ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

THÁI NGỌC DIỄM TRINH – 22521541

CLASS: NT219.O22.ANTT

OFF-CLASS LABS

LAB 4: PKI AND HASH FUNCTIONS

TP. HỒ CHÍ MINH, NĂM 2024

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

THÁI NGỌC DIỄM TRINH - 22521541

OFF-CLASS LABS

LAB 6: COLLISION AND LENGTH EXTENSION ATTACKS ON HASH FUNCTIONS

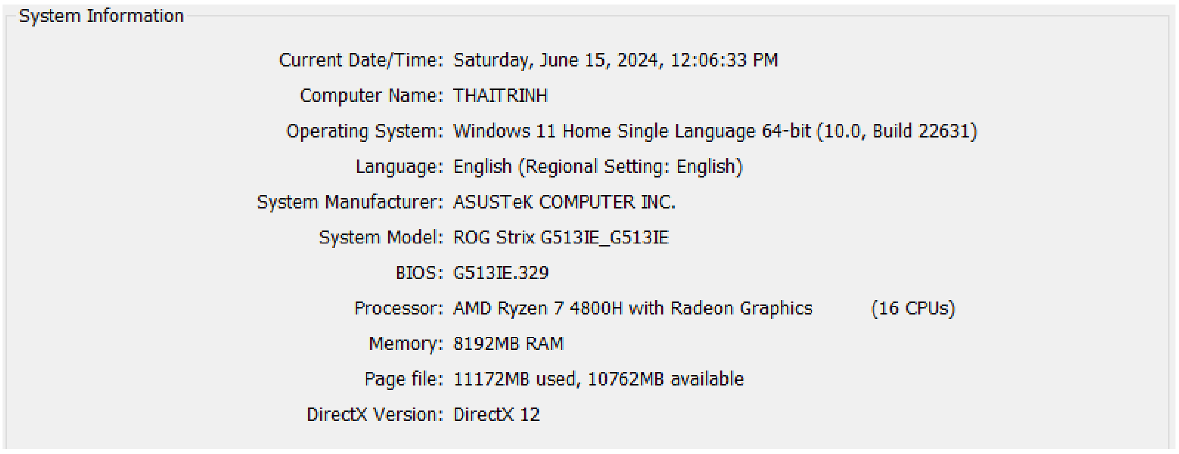
GIẢNG VIÊN HƯỚNG DẪN

TS. NGUYỄN NGỌC TỰ

TP. HỒ CHÍ MINH, NĂM 2024

MỤC LỤC

Chương 1. **HARDWARE RESOURCE**



System Information

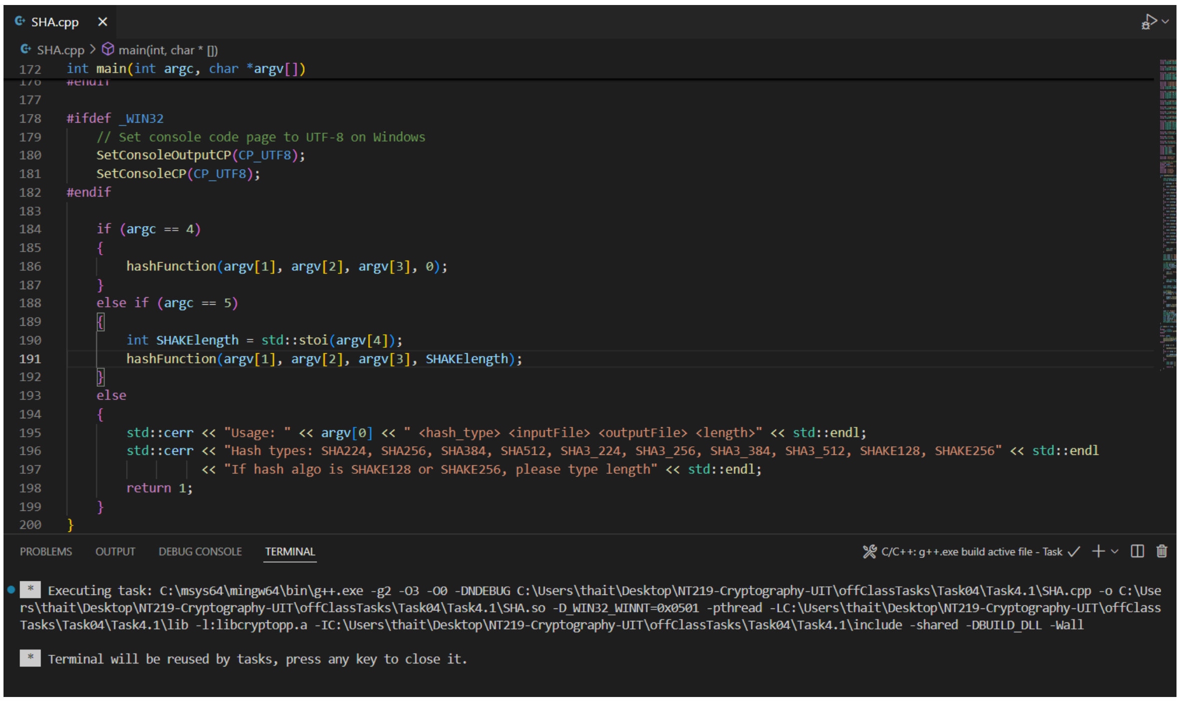
Current Date/Time:	Saturday, June 15, 2024, 12:06:33 PM
Computer Name:	THAITRINH
Operating System:	Windows 11 Home Single Language 64-bit (10.0, Build 22631)
Language:	English (Regional Setting: English)
System Manufacturer:	ASUSTeK COMPUTER INC.
System Model:	ROG Strix G513IE_G513IE
BIOS:	G513IE.329
Processor:	AMD Ryzen 7 4800H with Radeon Graphics (16 CPUs)
Memory:	8192MB RAM
Page file:	11172MB used, 10762MB available
DirectX Version:	DirectX 12

Chương 2. HASH FUNCTIONS

2.1. Build Tasks

2.1.1. Command Line:

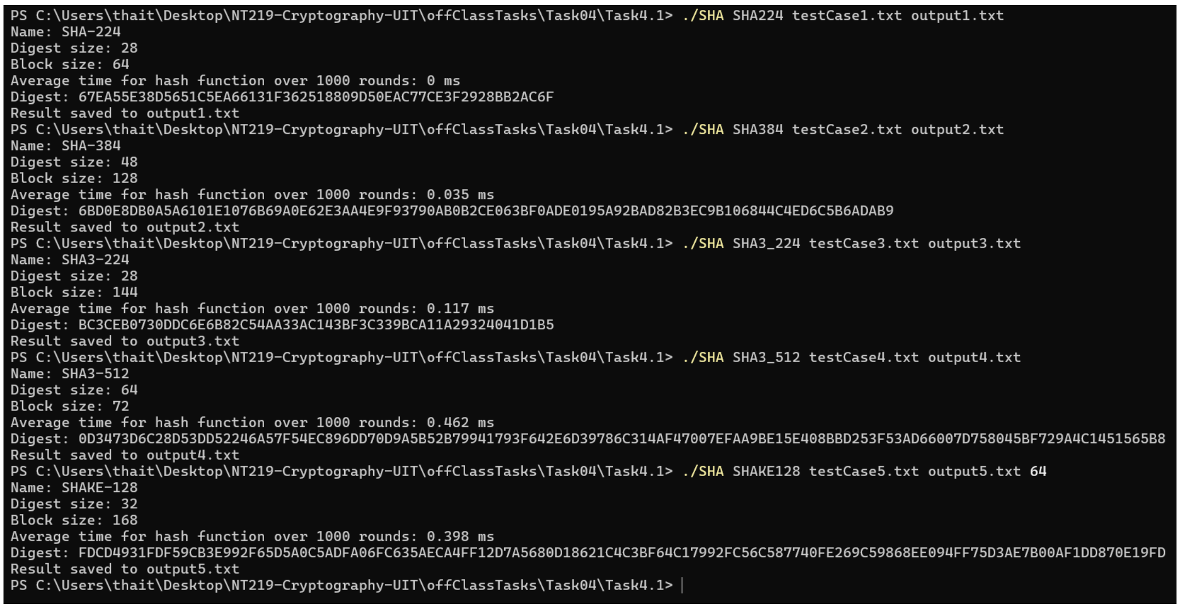
- Windows:



