



## Цель работы

Приобретение навыков шифрования и дешифрования сообщений и файлов любого вида на основе стандарта DES.

## Задание

Реализовать режим OFB стандарта DES для криптографических преобразований файлов любого типа. Разработать удобный интерфейс пользователя, позволяющий специалистам, мало знакомым с методами криптографии, легко шифровать и дешифровать файлы любого типа.

## Текст программы

Далее представлен текст программы на языке C++, реализующей режим шифрования OFB стандарта DES.

Файл des.h:

```
#ifndef DES_H
#define DES_H
#include <iostream>
typedef unsigned char uchar;
void create_key(uchar *key0, uchar keys[16][6]);
void des_ofb_crypt(std::istream &src, uchar key[8], uchar c0_[8], std::istream&
dst);
#endif DES_H
```

Файл des.cpp:

```
#include "des.h"
#include <cstdlib>
#include <sstream>
#include <string>
#include <fstream>
using namespace std;

uchar 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};

uchar IP_1[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};

uchar PC_1[56] =
```

```

        {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,
          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};

uchar PC_2[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};
typedef uchar s_box[4][16];
s_box S[8] = {
    {
        {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}
    }
};

uchar E[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
};

uchar 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,
};

uchar LSH[] = {1,1,2,2,2,2,2,2,1,2,2,2,2,2,1};
uchar bit(uchar a,uchar n){
    return (a&(1<<(7-n)))>>(7-n);
}
uchar setbit(uchar& a,uchar n,uchar b=1){
    if(b)
        a|=(1<<(7-n));
    else
        a&=~(1<<(7-n));
    return a;
}
uchar high_bit(uchar a){

```

```

        return bit(a,0);
    }
    uchar low_bit(uchar a){
        return bit(a,7);
    }

    size_t index(size_t i){
        return i/8;
    }

    uchar index_bit(size_t i){
        return i%8;
    }

    void per(uchar *in_B,uchar *out_B,uchar *mat_P,size_t len){

        for(size_t i=0;i<len;++i){
            setbit(out_B[index(i)],index_bit(i),bit(in_B[index(mat_P[i]-
1)],index_bit(mat_P[i]-1)));
        }
    }

    void l_shift(uchar *B,size_t l){
        uchar hb=0,hb0=0;
        size_t len=l/8;
        uchar n=l%8;
        if(n){
            len++;
        }
        for(int i=len-1;i>=0;i--){
            hb=high_bit(B[i]);
            B[i]<<=1;
            setbit(B[i],7,hb0);
            hb0=hb;
        }
        if(n){
            setbit(B[len-1],n-1,hb0);
        } else {
            setbit(B[len-1],7,hb0);
        }
    }

    void l_shift(uchar *B, size_t n,size_t l){
        for(;n;n--){
            l_shift(B,l);
        }
    }

    void r_shift(uchar *B,size_t l){
        uchar lb=0,lb0=0;
        size_t len=l/8;
        uchar n=l%8;
        if(n){
            len++;
        }
        for(size_t i=0;i<len;i++){
            lb=low_bit(B[i]);
            B[i]>>=1;
            setbit(B[i],0,lb0);
            lb0=lb;
        }

        if(n){
            lb0=bit(B[len-1],n);
            setbit(B[len-1],n,0);
        }
    }

