实验五 Hash函数MD5

1811494 刘旭萌

消息填充

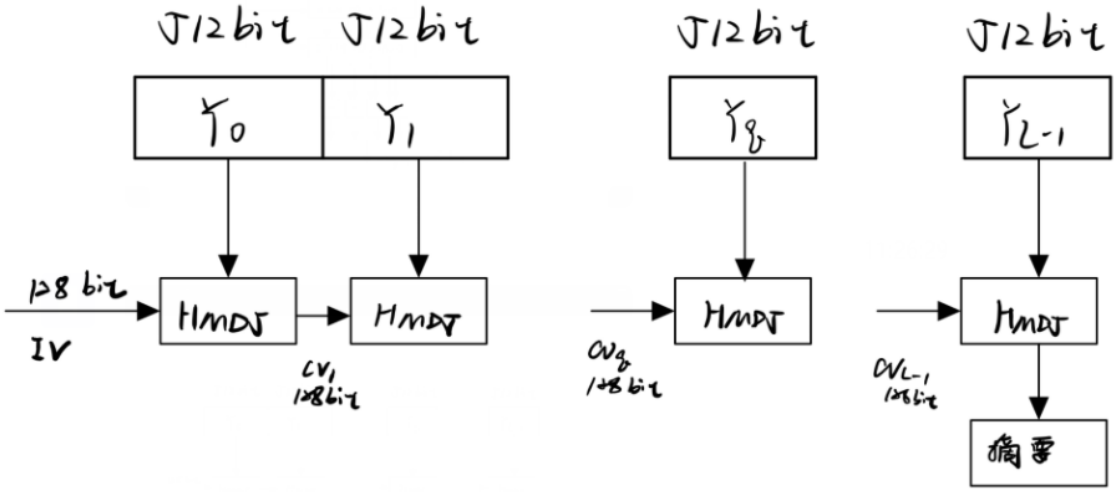
1. 将消息填充为长度模512余448比特，若填充前长度模512等于0，则需要再填充填充512比特，填充方式为最高位为1，后面为0
2. 后64位以**小端方式**填充原消息的长度，若原消息长度大于2^{64}比特，则填充模2^{64}的余数

```
void smallbit(int len, newint& a)
{
    int i = 0;
    len *= 8;
    for (int i = 0; i < 4; i++)
    {
        a.a[i] = len % 256;//小端：将低有效位填入低地址
        len /= 256;
    }
}
```