SHA.cpp × | SHA.cpp > main(int, char * [])

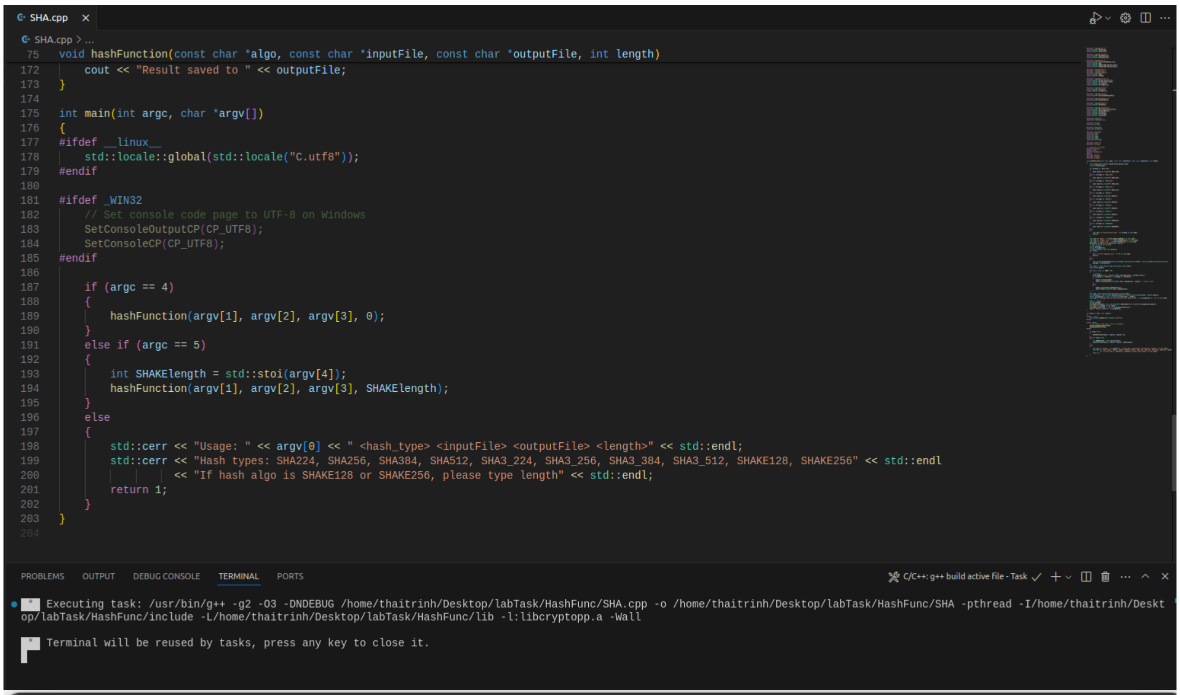
```cpp
int main(int argc, char *argv[])
#endif

#ifdef _WIN32
    // Set console code page to UTF-8 on Windows
    SetConsoleOutputCP(CP_UTF8);
    SetConsoleCP(CP_UTF8);
#endif

    if (argc == 4)
    {
        hashFunction(argv[1], argv[2], argv[3], 0);
    }
    else if (argc == 5)
    {
        int SHAKElength = std::stoi(argv[4]);
        hashFunction(argv[1], argv[2], argv[3], SHAKElength);
    }
    else
    {
        std::cerr << "Usage: " << argv[0] << " <hash_type> <inputFile> <outputFile> <length>" << std::endl;
        std::cerr << "Hash types: SHA224, SHA256, SHA384, SHA512, SHA3_224, SHA3_256, SHA3_384, SHA3_512, SHAKE128, SHAKE256" << std::endl
                  << "If hash algo is SHAKE128 or SHAKE256, please type length" << std::endl;
        return 1;
    }
}
```

```
 *  Executing task: C:\msys64\mingw64\bin\g++.exe -g2 -O3 -O0 -DNDEBUG C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1\SHA.cpp -o C:\Use
rs\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1\SHA.so -D_WIN32_WINNT=0x0501 -pthread -LC:\Users\thait\Desktop\NT219-Cryptography-UIT\offClass
Tasks\Task04\Task4.1\lib -l:libcryptopp.a -IC:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1\include -shared -DBUILD_DLL -Wall

 *  Terminal will be reused by tasks, press any key to close it.
```



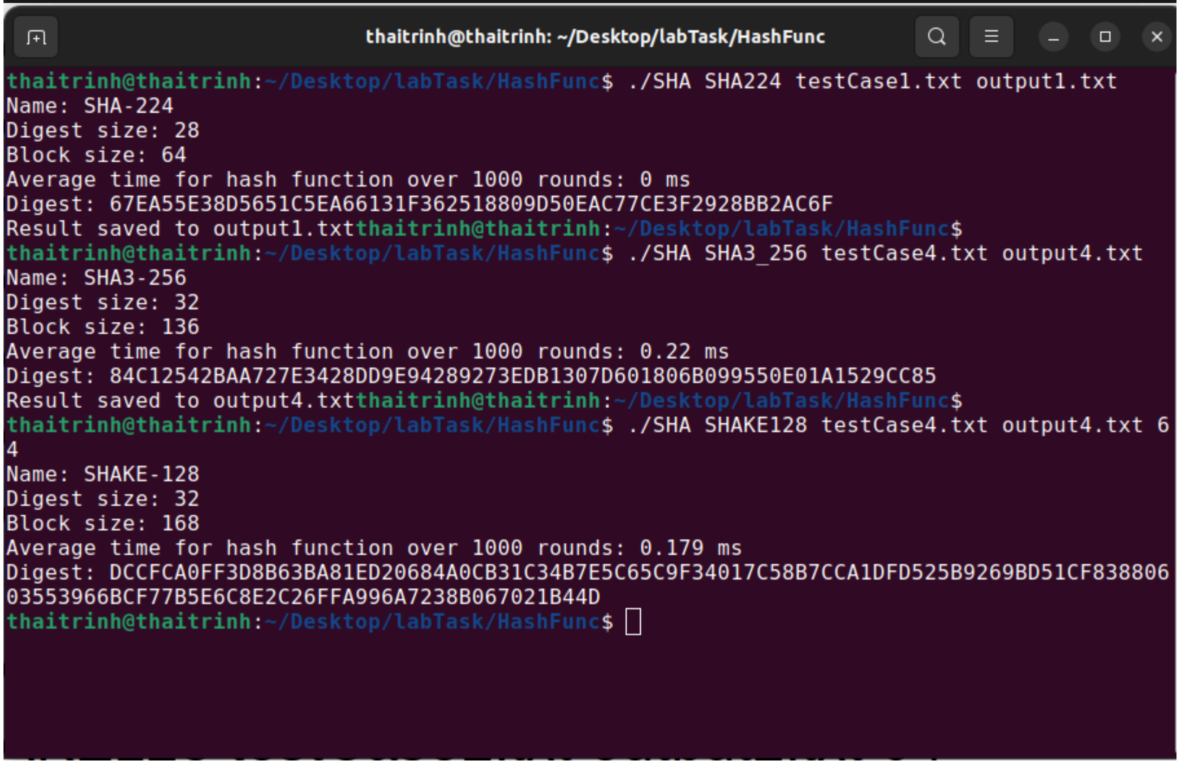
```
PS C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1> ./SHA SHA224 testCase1.txt output1.txt
Name: SHA-224
Digest size: 28
Block size: 64
Average time for hash function over 1000 rounds: 0 ms
Digest: 67EA55E38D5651C5EA66131F362518809D50EAC77CE3F2928BB2AC6F
Result saved to output1.txt
PS C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1> ./SHA SHA384 testCase2.txt output2.txt
Name: SHA-384
Digest size: 48
Block size: 128
Average time for hash function over 1000 rounds: 0.035 ms
Digest: 6BD0E8DB0A5A6101E1076B69A0E62E3AA4E9F93790AB0B2CE063BF0ADE0195A92BAD82B3EC9B106844C4ED6C5B6ADAB9
Result saved to output2.txt
PS C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1> ./SHA SHA3_224 testCase3.txt output3.txt
Name: SHA3-224
Digest size: 28
Block size: 144
Average time for hash function over 1000 rounds: 0.117 ms
Digest: BC3CEB0730DDC6E6B82C54AA33AC143BF3C339BCA11A29324041D1B5
Result saved to output3.txt
PS C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1> ./SHA SHA3_512 testCase4.txt output4.txt
Name: SHA3-512
Digest size: 64
Block size: 72
Average time for hash function over 1000 rounds: 0.462 ms
Digest: 0D3473D6C28D53DD52246A57F54EC896DD70D9A5B52B79941793F642E6D39786C314AF47007EFAA9BE15E408BBD253F53AD66007D758045BF729A4C1451565B8
Result saved to output4.txt
PS C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1> ./SHA SHAKE128 testCase5.txt output5.txt 64
Name: SHAKE-128
Digest size: 32
Block size: 168
Average time for hash function over 1000 rounds: 0.398 ms
Digest: FDCD4931FDF59CB3E992F65D5A0C5ADFA06FC635AECA4FF12D7A5680D18621C4C3BF64C17992FC56C587740FE269C59868EE094FF75D3AE7B00AF1DD870E19FD
Result saved to output5.txt
PS C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.1> |
```