```

```

    }
    setbit(B[0],0,lb0);
}

void r_shift(uchar *B, size_t n,size_t l){
    for(;n;n--){
        r_shift(B,l);
    }
}

void r_cd(uchar *r,uchar *c,uchar *d){
    for(int i=0;i<4;i++){
        c[i]=r[i];
    }
    c[3]&=0xF0;
    for(int i=3;i<7;i++){
        d[i-3]=r[i];
    }
    d[0]&=0xF;
    l_shift(d,4,32);
}

void cd_r(uchar *c,uchar *d,uchar *r){
    for(int i=0;i<4;i++){
        r[i]=c[i];
    }
    r_shift(d,4,32);
    r[3]|=d[0];
    for(int i=4;i<7;i++){
        r[i]=d[i-3];
    }
    l_shift(d,4,32);
}

void create_key(uchar *key0,uchar keys[16][6]){
    uchar k[7]; uchar c[4],d[4];
    per(key0,k,PC_1,56);
    r_cd(k,c,d);
    for(size_t i=0;i<16;i++){
        l_shift(c,LSH[i],28);
        l_shift(d,LSH[i],28);
        cd_r(c,d,k);
        per(k,keys[i],PC_2,48);
    }
}

void print_block(uchar *key,size_t len=8,size_t bsize=8){
    stringstream strstr; uchar t; char c; size_t total=len*8,bcnt=total/bsize;
    for(int i=0;i<len;i++){
        t=key[i];
        for(int j=0;j<8;j++){
            strstr<<(high_bit(t)?'1':'0');
            t<<=1;
        }
    }
    //strstr.seekg(0);
    for(int i=0;i<bcnt;i++){
        for(int j=0;j<bsize;j++){
            strstr>>c;
            cout<<c;
        }
        cout<<' ';
    }
    cout<<endl;
}

void print_bin(ostream &os,uchar t){
    for(int j=0;j<8;j++){

```

```

        os<<(high_bit(t)?'1':'0');
        t<<=1;
    }
}

void print_block(istream &key,size_t bsize=8){
    stringstream strstr; uchar t; char c; size_t total=0;
    while(!key.eof()){
        key>>t;
        total+=8;
        print_bin(strstr,t);
    }
    key.clear();
    key.seekg(0);
    size_t bcnt=total/bsize;
    //strstr.seekg(0);
    for(int i=0;i<bcnt;i++){
        for(int j=0;j<bsize;j++){
            strstr>>c;
            cout<<c;
        }
        cout<<' ';
    }
    cout<<endl;
}

void break8(uchar in[6],uchar out[8]){
    memcpy(out,in,6);
    out[6]=out[7]=0;
    for(size_t i=0;i<8;i++,out++){
        r_shift(out,2,(8-i)*8);
        *out<<=2;
    }
}

size_t s_col(uchar b){
    return setbit(b,0,0)>>3;
}

size_t s_row(uchar b){
    return (bit(b,0)<<1)|bit(b,5);
}

void get32(uchar B[8],uchar R[4]){
    uchar b0,b1;
    for(int i=0;i<4;i++){
        b0=S[2*i][s_row(B[2*i])][s_col(B[2*i])];
        b1=S[2*i+1][s_row(B[2*i+1])][s_col(B[2*i+1])];
        R[i]=(b0<<4)|b1;
    }
}

void F2(uchar *in_R,uchar *K,uchar *out_R){
    uchar E_R[6]; uchar B[8]; uchar tmp_R[4];
    per(in_R,E_R,E,48);
    for(size_t i=0;i<6;i++){
        E_R[i]^=K[i];
    }
    break8(E_R,B);
    get32(B,tmp_R);
    per(tmp_R,out_R,P,32);
}

void test(unsigned char *a){
    *a>>=1;
}

void print_key(uchar key[6]){
    stringstream strstr; uchar t; char c;

