Міністерство освіти і науки України  
НТУУ «КПІ ім. І. Сікорського»

Кафедра цифрових технологій в енергетиці

Безпека інформаційних систем

Лабораторна робота № 4

«Симетричні криптосистеми. Алгоритм DES»

Варіант №1

Виконав:  
студент 2-го курсу НН ІАТЕ  
групи ТР-15  
Руденко Владислав Ігорович

Перевірив:

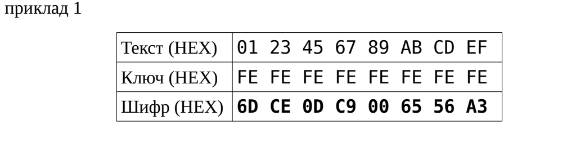
доц. Онисько А. І.

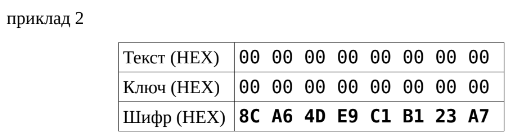
Київ – 2022

**Завдання**

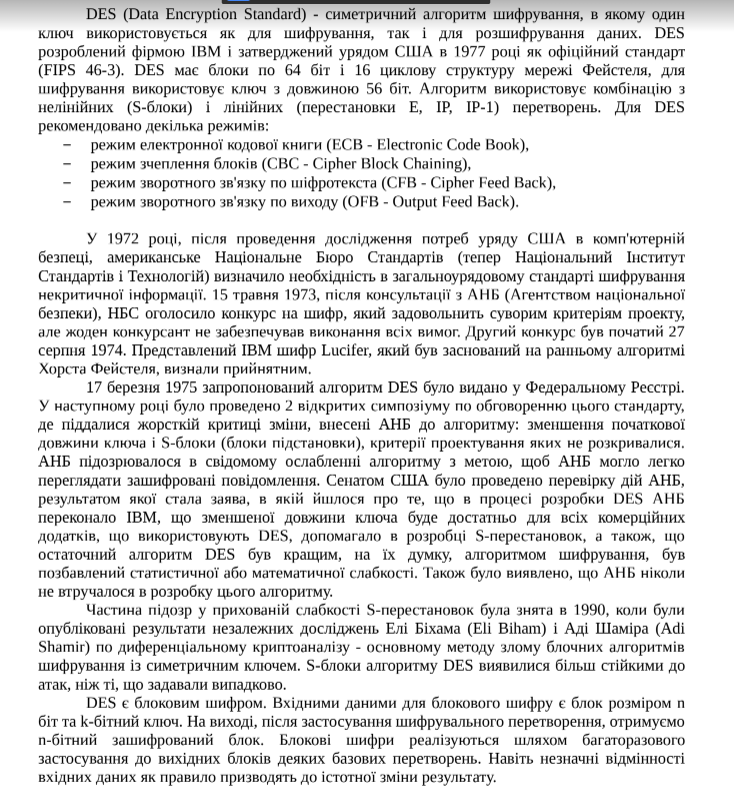
1. Необхідно зашифрувати перші вісім літер свого прізвища, імені та по-батькові залатинською транслітерацією за допомогою алгоритму DES. В якості пароля взяти слово«password». Для зменшення кількості обчислень в алгоритмі DES слід обмежитись лише одним раундом.

2. Написати програмне забезпечення, що реалізує алгоритм DES. Перевірити його роботу для наступних прикладів:



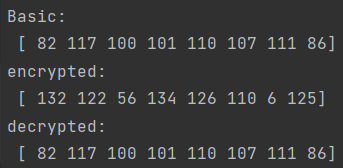


**Теоретичні відомості**



**Результати виконання**

****

****

**Приклади:**

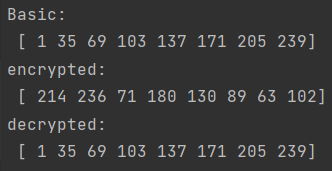
****

Рисунок 1 Приклад 1

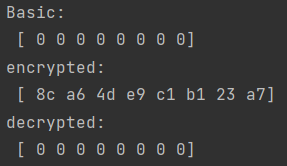


Рисунок 2 Приклад 2

**Висновок**

В ході виконання лабораторної роботи №4 здобуто навички симетричного шифрування методом DES. Результатом роботи є алгоритм DES, який здатен шифрувати та дешифрувати деякий текст. В роботі представлено як і власний приклад задачі, так і виконання прикладів за для перевірки коректності роботи програми.