- Ubuntu:



```cpp
void hashFunction(const char *algo, const char *inputFile, const char *outputFile, int length)
    cout << "Result saved to " << outputFile;
}

int main(int argc, char *argv[])
{
#ifdef __linux__
    std::locale::global(std::locale("C.utf8"));
#endif

#ifdef _WIN32
    // Set console code page to UTF-8 on Windows
    SetConsoleOutputCP(CP_UTF8);
    SetConsoleCP(CP_UTF8);
#endif

    if (argc == 4)
    {
        hashFunction(argv[1], argv[2], argv[3], 0);
    }
    else if (argc == 5)
    {
        int SHAKElength = std::stoi(argv[4]);
        hashFunction(argv[1], argv[2], argv[3], SHAKElength);
    }
    else
    {
        std::cerr << "Usage: " << argv[0] << " <hash_type> <inputFile> <outputFile> <length>" << std::endl;
        std::cerr << "Hash types: SHA224, SHA256, SHA384, SHA512, SHA3_224, SHA3_256, SHA3_384, SHA3_512, SHAKE128, SHAKE256" << std::endl
                  << "If hash algo is SHAKE128 or SHAKE256, please type length" << std::endl;
        return 1;
    }
}
```

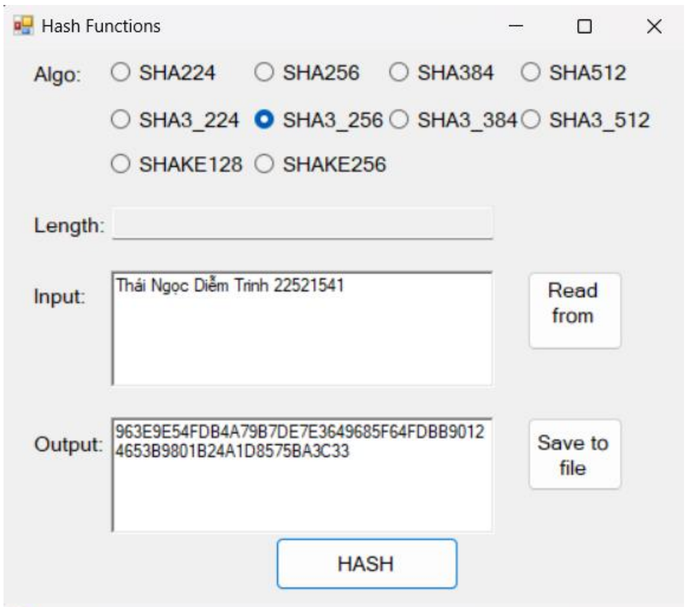
```
 *  Executing task: /usr/bin/g++ -g2 -O3 -DNDEBUG /home/thaitrinh/Desktop/labTask/HashFunc/SHA.cpp -o /home/thaitrinh/Desktop/labTask/HashFunc/SHA -pthread -I/home/thaitrinh/Deskt
op/labTask/HashFunc/include -L/home/thaitrinh/Desktop/labTask/HashFunc/lib -l:libcryptopp.a -Wall

 *  Terminal will be reused by tasks, press any key to close it.
```



thaitrinh@thaitrinh: ~/Desktop/labTask/HashFunc

```
thaitrinh@thaitrinh:~/Desktop/labTask/HashFunc$ ./SHA SHA224 testCase1.txt output1.txt
Name: SHA-224
Digest size: 28
Block size: 64
Average time for hash function over 1000 rounds: 0 ms
Digest: 67EA55E38D5651C5EA66131F362518809D50EAC77CE3F2928BB2AC6F
Result saved to output1.txtthaitrinh@thaitrinh:~/Desktop/labTask/HashFunc$
thaitrinh@thaitrinh:~/Desktop/labTask/HashFunc$ ./SHA SHA3_256 testCase4.txt output4.txt
Name: SHA3-256
Digest size: 32
Block size: 136
Average time for hash function over 1000 rounds: 0.22 ms
Digest: 84C12542BAA727E3428DD9E94289273EDB1307D601806B099550E01A1529CC85
Result saved to output4.txtthaitrinh@thaitrinh:~/Desktop/labTask/HashFunc$
thaitrinh@thaitrinh:~/Desktop/labTask/HashFunc$ ./SHA SHAKE128 testCase4.txt output4.txt 6
4
Name: SHAKE-128
Digest size: 32
Block size: 168
Average time for hash function over 1000 rounds: 0.179 ms
Digest: DCCFCA0FF3D8B63BA81ED20684A0CB31C34B7E5C65C9F34017C58B7CCA1DFD525B9269BD51CF838806
03553966BCF77B5E6C8E2C26FFA996A7238B067021B44D
thaitrinh@thaitrinh:~/Desktop/labTask/HashFunc$ 
```

2.1.2. GUI

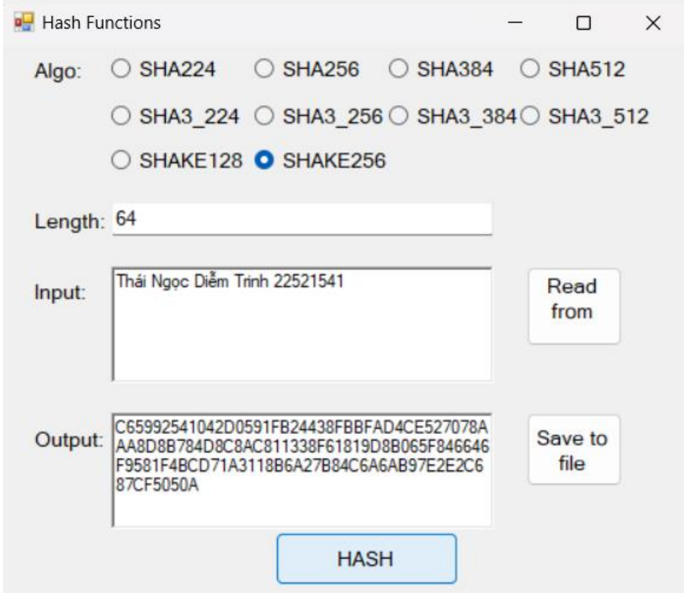


Algo: SHA224 SHA256 SHA384 SHA512 SHA3_224 [✓ SHA3_256] SHA3_384 SHA3_512 SHAKE128 SHAKE256

Length:

Input: Thái Ngọc Diễm Trinh 22521541

Output:
```
963E9E54FDB4A79B7DE7E3649685F64FDBB9012
4653B9801B24A1D8575BA3C33
```



Algo: SHA224 SHA256 SHA384 SHA512 SHA3_224 SHA3_256 SHA3_384 SHA3_512 SHAKE128 [✓ SHAKE256]

Length: 64

Input: Thái Ngọc Diễm Trinh 22521541

Output:
```
C65992541042D0591FB24438FBBFAD4CE527078A
AA8D8B784D8C8AC811338F61819D8B065F846646
F9581F4BCD71A3118B6A27B84C6A6AB97E2E2C6
87CF5050A
```

2.2. Computation Performance

Number of iterations: 1000

Time counter: mili seconds

2.2.1. SHA224

Test case	Windows	Linux
1 (1KB)	0	0
2 (20KB)	0.009	0.009
3 (50KB)	0.024	0.024
4 (100KB)	0.048	0.048
5 (200KB)	0.096	0.097
6 (1MB)	0.564	0.565
7 (5MB)	2.465	2.484

2.2.2. SHA256

Test case	Windows	Linux
1 (1KB)	0	0
2 (20KB)	0.009	0.009
3 (50KB)	0.024	0.024
4 (100KB)	0.048	0.049
5 (200KB)	0.096	0.097
6 (1MB)	0.564	0.575
7 (5MB)	2.613	2.495

2.2.3. SHA384

Test case	Windows	Linux
1 (1KB)	0.001	0.001
2 (20KB)	0.035	0.031
3 (50KB)	0.092	0.084
4 (100KB)	0.185	0.165
5 (200KB)	0.37	0.33
6 (1MB)	2.171	1.958
7 (5MB)	9.563	8.435

2.2.4. SHA512

Test case	Windows	Linux
1 (1KB)	0.001	0.001
2 (20KB)	0.035	0.031
3 (50KB)	0.092	0.084
4 (100KB)	0.184	0.166
5 (200KB)	0.37	0.329
6 (1MB)	2.178	1.975
7 (5MB)	9.378	8.384

2.2.5. SHA3-224

Test case	Windows	Linux
1 (1KB)	0.002	0.001
2 (20KB)	0.044	0.04
3 (50KB)	0.119	0.111
4 (100KB)	0.239	0.211
5 (200KB)	0.487	0.427
6 (1MB)	2.822	2.442
7 (5MB)	11.77	10.539

2.2.6. SHA3-256

Test case	Windows	Linux
1 (1KB)	0.002	0.001
2 (20KB)	0.047	0.043
3 (50KB)	0.122	0.113
4 (100KB)	0.246	00.221
5 (200KB)	0.491	0.445
6 (1MB)	2.891	2.586
7 (5MB)	12.426	11.154

2.2.7. SHA3-384

Test case	Windows	Linux
1 (1KB)	0.002	0.002
2 (20KB)	0.061	0.056
3 (50KB)	0.161	0.145
4 (100KB)	0.319	0.288
5 (200KB)	0.644	0.578
6 (1MB)	3.768	3.378
7 (5MB)	16.23	14.556

2.2.8. SHA3-512

Test case	Windows	Linux
1 (1KB)	0.003	0.003
2 (20KB)	0.088	0.081
3 (50KB)	0.231	0.209
4 (100KB)	0.467	0.421
5 (200KB)	0.926	0.844
6 (1MB)	5.433	4.886
7 (5MB)	23.48	21.042

2.2.9. SHAKE128 (digest size: 64)

Test case	Windows	Linux
1 (1KB)	0.001	0.001
2 (20KB)	0.038	0.034
3 (50KB)	0.099	0.092
4 (100KB)	0.199	0.181
5 (200KB)	0.398	0.361
6 (1MB)	2.331	2.103
7 (5MB)	10.241	9.072

2.2.10. SHAKE256 (digest size: 64)

Test case	Windows	Linux
1 (1KB)	0.002	0.001
2 (20KB)	0.047	0.042
3 (50KB)	0.122	0.113
4 (100KB)	0.25	0.222
5 (200KB)	0.5	0.445
6 (1MB)	2.889	2.593
7 (5MB)	12.614	11.199

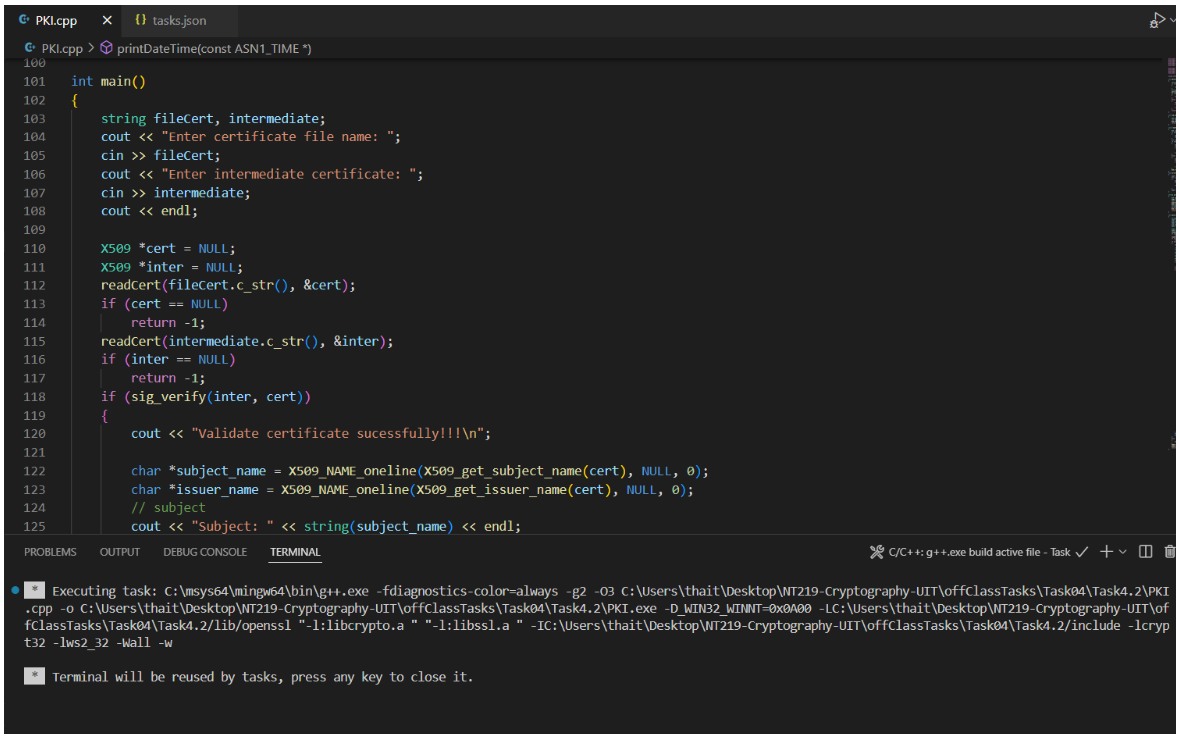
2.3. Comments:

For most algorithms and file sizes, running times on Windows and Ubuntu are comparable, but Ubuntu is slightly faster.

With larger files, the average time will be higher.