消息分组

明文每512比特为一组进行加密

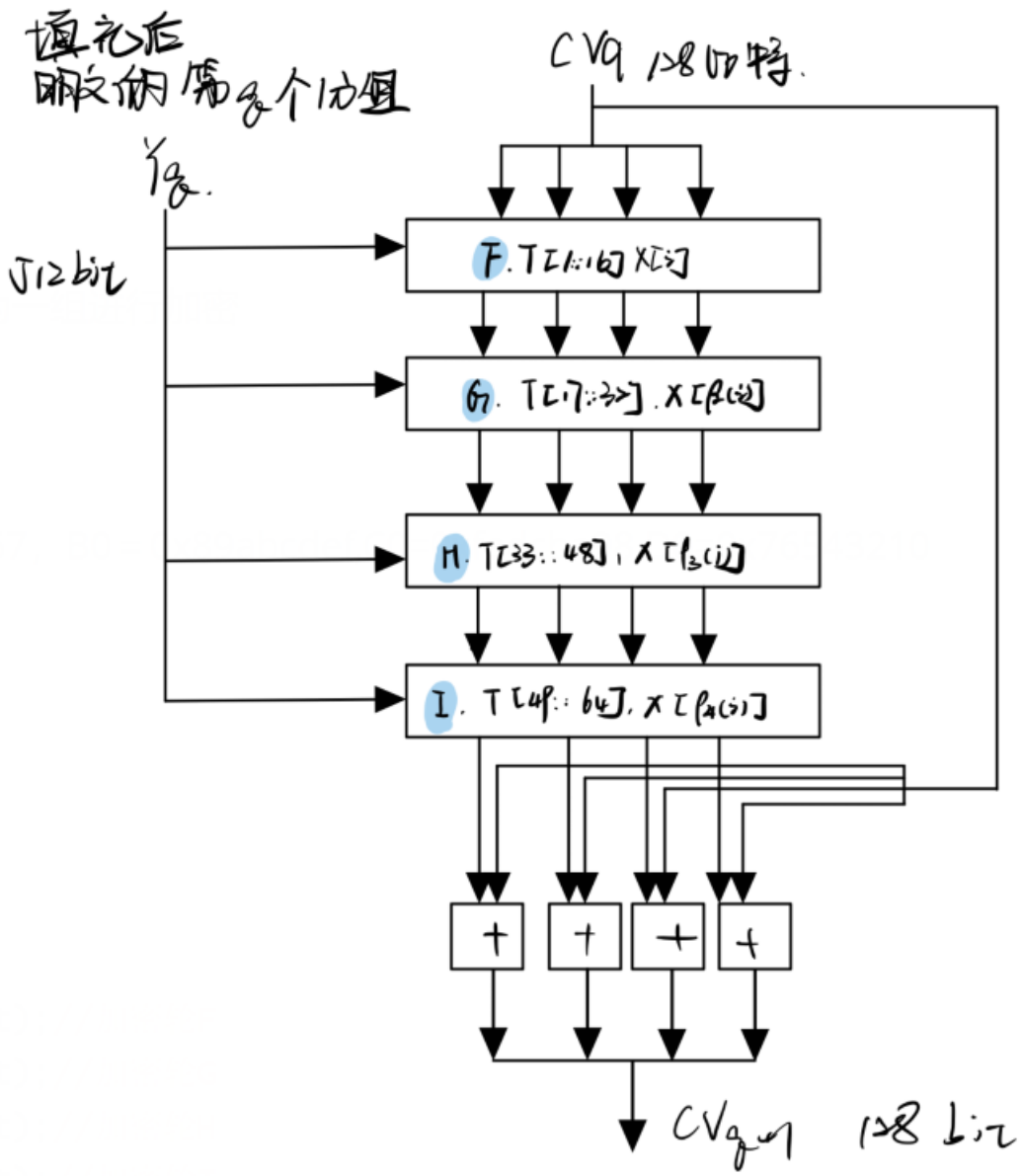
加密过程



$$CV_0 = IV;$$
$$CV_{q+1} = CV_q + RF_I[Y_q, RF_H[Y_q, RF_G[Y_q, RF_F[Y_q, CV_q]]]];$$
$$MD = CV_L;$$

+为模2^{32}加法

初值： A = 0x01234567，B = 0x89abcdef,C=0xfedcba98,D=0x76543210，以**小端方式**存储



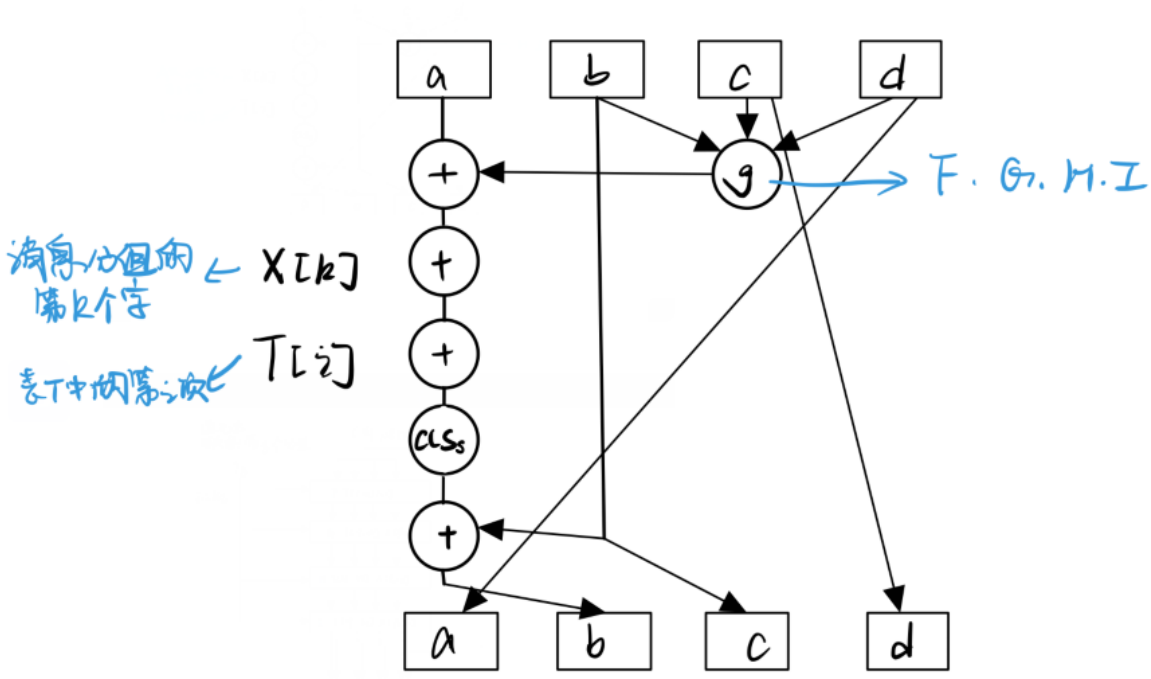
其中

$$\rho_2(x) = (1 + 5i) \bmod 16$$
$$\rho_3(x) = (5 + 3i) \bmod 16$$
$$\rho_4(x) = 7i \bmod 16$$

对于每轮加密

```
enF(i, txt);//加密轮F
enG(i, txt);//加密轮G
enH(i, txt);//加密轮H
enI(i, txt);//加密轮I
for (int j = 0; j < 4; j++)
{
    txt.fabcd[0][j]=add_(txt.fabcd[0][j], txt.iabcd[16][j]);
}
```

每个加密轮，加密逻辑类似



其中的加密函数g对于每轮有不同的计算方法

```
enF: (b & c) | (~b & d);
enG: (b & d) | (c & ~d);
enH: (b ^c^ d);
enI: c^(b|(~d));
```

对应实验指导书中

```
#define F(x, y, z) (((x) & (y)) | ((~x) & (z)))          //F 函数
```

```
#define G(x, y, z) (((x) & (z)) | ((y) & (~z)))          //G 函数
#define H(x, y, z) ((x) ^ (y) ^ (z))                     //H 函数
#define I(x, y, z) ((y) ^ ((x) | (~z)))                  //I 函数
```

CLS_s为左移相应位数

步数	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1（轮数）	7	12	17	22	7	12	17	22	7	12	17	22	7	12	17	22
2	5	9	14	20	5	9	14	20	5	9	14	20	5	9	14	20
3	4	11	16	23	4	11	16	23	4	11	16	23	4	11	16	23
4	6	10	21	6	6	10	21	6	6	10	21	6	6	10	21	6

程序功能即运行结果

数据结构

newint数据结构用于处理小端存储

```
struct newint
{
    int a[4];
    newint();
    void copy(newint b);
    void print();
    unsigned int value();//以小端方式读入和设置值
    void set(unsigned int);
    newint(char*);
};
```

text数据结构用于保存以此md5计算中的各个量

```
struct text
{
    int round;
    newint m[1500][16];//明文
    newint fabcd[17][4];//分别存储F、G、H、I四个部分计算过程量
    newint gabcd[17][4];
    newint habcd[17][4];
    newint iabcd[17][4];

    newint cipher[4];
    void cipherprint(); //打印cipher
    void tostring(char *);//将newint转化成字符串格式
};
```