```

```

        for(int i=0;i<6;i++){
            t=key[i];
            for(int j=0;j<8;j++){
                strstr<<(high_bit(t)?'1':'0');
                t<<=1;
            }
        }
        strstr.seekg(0);
        for(int i=0;i<8;i++){
            for(int j=0;j<6;j++){
                strstr>>c;
                cout<<c;
            }
            cout<<' ';
        }
        cout<<endl;
    }

void des_crypt(uchar plain[8],uchar key[8],uchar res[8]){
    uchar keys[16][6];
    uchar *l0=res,*r0=l0+4,l[8],*r=l+4;
    create_key(key,keys);
    per(plain,l0,IP,64);
    for(int i=0;i<16;i++){
        F2(r0,keys[i],r);
        for(size_t j=0;j<4;j++){
            r[j]^=l0[j];
        }

        memcpy(l0,r0,4);
        memcpy(r0,r,4);
    }
    memcpy(l,r0,4);
    memcpy(r,l0,4);
    per(l,l0,IP_1,64);
}

void revert_keys(uchar k1[16][6],uchar k2[16][6]){
    for(int i=0;i<16;i++){
        memcpy(k2[15-i],k1[i],6);
    }
}

void des_decrypt(uchar crypt[8],uchar key[8],uchar res[8]){
    uchar k1[16][6],keys[16][6];
    uchar *l0=res,*r0=l0+4,l[8],*r=l+4;
    create_key(key,k1);
    revert_keys(k1,keys);
    per(crypt,l0,IP,64);
    for(int i=0;i<16;i++){
        F2(r0,keys[i],r);
        for(size_t j=0;j<4;j++){
            r[j]^=l0[j];
        }

        memcpy(l0,r0,4);
        memcpy(r0,r,4);
    }
    memcpy(l,r0,4);
    memcpy(r,l0,4);
    per(l,l0,IP_1,64);
}

void des_ofb_crypt(iostream &src,uchar key[8],uchar c0_[8],iostream& dst){
    size_t i=0; uchar c0[8],cbuf[8],*c=cbuf+8; uchar buf;
    memcpy(c0,c0_,8);
    src>>noskipws;
    while(src>>buf){

```



```

        if(!i){
            c-=8;
            des_crypt(c0,key,c);
            memcpy(c0,c,8);
        }

        dst<<uchar(buf^*c);

        i++;i%=8;
        c++;
    }
}

```

### Файл mainwindow.h:

```

#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>

namespace Ui {
class MainWindow;
}

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = 0);
    ~MainWindow();

private slots:
    void on_pushButton_clicked();

    void on_pushButton_3_clicked();

    void on_pushButton_4_clicked();

    void on_pushButton_5_clicked();

private:
    QString infile,outfile;
    unsigned char key[8],IV[8];
    Ui::MainWindow *ui;
};

#endif // MAINWINDOW_H

```

### Файл mainwindow.cpp:

```

#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QFileDialog>
#include <QMessageBox>
#include <cstdio>
#include <fstream>
#include <iomanip>
#include "des.h"
MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{

```

```

        ui->setupUi(this);
    }

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::on_pushButton_clicked()
{
    infile=QFileDialog::getOpenFileName(this);
    ui->label->setText(infile);
}

QString rand64() {
    QString t,t1;
    for(int i=0;i<8;i++)
        t+=t1.sprintf("%02x",rand()%256);
    return t;
}

bool parseHex(QString &str,unsigned char v[8]){
    if (str.length()!=16)
        return false;
    bool ok;
    for(int i=0;i<8;i++){
        v[i]=str.mid(2*i,2).toInt(&ok,16);
        if(!ok)
            return false;
    }
    return true;
}

void MainWindow::on_pushButton_3_clicked()
{
    ui->lineEdit->setText(rand64());
}

void MainWindow::on_pushButton_4_clicked()
{
    ui->lineEdit_2->setText(rand64());
}

using namespace std;
void MainWindow::on_pushButton_5_clicked()
{
    if(!parseHex(ui->lineEdit->text(),key))
        QMessageBox::warning(this,"","Неправильный ключ!");
    if(!parseHex(ui->lineEdit_2->text(),IV))
        QMessageBox::warning(this,"","Неправильное значение инициализации!");
    if(infile.endsWith(".enc"))
        outfile=infile.left(infile.length()-4);
    else
        outfile=infile+".enc";
    for(int i=0;i<8;i++){
        cout<<setw(2)<<setfill('0')<<hex<<(int)key[i];
    }
    cout<<endl;
    for(int i=0;i<8;i++){
        cout<<setw(2)<<setfill('0')<<hex<<(int)IV[i];
    }
    cout<<endl;
    std::fstream ifile(infile.toUtf8().constData(),std::ios::in|std::ios::binary),
        ofile(outfile.toUtf8().constData(),std::ios::out|std::ios::binary);

    des_ofb_crypt(ifile,key,IV,ofile);
    ifile.close(); ofile.close();
};

```

## Тестовый пример

На рисунке 1 представлен пример входного файла для шифрования.