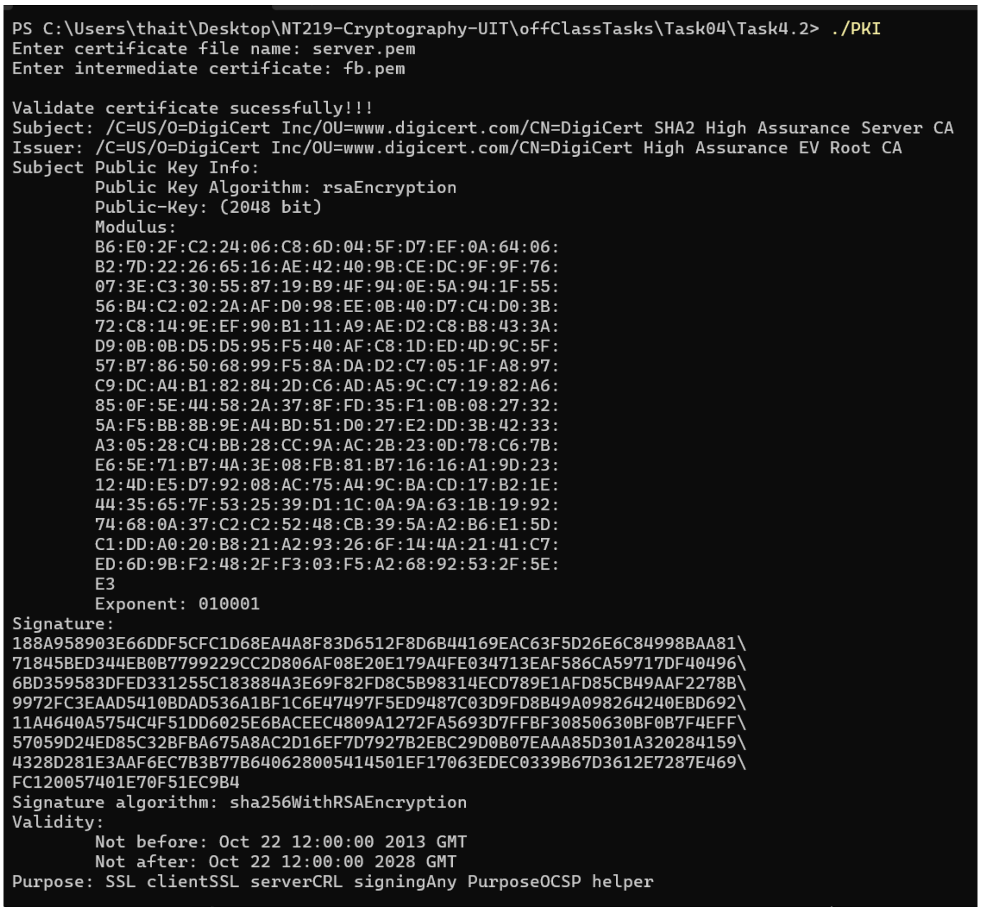
SHA2 algorithms (SHA224, SHA256, SHA384, SHA512) usually have lower running times than SHA3 and SHAKE algorithms with the same file size.

Within the same algorithm, variants with larger digest sizes often take longer (e.g., SHA256 vs. SHA224, SHA3-512 vs. SHA3-224).



```
int main()
{
    string fileCert, intermediate;
    cout << "Enter certificate file name: ";
    cin >> fileCert;
    cout << "Enter intermediate certificate: ";
    cin >> intermediate;
    cout << endl;

    X509 *cert = NULL;
    X509 *inter = NULL;
    readCert(fileCert.c_str(), &cert);
    if (cert == NULL)
        return -1;
    readCert(intermediate.c_str(), &inter);
    if (inter == NULL)
        return -1;
    if (sig_verify(inter, cert))
    {
        cout << "Validate certificate sucessfully!!!\n";

        char *subject_name = X509_NAME_oneline(X509_get_subject_name(cert), NULL, 0);
        char *issuer_name = X509_NAME_oneline(X509_get_issuer_name(cert), NULL, 0);
        // subject
        cout << "Subject: " << string(subject_name) << endl;
```

```
* Executing task: C:\msys64\mingw64\bin\g++.exe -fdiagnostics-color=always -g2 -O3 C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.2\PKI
.cpp -o C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.2\PKI.exe -D_WIN32_WINNT=0x0A00 -LC:\Users\thait\Desktop\NT219-Cryptography-UIT\of
fClassTasks\Task04\Task4.2/lib/openssl "-l:libcrypto.a " "-l:libssl.a " -IC:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.2/include -lcryp
t32 -lws2_32 -Wall -w

* Terminal will be reused by tasks, press any key to close it.
```



```
PS C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.2> ./PKI
Enter certificate file name: server.pem
Enter intermediate certificate: fb.pem

Validate certificate sucessfully!!!
Subject: /C=US/O=DigiCert Inc/OU=www.digicert.com/CN=DigiCert SHA2 High Assurance Server CA
Issuer: /C=US/O=DigiCert Inc/OU=www.digicert.com/CN=DigiCert High Assurance EV Root CA
Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
        Public-Key: (2048 bit)
        Modulus:
        B6:E0:2F:C2:24:06:C8:6D:04:5F:D7:EF:0A:64:06:
        B2:7D:22:26:65:16:AE:42:40:9B:CE:DC:9F:9F:76:
        07:3E:C3:30:55:87:19:B9:4F:94:0E:5A:94:1F:55:
        56:B4:C2:02:2A:AF:D0:98:EE:0B:40:D7:C4:D0:3B:
        72:C8:14:9E:EF:90:B1:11:A9:AE:D2:C8:B8:43:3A:
        D9:0B:0B:D5:D5:95:F5:40:AF:C8:1D:ED:4D:9C:5F:
        57:B7:86:50:68:99:F5:8A:DA:D2:C7:05:1F:A8:97:
        C9:DC:A4:B1:82:84:2D:C6:AD:A5:9C:C7:19:82:A6:
        85:0F:5E:44:58:2A:37:8F:FD:35:F1:0B:08:27:32:
        5A:F5:BB:8B:9E:A4:BD:51:D0:27:E2:DD:3B:42:33:
        A3:05:28:C4:BB:28:CC:9A:AC:2B:23:0D:78:C6:7B:
        E6:5E:71:B7:4A:3E:08:FB:81:B7:16:16:A1:9D:23:
        12:4D:E5:D7:92:08:AC:75:A4:9C:BA:CD:17:B2:1E:
        44:35:65:7F:53:25:39:D1:1C:0A:9A:63:1B:19:92:
        74:68:0A:37:C2:C2:52:48:CB:39:5A:A2:B6:E1:5D:
        C1:DD:A0:20:B8:21:A2:93:26:6F:14:4A:21:41:C7:
        ED:6D:9B:F2:48:2F:F3:03:F5:A2:68:92:53:2F:5E:
        E3
        Exponent: 010001
Signature:
188A958903E66DDF5CFC1D68EA4A8F83D6512F8D6B44169EAC63F5D26E6C84998BAA81\
71845BED344EB0B7799229CC2D806AF08E20E179A4FE034713EAF586CA59717DF40496\
6BD359583DFED331255C183884A3E69F82FD8C5B98314ECD789E1AFD85CB49AAF2278B\
9972FC3EAAD5410BDAD536A1BF1C6E47497F5ED9487C03D9FD8B49A098264240EBD692\
11A4640A5754C4F51DD6025E6BACEEC4809A1272FA5693D7FFBF30850630BF0B7F4EFF\
57059D24ED85C32BFBA675A8AC2D16EF7D7927B2EBC29D0B07EAAA85D301A320284159\
4328D281E3AAF6EC7B3B77B640628005414501EF17063EDEC0339B67D3612E7287E469\
FC120057401E70F51EC9B4
Signature algorithm: sha256WithRSAEncryption
Validity:
        Not before: Oct 22 12:00:00 2013 GMT
        Not after: Oct 22 12:00:00 2028 GMT
Purpose: SSL clientSSL serverCRL signingAny PurposeOCSP helper
```