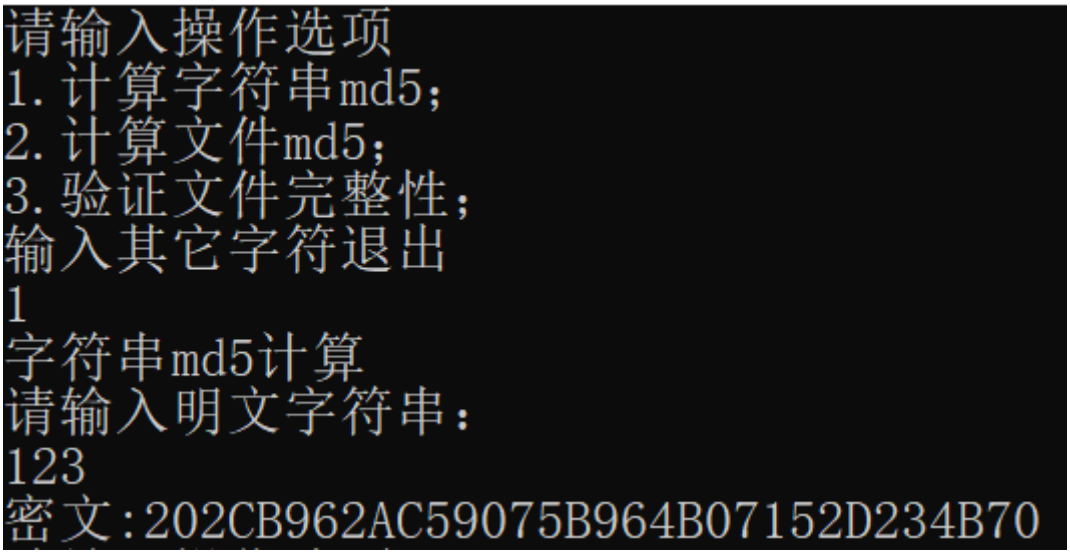
加密核心函数

```
void in(char* a, text& txt,int len)//输入明文
{
    memset(txt.m, 0, sizeof(txt.m));

    int index = 0;
    int i;
```

```
    i = len;
    int j;
    for (j = 0; j < i + 64; j++)
    {//处理文件和字符串稍有区别，这里用处理文件的方式进行演示
        //每512个字节一组
        txt.m[j / 64][(j%64)/4].a[j % 4] = a[j]&0xff;//&0xff解决符号扩展问题
    }

    txt.m[i / 64][(i % 64) / 4].a[i % 4] = 128;//补充内容最高位填充1
    i += 1;

    int t = i * 8 / 512;
    if (i * 8 % 512 > 448)
        t++;
    smallbit(len, txt.m[t][14]);//末64位以小端方式填入比特长度
    big(A, txt.fabcd[0][0]);//将A、B、C、D四个数以大端方式写入
    big(B, txt.fabcd[0][1]);
    big(C, txt.fabcd[0][2]);
    big(D, txt.fabcd[0][3]);
    txt.round = t;

}
```