**Лістинг програми**

#include <iostream>  
#include <ostream>  
#include <stdint.h>  
  
using namespace std;  
  
#define Shift(x, L) ((((x) << (L)) | ((x) >> (-(L) & 27))) & (((uint64\_t)1 << 32) - 1))  
static const uint8\_t Sbox[8][4][16] = {  
 {  
 {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},  
 },  
 {  
 {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 },  
 },  
 {  
 {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},  
 },  
 {  
 {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},  
 },  
 {  
 {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 },  
 },  
 {  
 {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},  
 },  
 {  
 {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},  
 },  
 {  
 {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},  
 },  
};  
  
void DES(uint8\_t \* to, uint8\_t mode, uint8\_t \* keys8b, uint8\_t \* from, size\_t length);  
void Keysgenerator(uint64\_t key64b, uint64\_t \* keys48b);  
void circlepreparation(uint8\_t mode, uint32\_t \* N1, uint32\_t \* N2, uint64\_t \* keys48b);  
uint32\_t Function(uint32\_t block32b, uint64\_t key48b);  
uint32\_t substitutions(uint64\_t block48b);  
uint64\_t Transfer8to64(uint8\_t \* keys8b)  
{  
 uint64\_t b64b;  
 for (uint8\_t \*i = keys8b; i < keys8b + 8; ++i) {  
 b64b = (b64b << 8) | \*i;  
 }  
 return b64b;  
}  
static void print(uint8\_t \* array, size\_t length, string text = "")  
{  
 cout << text+":" << endl <<" [";  
 for (size\_t i = 0; i < length; ++i)  
 cout << " " << +array[i];  
 cout << "]" << endl;  
}  
  
int main(void) {  
 uint8\_t  
 encrypted[64],  
 decrypted[64],  
 text[64] = {'R','u','d','e','n','k','o','V'},  
 keys8b[8] = {'p','a','s','s','w','o','r','d'};  
 size\_t length = 8;  
 print(text, length, "Basic");  
 DES(encrypted, 'E', keys8b, text, length);  
 print(encrypted,length, "encrypted");  
 DES(decrypted, 'D', keys8b, encrypted, length);  
 print(decrypted, length,"decrypted");  
}  
  
void DES(uint8\_t \* output, uint8\_t mode, uint8\_t \* keys8b, uint8\_t \* input, size\_t length) {  
 uint64\_t keys48b[16] = {0};  
 uint32\_t Left, Right;  
  
 Keysgenerator(Transfer8to64(keys8b), keys48b);  
  
 for (size\_t i = 0; i < length; i += 8)  
 {  
 {  
 uint64\_t b64b = Transfer8to64(input + i);  
  
 const uint8\_t IP[64] = {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,};  
 uint64\_t new\_b64b = 0;  
 for (uint8\_t j = 0 ; j < 64; ++j) {  
 new\_b64b |= ((b64b >> (64 - IP[j])) & 0x01) << (63 - j);  
 }  
 Left = (uint32\_t)(new\_b64b >> 32);  
 Right = (uint32\_t)(new\_b64b);  
 }  
   
 circlepreparation(mode, &Left, &Right, keys48b);  
 {  
 uint64\_t b64b;  
 b64b = (uint64\_t)Left;  
 b64b = (uint64\_t)(b64b << 32) | Right;  
  