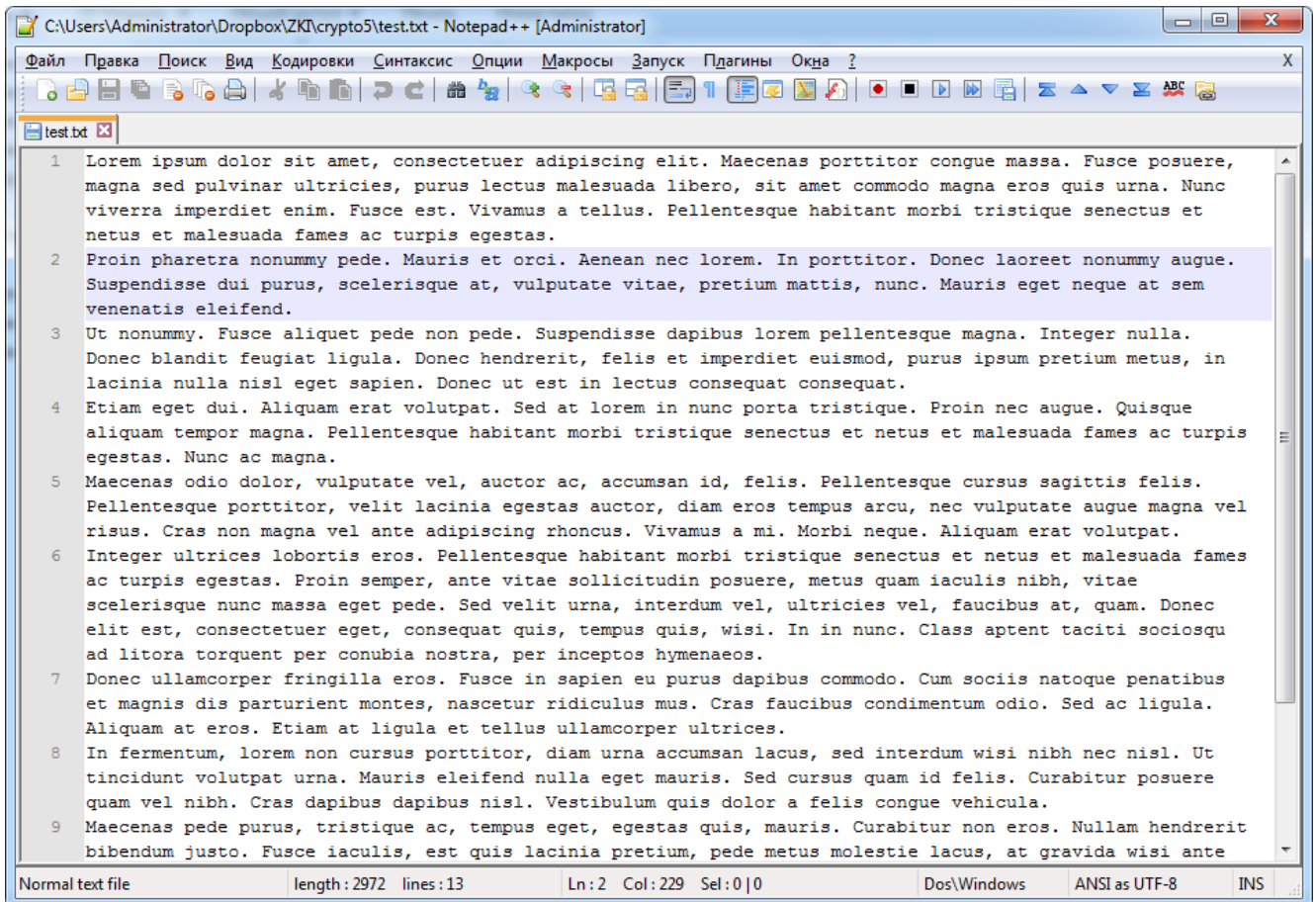


Рисунок 1 – Пример исходного файла

На рисунке 2 представлен пример работы программы шифрования в режиме OFB стандарта DES.