字符串md5计算

选项参数为 `2`



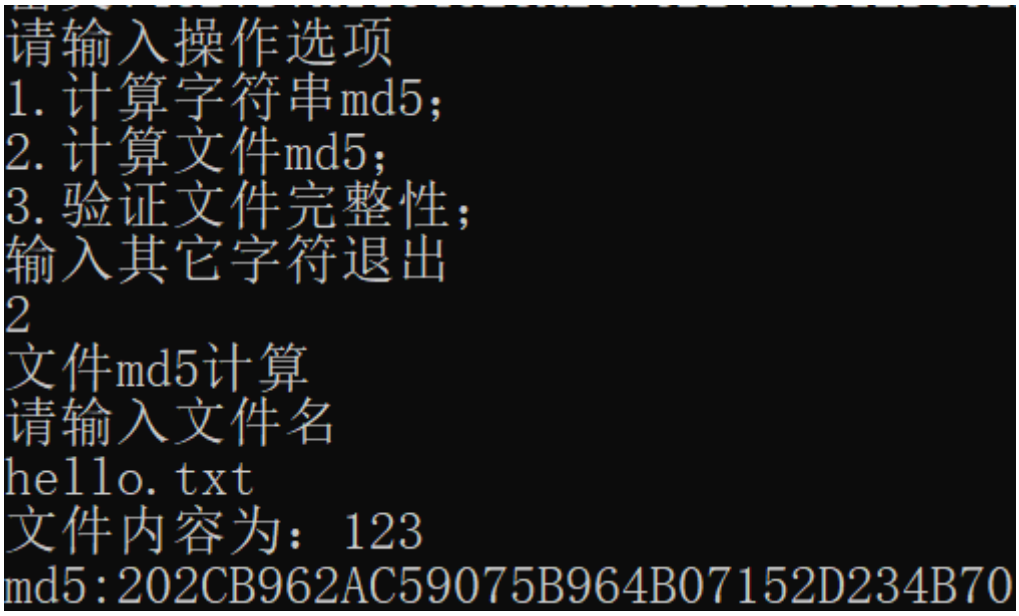
对字符串 `123` 进行md5计算

结果为 `202CB962AC59075B964B07152D234B70`

```
cout << "字符串md5计算" << endl;
char s[2048];
memset(s, 0, 2048);
cout << "请输入明文字符串：" << endl;
cin>>s;
text txt;
in(s, txt);//将字符串转化为对应格式
encrypt(txt);//加密计算
cout << "密文:";
txt.cipherprint();
```

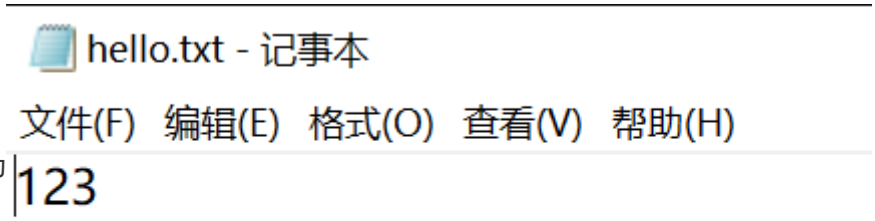
文件md5计算

选项参数为`2`



对名为`hello.txt`的文件进行计算

文件内容为



能够看到程序可以完整地识别文件内容，并进行计算，由于txt格式没有附加文件头等内容，所以记事本所写下的内容就是待计算的内容（123），可以看到，计算出的md5和上一部分直接对字符串`123`进行计算是一致的

使用命令行自带工具，验证了前面的假设

结合windows自带的md5计算工具进行查看

```
certutil -hashfile [filename] [option]
```

```
C:\Users\lxm>certutil -hashfile E:\study\大三下\网络安全技术\作业\MD5\MD5\hello.txt MD5
MD5 的 E:\study\大三下\网络安全技术\作业\MD5\MD5\hello.txt 哈希:
202cb962ac59075b964b07152d234b70
CertUtil: -hashfile 命令成功完成。
```