Chương 4. MD5 COLLISION ATTACKS

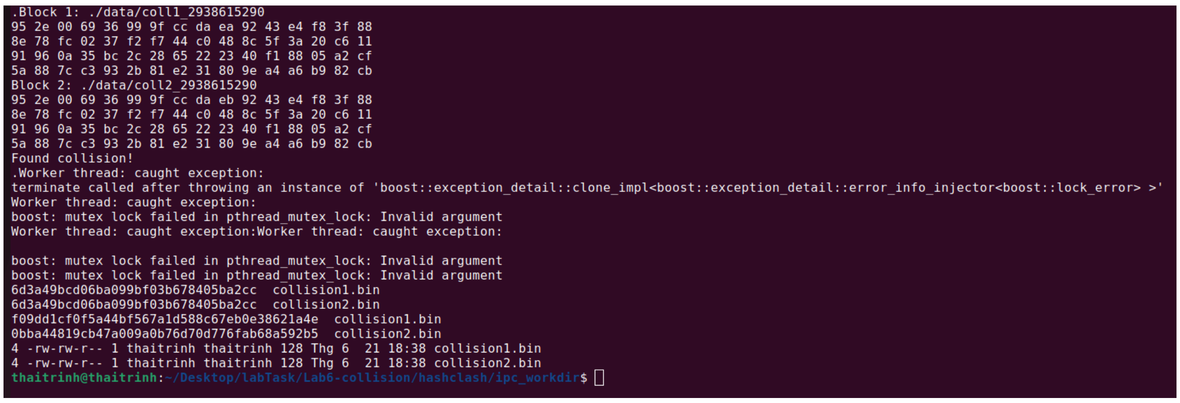
4.1. Two collision messages have the same prefix string

- Generate yourself prefix string

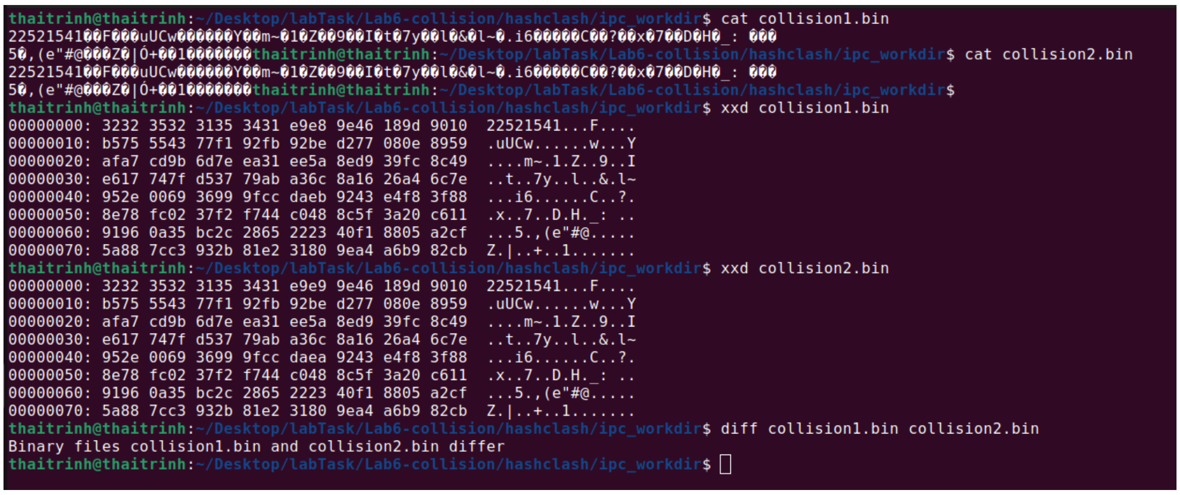
- Compute the two output files that have the same MD5 digest

Chạy lệnh:

```
echo -n "22521541" > prefix.txt

../scripts/generic_ipc.sh prefix.txt
```



```
.Block 1: ./data/coll1_2938615290
95 2e 00 69 36 99 9f cc da ea 92 43 e4 f8 3f 88
8e 78 fc 02 37 f2 f7 44 c0 48 8c 5f 3a 20 c6 11
91 96 0a 35 bc 2c 28 65 22 23 40 f1 88 05 a2 cf
5a 88 7c c3 93 2b 81 e2 31 80 9e a4 a6 b9 82 cb
Block 2: ./data/coll2_2938615290
95 2e 00 69 36 99 9f cc da eb 92 43 e4 f8 3f 88
8e 78 fc 02 37 f2 f7 44 c0 48 8c 5f 3a 20 c6 11
91 96 0a 35 bc 2c 28 65 22 23 40 f1 88 05 a2 cf
5a 88 7c c3 93 2b 81 e2 31 80 9e a4 a6 b9 82 cb
Found collision!
.Worker thread: caught exception:
terminate called after throwing an instance of 'boost::exception_detail::clone_impl<boost::exception_detail::error_info_injector<boost::lock_error> >'
Worker thread: caught exception:
boost: mutex lock failed in pthread_mutex_lock: Invalid argument
Worker thread: caught exception:Worker thread: caught exception:

boost: mutex lock failed in pthread_mutex_lock: Invalid argument
boost: mutex lock failed in pthread_mutex_lock: Invalid argument
6d3a49bcd06ba099bf03b678405ba2cc  collision1.bin
6d3a49bcd06ba099bf03b678405ba2cc  collision2.bin
f09dd1cf0f5a44bf567a1d588c67eb0e38621a4e  collision1.bin
0bba44819cb47a009a0b76d70d776fab68a592b5  collision2.bin
4 -rw-rw-r-- 1 thaitrinh thaitrinh 128 Thg 6  21 18:38 collision1.bin
4 -rw-rw-r-- 1 thaitrinh thaitrinh 128 Thg 6  21 18:38 collision2.bin
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/ipc_workdir$
```



```
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/ipc_workdir$ cat collision1.bin
22521541��F���uUCw������Y��m~�1�Z��9��I�t�7y��l�&�l~�.i6�����C��?��x�7��D�H�_: ���
5�,(e"#@���Z�|Ó+��1�������thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/ipc_workdir$ cat collision2.bin
22521541��F���uUCw������Y��m~�1�Z��9��I�t�7y��l�&�l~�.i6�����C��?��x�7��D�H�_: ���
5�,(e"#@���Z�|Ó+��1�������thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/ipc_workdir$
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/ipc_workdir$ xxd collision1.bin
00000000: 3232 3532 3135 3431 e9e8 9e46 189d 9010  22521541...F....
00000010: b575 5543 77f1 92fb 92be d277 080e 8959  .uUCw......w...Y
00000020: afa7 cd9b 6d7e ea31 ee5a 8ed9 39fc 8c49  ....m~.1.Z..9..I
00000030: e617 747f d537 79ab a36c 8a16 26a4 6c7e  ..t..7y..l..&.l~
00000040: 952e 0069 3699 9fcc daeb 9243 e4f8 3f88  ...i6......C..?.
00000050: 8e78 fc02 37f2 f744 c048 8c5f 3a20 c611  .x..7..D.H._: ..
00000060: 9196 0a35 bc2c 2865 2223 40f1 8805 a2cf  ...5.,(e"#@.....
00000070: 5a88 7cc3 932b 81e2 3180 9ea4 a6b9 82cb  Z.|..+..1.......
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/ipc_workdir$ xxd collision2.bin
00000000: 3232 3532 3135 3431 e9e9 9e46 189d 9010  22521541...F....
00000010: b575 5543 77f1 92fb 92be d277 080e 8959  .uUCw......w...Y
00000020: afa7 cd9b 6d7e ea31 ee5a 8ed9 39fc 8c49  ....m~.1.Z..9..I
00000030: e617 747f d537 79ab a36c 8a16 26a4 6c7e  ..t..7y..l..&.l~
00000040: 952e 0069 3699 9fcc daea 9243 e4f8 3f88  ...i6......C..?.
00000050: 8e78 fc02 37f2 f744 c048 8c5f 3a20 c611  .x..7..D.H._: ..
00000060: 9196 0a35 bc2c 2865 2223 40f1 8805 a2cf  ...5.,(e"#@.....
00000070: 5a88 7cc3 932b 81e2 3180 9ea4 a6b9 82cb  Z.|..+..1.......
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/ipc_workdir$ diff collision1.bin collision2.bin
Binary files collision1.bin and collision2.bin differ
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/ipc_workdir$
```