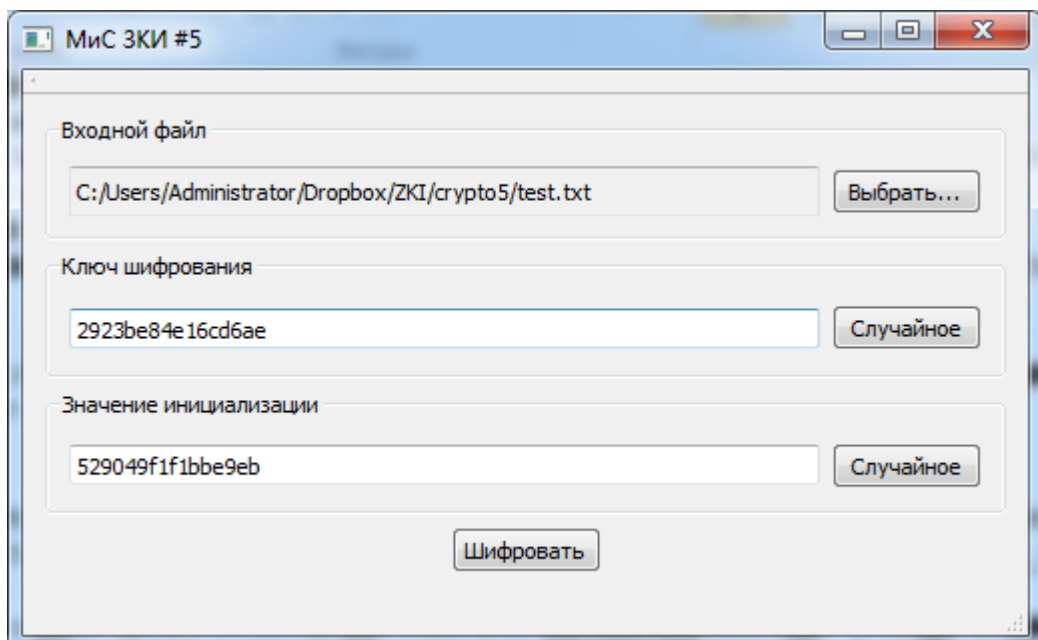