经过对比，发现一致

程序也实现了**对图片等其它文件**进行计算

例如计算该图片

```
C:\Users\lxm>xxd E:\study\大三下\网络安全技术\作业\MD5\MD5\4.jpg
00000000: ffd8 ffe0 0010 4a46 4946 0001 0101 0048  ......JFIF.....H
00000010: 0048 0000 ffe1 0022 4578 6966 0000 4d4d  .H....."Exif..MM
00000020: 002a 0000 0008 0001 0112 0003 0000 0001  .*..............
00000030: 0001 0000 0000 0000 ffdb 0043 0002 0101  ...........C....
00000040: 0201 0102 0202 0202 0202 0203 0503 0303  ................
00000050: 0303 0604 0403 0507 0607 0707 0607 0708  ................
00000060: 090b 0908 080a 0807 070a 0d0a 0a0b 0c0c  ................
00000070: 0c0c 0709 0e0f 0d0c 0e0b 0c0c 0cff db00  ................
00000080: 4301 0202 0203 0303 0603 0306 0c08 0708  C...............
00000090: 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c  ................
000000a0: 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c  ................
000000b0: 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c 0c0c  ................
000000c0: 0c0c ffc0 0011 0801 ce02 b203 0122 0002  ............."..
000000d0: 1101 0311 01ff c400 1f00 0001 0501 0101  ................
000000e0: 0101 0100 0000 0000 0000 0001 0203 0405  ................
000000f0: 0607 0809 0a0b ffc4 00b5 1000 0201 0303  ................
00000100: 0204 0305 0504 0400 0001 7d01 0203 0004  ..........}.....
00000110: 1105 1221 3141 0613 5161 0722 7114 3281  ...!1A..Qa."q.2.
```

.....

```
00006330: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006340: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006350: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006360: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006370: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006380: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006390: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
000063a0: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
000063b0: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
000063c0: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
000063d0: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
000063e0: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
000063f0: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006400: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006410: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006420: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006430: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006440: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006450: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006460: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006470: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006480: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
00006490: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
000064a0: 8a28 a002 8a28 a002 8a28 a002 8a28 a002  .(...(...(...(..
000064b0: 8a28 a002 8a28 a002 8a28 a00f ffd9        .(...(...(....
```

计算结果为

```
请输入文件名
4.jpg
文件内容为:    ?
md5:C6F74ECB5089B5787D96341E9C9840BB
```

结果一致

```
C:\Users\lxm>certutil -hashfile E:\study\大三下\网络安全技术\作业\MD5\MD5\4.jpg MD5
MD5 的 E:\study\大三下\网络安全技术\作业\MD5\MD5\4.jpg 哈希:
c6f74ecb5089b5787d96341e9c9840bb
CertUtil: -hashfile 命令成功完成。
```

```
cout << "文件md5计算\n请输入文件名" << endl;
char filename[20] = {};
cin >> filename;
char file[50000] = {};
int len = readfile(file, filename);//返回文件长度，直接使用strlen会被文件中\0字符截断
text txt;
in(file, txt, len);
encrypt(txt);
cout << "md5:";
txt.cipherprint();
```