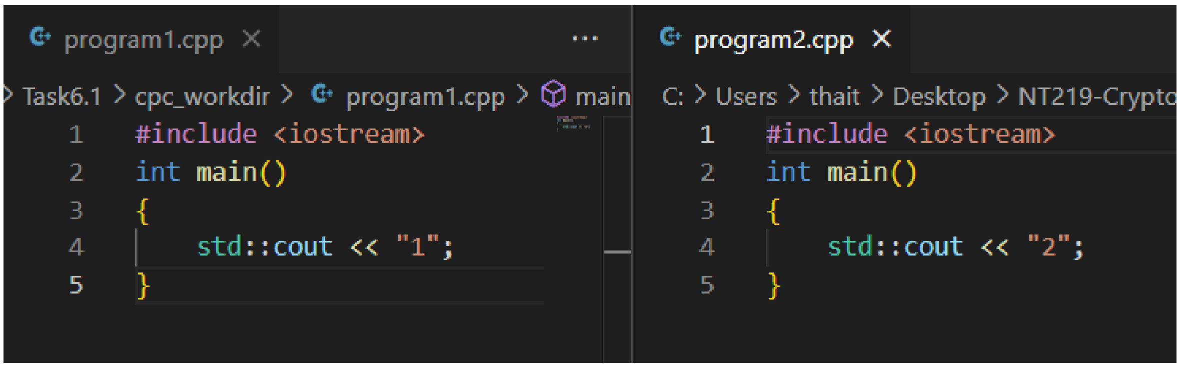
4.2. Two different C++ programs but have the same MD5

- Code yourself two short C++ programs

- Compiler your codes code1, code2

- Run hashclash to generate two program with the same MD5 digest

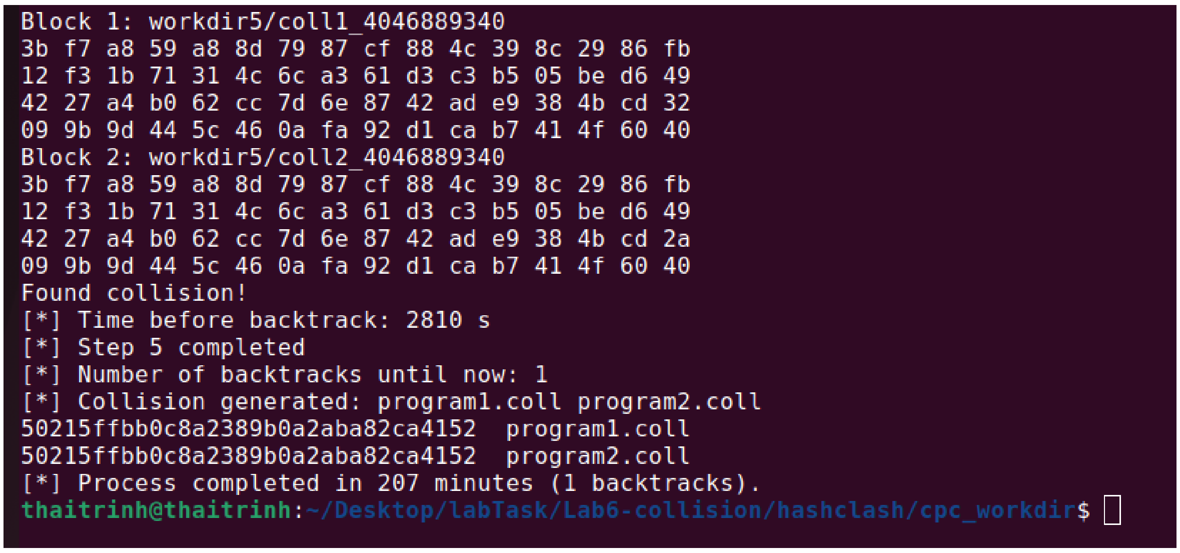
Viết 2 chương trình khác nhau và build thành file thực thi:


program1.cpp
Task6.1 > cpc_workdir > program1.cpp > main
program2.cpp
C: > Users > thait > Desktop > NT219-Crypto

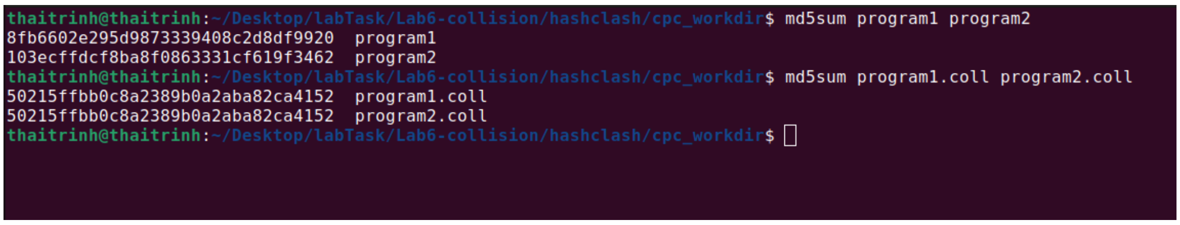
```cpp
#include <iostream>
int main()
{
    std::cout << "1";
}
```

```cpp
#include <iostream>
int main()
{
    std::cout << "2";
}
```

Chạy lệnh `../scripts/cpc.sh program1 program2`. Thời gian chạy 207 phút.



```
Block 1: workdir5/coll1_4046889340
3b f7 a8 59 a8 8d 79 87 cf 88 4c 39 8c 29 86 fb
12 f3 1b 71 31 4c 6c a3 61 d3 c3 b5 05 be d6 49
42 27 a4 b0 62 cc 7d 6e 87 42 ad e9 38 4b cd 32
09 9b 9d 44 5c 46 0a fa 92 d1 ca b7 41 4f 60 40
Block 2: workdir5/coll2_4046889340
3b f7 a8 59 a8 8d 79 87 cf 88 4c 39 8c 29 86 fb
12 f3 1b 71 31 4c 6c a3 61 d3 c3 b5 05 be d6 49
42 27 a4 b0 62 cc 7d 6e 87 42 ad e9 38 4b cd 2a
09 9b 9d 44 5c 46 0a fa 92 d1 ca b7 41 4f 60 40
Found collision!
[*] Time before backtrack: 2810 s
[*] Step 5 completed
[*] Number of backtracks until now: 1
[*] Collision generated: program1.coll program2.coll
50215ffbb0c8a2389b0a2aba82ca4152  program1.coll
50215ffbb0c8a2389b0a2aba82ca4152  program2.coll
[*] Process completed in 207 minutes (1 backtracks).
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/cpc_workdir$
```

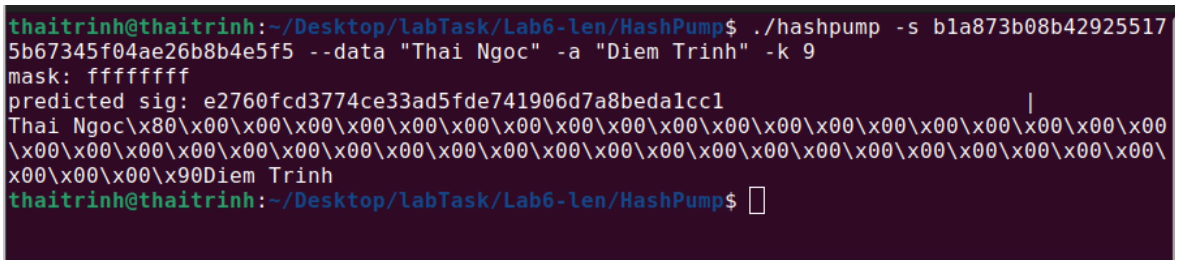


```
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/cpc_workdir$ md5sum program1 program2
8fb6602e295d9873339408c2d8df9920  program1
103ecffdcf8ba8f0863331cf619f3462  program2
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/cpc_workdir$ md5sum program1.coll program2.coll
50215ffbb0c8a2389b0a2aba82ca4152  program1.coll
50215ffbb0c8a2389b0a2aba82ca4152  program2.coll
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-collision/hashclash/cpc_workdir$
```

