Computer Science and Engineering Department, SVNIT, Surat
Information Security & Cryptography
ASSIGNMENT- 6

U20CS005
BANSI MARAKANA

Write a program to calculate the message digest of a text using the MD5 algorithm.

```
#include <bits/stdc++.h>
using namespace std;
#define MD5_INPUT_LENGTH 64
typedef unsigned char uchar_8;
typedef unsigned int uint_32;
typedef unsigned long uint_64;

static const uint_32 T[64] = {
    0xd76aa478, 0xe8c7b756, 0x242070db, 0xc1bdceee,
    0xf57c0faf, 0x4787c62a, 0xa8304613, 0xfd469501,
    0x698098d8, 0x8b44f7af, 0xffff5bb1, 0x895cd7be,
    0x6b901122, 0xfd987193, 0xa679438e, 0x49b40821,
    0xf61e2562, 0xc040b340, 0x265e5a51, 0xe9b6c7aa,
    0xd62f105d, 0x02441453, 0xd8a1e681, 0xe7d3fbc8,
    0x21e1cde6, 0xc33707d6, 0xf4d50d87, 0x455a14ed,
    0xa9e3e905, 0xfcefa3f8, 0x676f02d9, 0x8d2a4c8a,
    0xfffa3942, 0x8771f681, 0x6d9d6122, 0xfde5380c,
    0xa4beea44, 0x4bdecfa9, 0xf6bb4b60, 0xbebfbc70,
    0x289b7ec6, 0xeaa127fa, 0xd4ef3085, 0x04881d05,
    0xd9d4d039, 0xe6db99e5, 0x1fa27cf8, 0xc4ac5665,
    0xf4292244, 0x432aff97, 0xab9423a7, 0xfc93a039,
    0x655b59c3, 0x8f0ccc92, 0xffeff47d, 0x85845dd1,
    0x6fa87e4f, 0xfe2ce6e0, 0xa3014314, 0x4e0811a1,
    0xf7537e82, 0xbd3af235, 0x2ad7d2bb, 0xeb86d391};

static const uint_32 A0 = 0x01234567;
static const uint_32 B0 = 0x89abcdef;
static const uint_32 C0 = 0xfedcba98;
static const uint_32 D0 = 0x76543210;

uint_32 F(uint_32 x, uint_32 y, uint_32 z)
{
    return ((x & y) | (~x & z));
```

```
}

uint_32 G(uint_32 x, uint_32 y, uint_32 z)
{
    return ((x & z) | (y & ~z));
}

uint_32 H(uint_32 x, uint_32 y, uint_32 z)
{
    return (x ^ y ^ z);
}

uint_32 I(uint_32 x, uint_32 y, uint_32 z)
{
    return (y ^ (x & ~z));
}

uint_32 shift(uint_32 x, uint_32 s)
{
    return ((x << s) | (x >> (32 - s)));
}

void FF(uint_32 &a, uint_32 b, uint_32 c, uint_32 d, uint_32 x, uint_32 s,
uint_32 t)
{
    a += F(b, c, d) + x + t;
    a = shift(a, s) + b;
}

void GG(uint_32 &a, uint_32 b, uint_32 c, uint_32 d, uint_32 x, uint_32 s,
uint_32 t)
{
    a += G(b, c, d) + x + t;
    a = shift(a, s) + b;
}

void HH(uint_32 &a, uint_32 b, uint_32 c, uint_32 d, uint_32 x, uint_32 s,
uint_32 t)
{
    a += H(b, c, d) + x + t;
```

```
    a = shift(a, s) + b;
}

void II(uint_32 &a, uint_32 b, uint_32 c, uint_32 d, uint_32 x, uint_32 s,
uint_32 t)
{
    a += I(b, c, d) + x + t;
    a = shift(a, s) + b;
}

uint_32 *md5Pad(char *charBuf, uint_64 len)
{
    uint_64 newLen;
    for (newLen = len * 8 + 1; newLen % 512 != 448; newLen++);
    newLen /= 8;
    uint_32 *buf = new uint_32[newLen + 64];
    memset(buf, 0, newLen + 64);
    memcpy(buf, charBuf, len);
    buf[len] = 0x80;
    uint_32 bitsLen = len * 8;
    memcpy(buf + newLen, &bitsLen, 4);
    return buf;
}

void MD5(uint_32 *outBuf, uint_32 *inBuf)
{
    uint_32 a, b, c, d;
    a = A0;
    b = B0;
    c = C0;
    d = D0;

    // Shift amounts 1st round
    static const uchar_8 S11 = 7, S12 = 12, S13 = 17, S14 = 22;
    FF(a, b, c, d, inBuf[0], S11, T[0]); /* 1 */
    FF(d, a, b, c, inBuf[1], S12, T[1]); /* 2 */
    FF(c, d, a, b, inBuf[2], S13, T[2]); /* 3 */
    FF(b, c, d, a, inBuf[3], S14, T[3]); /* 4 */
    FF(a, b, c, d, inBuf[4], S11, T[4]); /* 5 */
    FF(d, a, b, c, inBuf[5], S12, T[5]); /* 6 */
```

```
    FF(c, d, a, b, inBuf[6], S13, T[6]); /* 7 */
    FF(b, c, d, a, inBuf[7], S14, T[7]); /* 8 */
    FF(a, b, c, d, inBuf[8], S11, T[8]);   /* 9 */
    FF(d, a, b, c, inBuf[9], S12, T[9]);   /* 10 */
    FF(c, d, a, b, inBuf[10], S13, T[10]); /* 11 */
    FF(b, c, d, a, inBuf[11], S14, T[1]);  /* 12 */