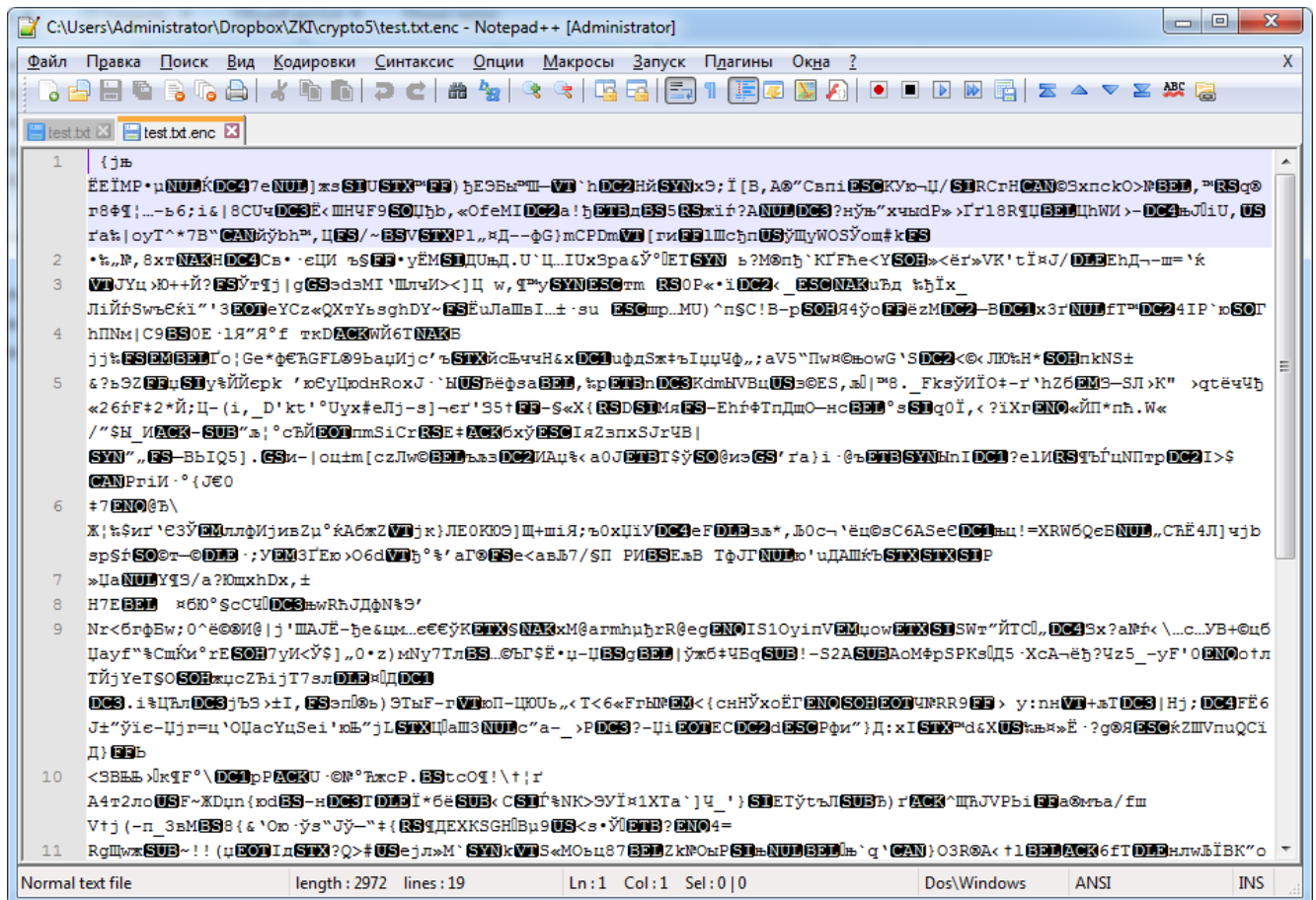


