, uint16_t K, uint16_t IV, uint8_t max_rounds) {
  uint8_t sections = 4; // 48 / 12 = 4;

  uint16_t C[4] = {0, 0, 0, 0};
  uint16_t P_[4] = {
    (uint16_t)(0x0fff & (uint16_t)(P >> 36)),
    (uint16_t)(0x0fff & (uint16_t)(P >> 24)),
    (uint16_t)(0x0fff & (uint16_t)(P >> 12)),
    (uint16_t)(0x0fff & (uint16_t)P)
  };
```

```
  uint16_t C_ = IV;
  for (int i = 0; i < sections; i += 1) {
    C_ = P_[i] ^ C_;
    C[i] = sdes_encrypt(C_, K, max_rounds);
    C_ = C[i];
  }

  return (uint64_t)(C[0]) << 36 | (uint64_t)(C[1]) << 24 | (uint64_t)(C[2]) << 12 | (uint64_t)(C[3]);
}

uint64_t sdes_cbc_decrypt(uint64_t C, uint16_t K, uint16_t IV, uint8_t max_rounds) {
  uint8_t sections = 4; // 48 / 12 = 4;

  uint16_t P[4] = {0, 0, 0, 0};
  uint16_t C_[4] = {
    (uint16_t)(0x0fff & (uint16_t)(C >> 36)),
    (uint16_t)(0x0fff & (uint16_t)(C >> 24)),
    (uint16_t)(0x0fff & (uint16_t)(C >> 12)),
    (uint16_t)(0x0fff & (uint16_t)C)
  };

  uint16_t P_ = IV;
  for (int i = 0; i < sections; i += 1) {
    P[i] = sdes_decrypt(C_[i], K, max_rounds) ^ P_;
    P_ = C_[i];
  }

  return (uint64_t)(P[0]) << 36 | (uint64_t)(P[1]) << 24 | (uint64_t)(P[2]) << 12 | (uint64_t)(P[3]);
}

int main() {

  uint64_t P  = 0b111100001111010101010101101010101010000011110000;
  uint16_t K  = 0b010011001;
  uint16_t IV = 0b111000111000;
  uint64_t expected_C = 0b101111101010011101010111110100101110100011101000;

  std::cout << "Item #2 :: \n";
  std::cout << "Plaintext  :: ";
  printb<48>(P);
  std::cout << "\n";

  std::cout << "Key        :: ";
  printb<9>(K);
  std::cout << "\n\n";

  uint64_t C  = sdes_cbc_encrypt(P, K, IV, 4);
  std::cout << "Ciphertext :: ";
  printb<48>(C);
```

```
  std::cout << "\n\n";

  uint64_t P_ = sdes_cbc_decrypt(C, K, IV, 4);
  std::cout << "Plaintext` :: ";
  printb<48>(P_);
  std::cout << "\n\n\n";


  std::cout << "Item #3 :: \n";
  uint64_t P1 = 0b110011001100110011001100110011001100110011001100;
  uint64_t P2 = 0b110011001100100011001100110011001100110011001100;

  uint64_t C1 = sdes_cbc_encrypt(P1, K, IV, 4);
  uint64_t C2 = sdes_cbc_encrypt(P2, K, IV, 4);

  std::cout << "P1 :: ";
  printb<48>(P1);
  std::cout << "\n";

  std::cout << "P2 :: ";
  printb<48>(P2);
  std::cout << "\n\n";

  std::cout << "C1 :: ";
  printb<48>(C1);
  std::cout << "\n";

  std::cout << "C2 :: ";
  printb<48>(C2);
  std::cout << "\n";

  return 0;
}
```