 const uint8\_t FP[64] = {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,};  
 uint64\_t new\_b64b = 0;  
 for (uint8\_t j = 0 ; j < 64; ++j) {  
 new\_b64b |= ((b64b >> (64 - FP[j])) & 0x01) << (63 - j);  
 }  
 for (size\_t j = 0; j < 8; ++j) {  
 (output + i)[j] = (uint8\_t)(new\_b64b >> ((7 - j) \* 8));  
 }  
 }  
 }  
}  
void Keysgenerator(uint64\_t key64b, uint64\_t \* keys48b) {  
 uint32\_t Left = 0, Right = 0;  
 const uint8\_t K1P[28] = {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,};  
 const uint8\_t K2P[28] = {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 ,};  
 for (uint8\_t i = 0; i < 28; ++i) {  
 Left |= ((key64b >> (64 - K1P[i])) & 0x01) << (31 - i);  
 Right |= ((key64b >> (64 - K2P[i])) & 0x01) << (31 - i);  
 }  
   
 const uint8\_t CP[48] = {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,};  
 uint64\_t b56b;  
 uint8\_t n;  
 for (uint8\_t i = 0; i < 16; ++i) {  
 switch(i) {  
 case 0: case 1: case 8: case 15: n = 1; break;  
 default: n = 2; break;  
 }  
  
 Left = Shift(Left, n);  
 Right = Shift(Right, n);  
 b56b = (Left >> 4);  
 b56b = ((b56b << 32) | Right) << 4;  
 for (uint8\_t i = 0 ; i < 48; ++i) {  
 keys48b[i] |= ((b56b >> (64 - CP[i])) & 0x01) << (63 - i);  
 }  
 }  
}  
void circlepreparation(uint8\_t mode, uint32\_t \* Left, uint32\_t \* Right, uint64\_t \* keys48b) {  
 switch(mode) {  
 case 'E': {  
 for (int8\_t round = 0; round < 16; ++round) {  
 uint32\_t temp = \*Right;  
 \*Right = Function(\*Right, keys48b[round]) ^ \*Left;  
 \*Left = temp;  
 }  
 uint32\_t temp = \*Left;  
 \*Left = \*Right;  
 \*Right = temp;  
 break;  
 }  
 case 'D': {  
 for (int8\_t round = 15; round >= 0; --round) {  
 uint32\_t temp = \*Right;  
 \*Right = Function(\*Right, keys48b[round]) ^ \*Left;  
 \*Left = temp;  
 }  
 uint32\_t temp = \*Left;  
 \*Left = \*Right;  
 \*Right = temp;  
 break;  
 }  
 }  
}  
uint32\_t Function(uint32\_t b32b, uint64\_t key48b) {  
 const uint8\_t EP[48] = {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 ,};  
 uint64\_t b48b = 0;  
 for (uint8\_t i = 0 ; i < 48; ++i) {  
 b48b |= (uint64\_t)((b32b >> (32 - EP[i])) & 0x01) << (63 - i);  
 }  
 b48b ^= key48b;  
 b32b = substitutions(b48b);  
   
 const uint8\_t P[32] = {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,};  
 uint32\_t new\_b32b = 0;  
 for (uint8\_t i = 0 ; i < 32; ++i) {  
 new\_b32b |= ((b32b >> (32 - P[i])) & 0x01) << (31 - i);  
 }  
 return new\_b32b;  
}  
uint32\_t substitutions(uint64\_t b48b) {  
 uint8\_t bs4b[4], bs6b[8] = {0};  
 for (uint8\_t i = 0; i < 8; ++i) {  
 bs6b[i] = (b48b >> (58 - (i \* 6))) << 2;  
 }  
 {  
 uint8\_t b2b, b4b;  
  
 for (uint8\_t i = 0, j = 0; i < 8; i += 2, ++j) {  
 b2b = ((bs6b[i] >> 6) & 0x2) | ((bs6b[i] >> 2) & 0x1);  
 b4b = (bs6b[i] >> 3) & 0xF;  
 bs4b[j] = Sbox[i][b2b][b4b];  
  
 b2b = ((bs6b[i+1] >> 6) & 0x2) | ((bs6b[i+1] >> 2) & 0x1);  
 b4b = (bs6b[i+1] >> 3) & 0xF;  
 bs4b[j] = (bs4b[j] << 4) | Sbox[i+1][b2b][b4b];  
 }   
 }  
 uint32\_t b32b;  
 for (uint8\_t \*p = bs4b; p < bs4b + 4; ++p) {  
 b32b = (b32b << 8) | \*p;  
 }  
 return b32b;  
   
}