    FF(a, b, c, d, inBuf[12], S11, T[12]); /* 13 */
    FF(d, a, b, c, inBuf[13], S12, T[13]); /* 14 */
    FF(c, d, a, b, inBuf[14], S13, T[14]); /* 15 */
    FF(b, c, d, a, inBuf[15], S14, T[15]); /* 16 */

    // Shift amounts 2nd round
    static const uchar_8 S21 = 5, S22 = 9, S23 = 14, S24 = 20;
    GG(a, b, c, d, inBuf[1], S21, T[16]);  /* 17 */
    GG(d, a, b, c, inBuf[6], S22, T[17]);  /* 18 */
    GG(c, d, a, b, inBuf[11], S23, T[18]); /* 19 */
    GG(b, c, d, a, inBuf[0], S24, T[19]);  /* 20 */
    GG(a, b, c, d, inBuf[5], S21, T[20]);  /* 21 */
    GG(d, a, b, c, inBuf[10], S22, T[21]); /* 22 */
    GG(c, d, a, b, inBuf[15], S23, T[22]); /* 23 */
    GG(b, c, d, a, inBuf[4], S24, T[23]);  /* 24 */
    GG(a, b, c, d, inBuf[9], S21, T[24]);  /* 25 */
    GG(d, a, b, c, inBuf[14], S22, T[25]); /* 26 */
    GG(c, d, a, b, inBuf[3], S23, T[26]);  /* 27 */
    GG(b, c, d, a, inBuf[8], S24, T[27]);  /* 28 */
    GG(a, b, c, d, inBuf[13], S21, T[28]); /* 29 */
    GG(d, a, b, c, inBuf[2], S22, T[29]);  /* 30 */
    GG(c, d, a, b, inBuf[7], S23, T[30]);  /* 31 */
    GG(b, c, d, a, inBuf[12], S24, T[31]); /* 32 */

    // Shift amounts 3rd round
    static const uchar_8 S31 = 4, S32 = 11, S33 = 16, S34 = 23;
    HH(a, b, c, d, inBuf[5], S31, T[32]);  /* 33 */
    HH(d, a, b, c, inBuf[8], S32, T[33]);  /* 34 */
    HH(c, d, a, b, inBuf[11], S33, T[34]); /* 35 */
    HH(b, c, d, a, inBuf[14], S34, T[35]); /* 36 */
    HH(a, b, c, d, inBuf[1], S31, T[36]);  /* 37 */
    HH(d, a, b, c, inBuf[4], S32, T[37]);  /* 38 */
    HH(c, d, a, b, inBuf[7], S33, T[38]);  /* 39 */
    HH(b, c, d, a, inBuf[10], S34, T[39]); /* 40 */
```

```
    HH(a, b, c, d, inBuf[13], S31, T[40]); /* 41 */
    HH(d, a, b, c, inBuf[0], S32, T[41]);  /* 42 */
    HH(c, d, a, b, inBuf[3], S33, T[42]);  /* 43 */
    HH(b, c, d, a, inBuf[6], S34, T[43]);  /* 44 */
    HH(a, b, c, d, inBuf[9], S31, T[44]);  /* 45 */
    HH(d, a, b, c, inBuf[12], S32, T[45]); /* 46 */
    HH(c, d, a, b, inBuf[15], S33, T[46]); /* 47 */
    HH(b, c, d, a, inBuf[2], S34, T[47]);  /* 48 */

    // Shift amounts 4th round
    static const uchar_8 S41 = 6, S42 = 10, S43 = 15, S44 = 21;
    II(a, b, c, d, inBuf[0], S41, T[48]);  /* 49 */
    II(d, a, b, c, inBuf[7], S42, T[49]);  /* 50 */
    II(c, d, a, b, inBuf[14], S43, T[50]); /* 51 */
    II(b, c, d, a, inBuf[5], S44, T[51]);  /* 52 */
    II(a, b, c, d, inBuf[12], S41, T[52]); /* 53 */
    II(d, a, b, c, inBuf[3], S42, T[53]);  /* 54 */
    II(c, d, a, b, inBuf[10], S43, T[54]); /* 55 */
    II(b, c, d, a, inBuf[1], S44, T[55]);  /* 56 */
    II(a, b, c, d, inBuf[8], S41, T[56]);  /* 57 */
    II(d, a, b, c, inBuf[15], S42, T[57]); /* 58 */
    II(c, d, a, b, inBuf[6], S43, T[58]);  /* 59 */
    II(b, c, d, a, inBuf[13], S44, T[59]); /* 60 */
    II(a, b, c, d, inBuf[4], S41, T[60]);  /* 61 */
    II(d, a, b, c, inBuf[11], S42, T[61]); /* 62 */
    II(c, d, a, b, inBuf[2], S43, T[62]);  /* 63 */
    II(b, c, d, a, inBuf[9], S44, T[63]);  /* 64 */

    a += A0;
    b += B0;
    c += C0;
    d += D0;
    outBuf[0] = a;
    outBuf[1] = b;
    outBuf[2] = c;
    outBuf[3] = d;
    return;
}

int main()
```

```
{
    char *srcStr = "I am learning Cryptography";
    uint_32 *padded = md5Pad(srcStr, strlen(srcStr));
    uint_32 result[4] = {0};
    MD5(result, padded);
    uchar_8 digestChars[16] = {0};
    memcpy(digestChars, result, 16);
    printf("Message Digest: %2.2x%2.2x%2.2x%2.2x\n", result[0], result[1],
result[2], result[3]);
    delete[] padded;
    return 0;
}
```

```
PS D:\BANSI MARAKANA\ISC> ./a
Message Digest: af931c09ae607585c22e942161748b09
```