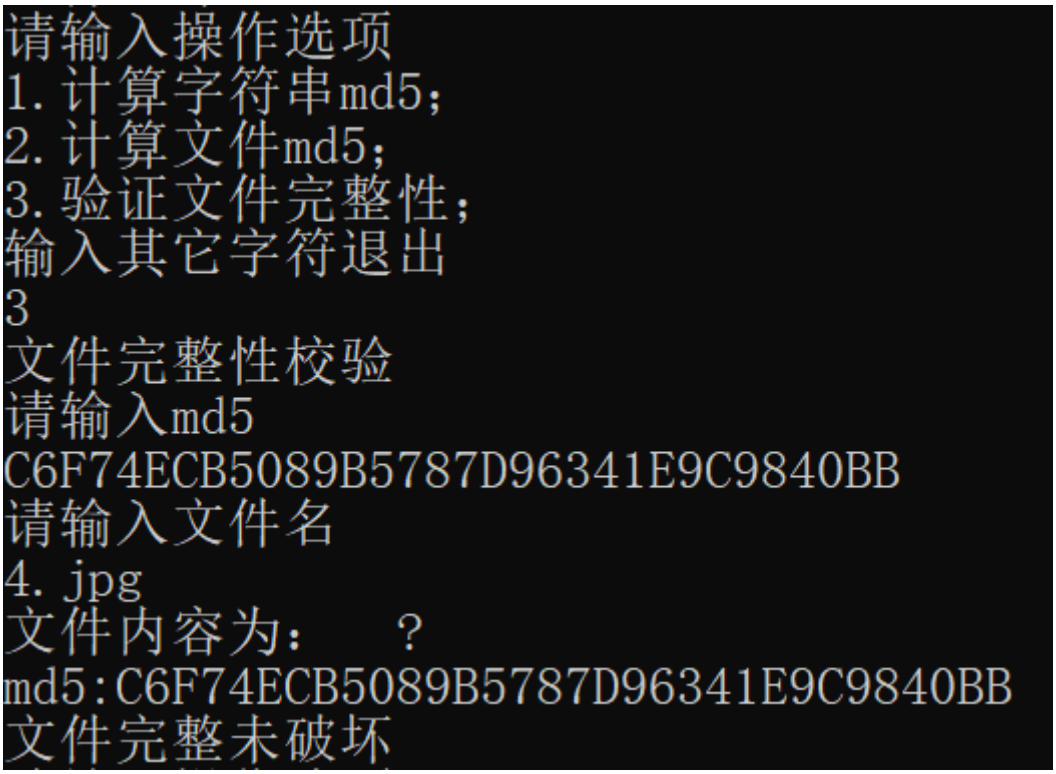
```
int readfile(char* file, char* filename)
{
```

```
    ifstream in(filename,ifstream::binary);//以二进制方式读入文件，以处理图片等其它形式文件
    if (!in)//文件打开失败
    {
        cout << "error" << endl;
    }
    int len = 0;
    char t = in.get();
    while (in) {//逐字节读入
        file[len++] = t;
        t = in.get();
    }
    cout <<"文件内容为："<< file << endl;
    in.close();//关闭文件句柄
    return len;
}
```

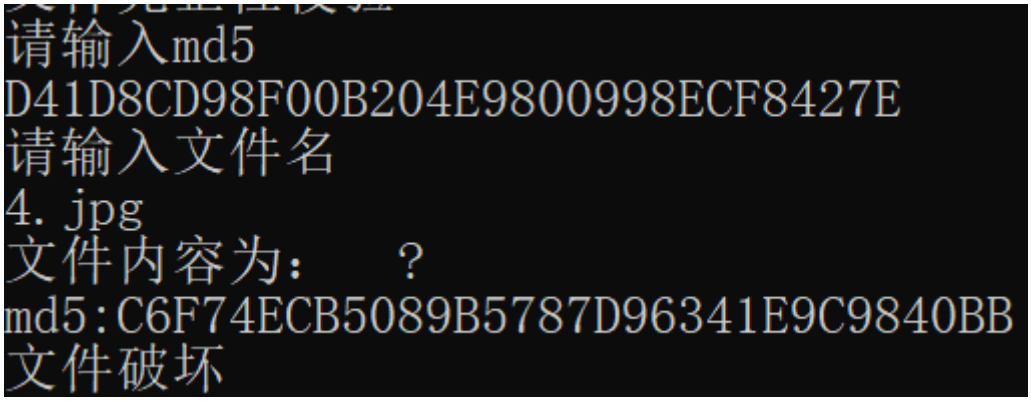
验证文件完整性

选项参数`3`

若输入md5与计算一致：



若不一致



计算过程与上一部分一致，只是添加了一个newint转字符串的函数，方便与输入的md5进行比较

```
void text::tostring(char* cipher)
{
    int len = 0;
    for (int i = 0; i < 4; i++)
    {
        char str[5] = {};
        for (int j = 0; j < 4; j++)
        {
            sprintf(str, "%02X", fabcd[0][i].a[j]);//使用sprintf，将打印样式存入str中
            int s = strlen(str);
            for (int k = 0; k < s; k++)
            {
                cipher[len++] = str[k];
            }
        }
    }
}
```