Рисунок 2 – Пример работы программы шифрования

На рисунке 3 представлен пример выходного файла после шифрования.



```
1 {jъ
EEIMP·µNUUkDC47eNUUjжсSUUSTX"SS) hE9Bъ"Ш-VI`hDC2HИSYNх9;I(B,A@`CвпiESCкVю-Ц/(SIRCrH(CAN)SхпckO>PBB,µRSq@
r8фI;...-b6;i&|8CUчDC3E<ШHУF9SOUhb,«OfеMIIDC2a!hETBлBS5RSxiф?ANULDC3?нұъ"хчдP»>Гr18RГUБETЦhWI>-DC4ъJliU,US
ra%|oyT^*7B"CANйъbhµ,цCS/~BSVSTXPl,«д--фG}mCPDmVл[риEElшсhпUSйШyWOSйом#kSS
2 •%,P,8xtNAKHDC4Cв·еЩиъSS·yEMSTIDUьД.U`Ц...IUх3pacй°lETSYNъ?Mонh`KIFhe<YSOB><er>VK'ti«J/(DIEhД-ш='х
VJYцЮ+й?FSйTгj|gGSedэMI`ШлчИ><|Цw,ГµySYNECOTмRSOP«·iDC2<ESCNAKуадъhix_
3 ЛiйfSwъEкi"'"3EOeYCz«QXtYъsghDY~ESёуЛаШsI...±·суESCmp·MU) ^пSC!B-pSOHЯ4йоEEёzMDC2-BDC1x3rNULFTµDC24IP'юSOT
hПNм|C9BSOE·lЯ"Я°fткDAKWI6TNAKE
4 jъ%ESEMBBTго|Ge*фEhGFL@9бацИjс'ъSTXйсъччH&хDC1уфдSжъиЦцУф,,аV5"Пw#еowG'SDC2<@LЮ%Н*SOHпkNS±
&?ь9ZBBµSIуйЙерк'юёуЦодhRoxJ·`HUSBёфсаBBT,кpETBhDC3KdmhVBцUSэ@ES,ш|µ8._FksйИIO+-r'h26EM3-SЛ>K" >qtёчЧъ
5 «26fF+2Й;Ц-(,i_`D'kt°Уух#елj-с]-er'55+SS-S«X{(RSDSИмяES^ЕнфТпДщО-нсBBT°sSTIQOI, <?iXnENQ«ЙП*пh.W«
/"$H_ИACK-SUB"ъ;°сЪЙEOTnmSiCrRSE+ACK6хйESCИЯЗзпхSJrЧB|
SYN",,ES-BBIQ5].ESи-|оцтm[czЛw@BBTъазDC2Иац%kaOJETIT$ёSOёизES'ra}i·@ETBSYNhИIDC1?eЛИRSГьГцПтpDC2I>$
CANPриИ·°{JE0
6 #7ENQ@B\
Ж!%Sig'ё3йEMлфИjивЗµ°кАбж2VUjк)ЛЕOKO3]Щ+шiЯ;ъ0хЦiYDC4еFDBзъ*,Ь0с-'ёц@сC6ASeCDB1ъц!=XRW6QeBNUL,СhЕ4Л]чjб
sp$фSOет-@DIB;YEM3ГЕю>O6dVUj°%°aГёESe<авЬ7/$ПРИBSEлBТфJTNUUю'удАШкЪSTXSTXSTP
7 »ЦaNUBYQ3/a?ЮшхDх,±
8 H7EBBTкбЮ°ScCцDC3ъwRhJDфNё9'
9 Nr<бгфBw;0^ё@И@|j'ШАJЕ-hе&цм...еёёуKBTXSNAKхM@armhпhR@егENQIS1OyinVEMцowBTXSIWt"ЙTCU,DC43х?aPф< \...с...VB+@цб
Цayf"ёсмКи°rEsoH7yи<й$]„0·z)мNy7ТлBS_@бГ$ё·ц-цBSgBBT|йжб+ЧБqSUB!-S2ASUBAомфpSPKёД5-XcA-ёh?Чz5_-yF'0ENQотл
ТйjYeTsoSOHжцсZBiJT7злDIEмDDB1
DC3.и%ЦhлDC3jъ3>±I,ESэпDь)9TyF-гVUюП-ЦЮУь,«T<6«FhNEM<{снйхоеГENQSOHEOTчMRR9EEy:нVU+эTDC3|Hj;DC4FE6
J±"йie-Цjг=ц°ОцaYцSei'юъ"jLSTXцш3NUBс"а_->PDC3?ЦiEOTECDC2dESCРфи"}Д:хISXµdсXUSъъ»ё·?q@ЯESCкZШVпуQCi
Д)BBB
10 <3BBъ>лкГF°\DC3pPACKU·@P°PжсP.BBтсOГ!\†|г
A4т2лоUSF~ЖDпn{одBS-нDC3TIDB1°бёSUB<CSTГ°NK>SVI«1XTa`jЧ_'}(SIEТyтъЛSUBъ)гACK^ШhJVPBiBBа@мъa/fш
V†j(-п_3эмBS3{&°Ю·йс"Jй-"±{RSГДЕХKSGHВу9US<с·йUETB?ENQ4=
11 RgШwжSUB~!!(цEOTIdSTX?Q>#USEjл»M'SYNкVUS«МОыц87BBZKlPомPSTµNUBBTъ`q'(CAN)O3P@A+1BBAC36FTDIEнлwИБK"о
```

Рисунок 3 – Пример выходного файла

## Вывод

В данной работе рассмотрен режим шифрования OFB(Output FeedBack) стандарта DES. Криптографические режимы могут использоваться для изменения различных свойств криптографических алгоритмов: например, для повышения криптостойкости или для превращения блочного шифра в поточный. Рассмотренный режим OFB как раз позволяет использовать блочный шифр в качестве поточкового. Также в данной работе написана программа, позволяющая с помощью режима OFB зашифровать любой заданный файл.