Chương 5. LENGTH EXTENSION ATTACKS ON MAC IN FORM: H(K||M), K IS SECRET KEY

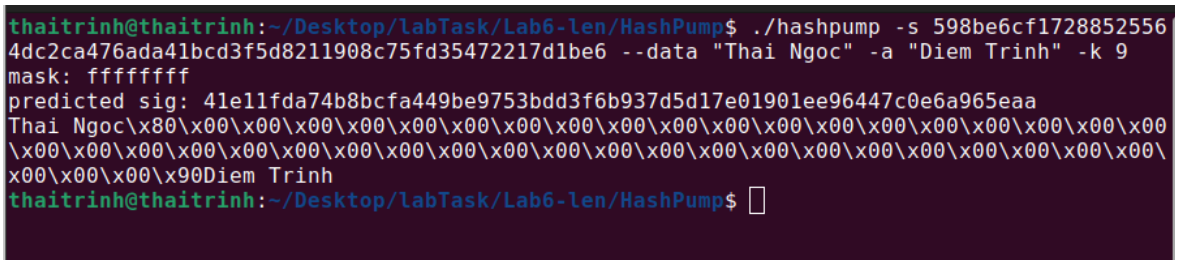
5.1. Show length extension attacks on MAC using SHA1, SHA256, SHA512 using HashPump tool

SHA1:



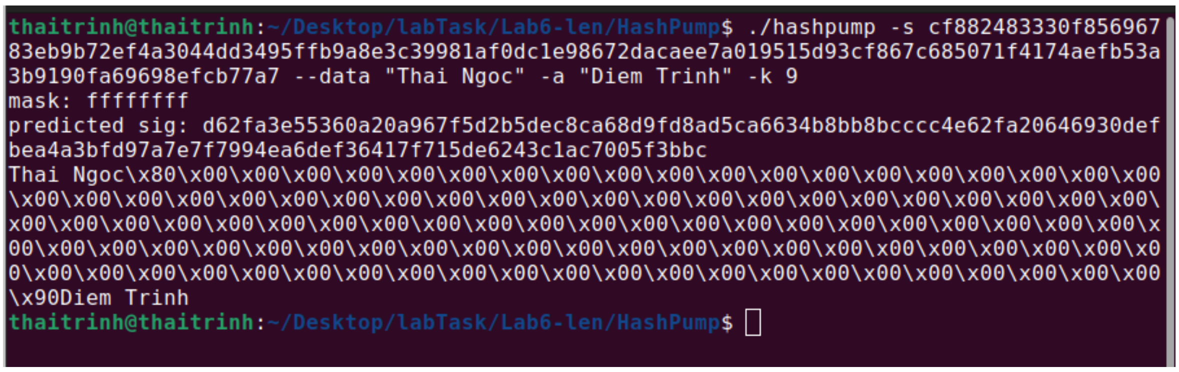
```
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-len/HashPump$ ./hashpump -s b1a873b08b42925517
5b67345f04ae26b8b4e5f5 --data "Thai Ngoc" -a "Diem Trinh" -k 9
mask: ffffffff
predicted sig: e2760fcd3774ce33ad5fde741906d7a8beda1cc1
Thai Ngoc\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x90Diem Trinh
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-len/HashPump$ 
```

SHA256:



```
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-len/HashPump$ ./hashpump -s 598be6cf1728852556
4dc2ca476ada41bcd3f5d8211908c75fd35472217d1be6 --data "Thai Ngoc" -a "Diem Trinh" -k 9
mask: ffffffff
predicted sig: 41e11fda74b8bcfa449be9753bdd3f6b937d5d17e01901ee96447c0e6a965eaa
Thai Ngoc\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x90Diem Trinh
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-len/HashPump$ 
```

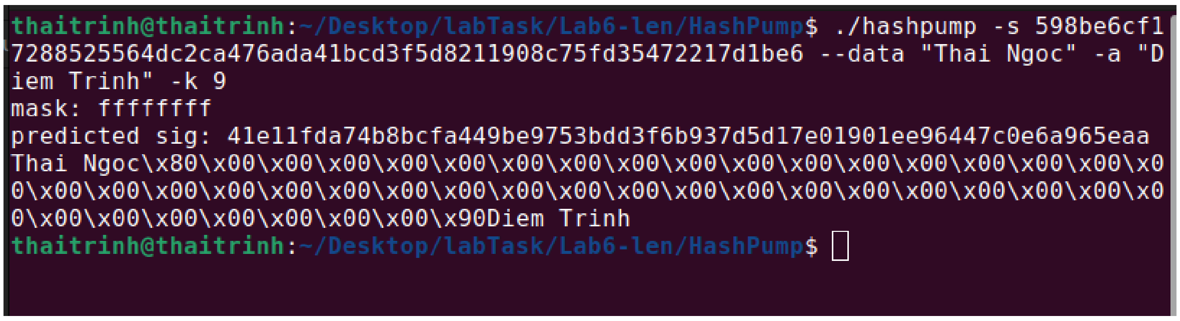
SHA512:



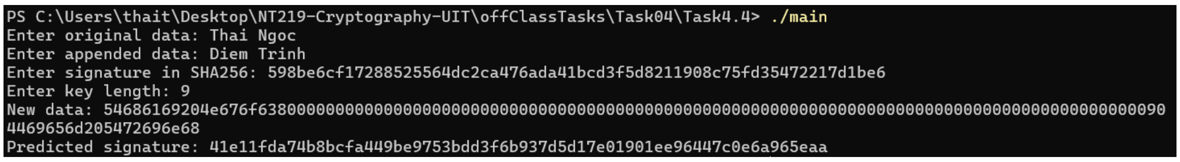
```
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-len/HashPump$ ./hashpump -s cf882483330f856967
83eb9b72ef4a3044dd3495ffb9a8e3c39981af0dc1e98672dacaee7a019515d93cf867c685071f4174aefb53a
3b9190fa69698efcb77a7 --data "Thai Ngoc" -a "Diem Trinh" -k 9
mask: ffffffff
predicted sig: d62fa3e55360a20a967f5d2b5dec8ca68d9fd8ad5ca6634b8bb8bcccc4e62fa20646930def
bea4a3bfd97a7e7f7994ea6def36417f715de6243c1ac7005f3bbc
Thai Ngoc\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x90Diem Trinh
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-len/HashPump$ 
```

5.2. Coding self programs that can attacks on MAC using SHA256 (for bonus 5/100 points)

- Automatic compute the padded part for any input (k||m);
- Compute the digest using length extension attacks with any extend string;



```
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-len/HashPump$ ./hashpump -s 598be6cf1
7288525564dc2ca476ada41bcd3f5d8211908c75fd35472217d1be6 --data "Thai Ngoc" -a "D
iem Trinh" -k 9
mask: ffffffff
predicted sig: 41e11fda74b8bcfa449be9753bdd3f6b937d5d17e01901ee96447c0e6a965eaa
Thai Ngoc\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x90Diem Trinh
thaitrinh@thaitrinh:~/Desktop/labTask/Lab6-len/HashPump$ 
```



```
PS C:\Users\thait\Desktop\NT219-Cryptography-UIT\offClassTasks\Task04\Task4.4> ./main
Enter original data: Thai Ngoc
Enter appended data: Diem Trinh
Enter signature in SHA256: 598be6cf17288525564dc2ca476ada41bcd3f5d8211908c75fd35472217d1be6
Enter key length: 9
New data: 5468616920 4e676f638000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000090
4469656d205472696e68
Predicted signature: 41e11fda74b8bcfa449be9753bdd3f6b937d5d17e01901ee96447c0e6a965eaa
```

Chương 6. **ATTACK ON DIGITAL CERTIFICATE (FOR BONUS 10/100 POINTS)**

- Generate a digital certificate using MD5 and RSA using openssl;
- Compute an other digital certificate with the same signature but other subject using hashclash tool