CRIPTOGRAFIA MAII - FIB

**The cryptographic hash function SHA-256**

# General description

SHA-256 (secure hash algorithm, FIPS 182-2) is a cryptographic hash function with digest length of 256 bits. It is a keyless hash function; that is, an MDC (Manipulation Detection Code).

A message is processed by blocks of 512 = 16 × 32 bits, each block requiring 64 rounds.

# Basic operations

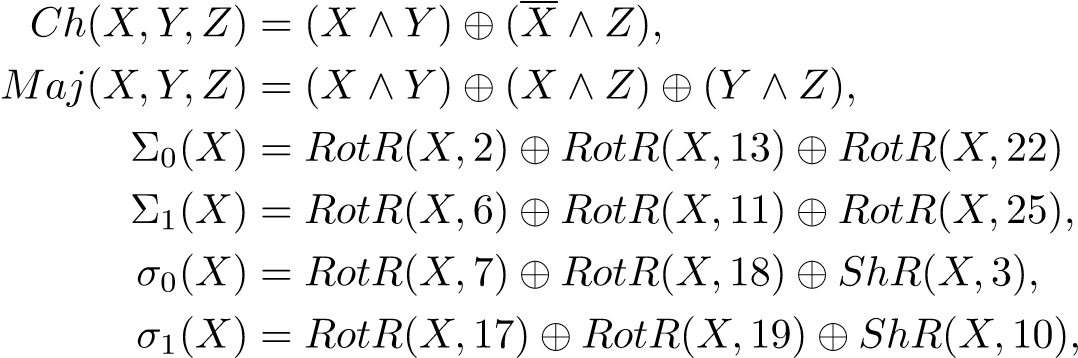
* Boolean operations AND, XOR and OR, denoted by ∧, ⊕ and ∨, respectively.
* Bitwise complement, denoted by ¯.
* Integer addition modulo 232, denoted by *A* + *B*.

Each of them operates on 32-bit words. For the last operation, binary words are interpreted as integers written in base 2.

* *RotR*(*A,n*) denotes the circular right shift of *n* bits of the binary word *A*.
* *ShR*(*A,n*) denotes the right shift of *n* bits of the binary word *A*.
* *A*k*B* denotes the concatenation of the binary words *A* and *B*.

# Functions and constants

The algorithm uses the functions:

*,*

and the 64 binary words *Ki* given by the 32 first bits of the fractional parts of the cube roots of the first 64 prime numbers:

0x428a2f98 0x71374491 0xb5c0fbcf 0xe9b5dba5 0x3956c25b 0x59f111f1 0x923f82a4 0xab1c5ed5

0xd807aa98 0x12835b01 0x243185be 0x550c7dc3 0x72be5d74 0x80deb1fe 0x9bdc06a7 0xc19bf174

0xe49b69c1 0xefbe4786 0x0fc19dc6 0x240ca1cc 0x2de92c6f 0x4a7484aa 0x5cb0a9dc 0x76f988da

0x983e5152 0xa831c66d 0xb00327c8 0xbf597fc7 0xc6e00bf3 0xd5a79147 0x06ca6351 0x14292967

0x27b70a85 0x2e1b2138 0x4d2c6dfc 0x53380d13 0x650a7354 0x766a0abb 0x81c2c92e 0x92722c85

0xa2bfe8a1 0xa81a664b 0xc24b8b70 0xc76c51a3 0xd192e819 0xd6990624 0xf40e3585 0x106aa070

0x19a4c116 0x1e376c08 0x2748774c 0x34b0bcb5 0x391c0cb3 0x4ed8aa4a 0x5b9cca4f 0x682e6ff3

0x748f82ee 0x78a5636f 0x84c87814 0x8cc70208 0x90befffa 0xa4506ceb 0xbef9a3f7 0xc67178f2

# Padding

To ensure that the message[[1]](#footnote-1) has length multiple of 512 bits:

* first, a bit 1 is appended,
* next, *k* bits 0 are appended, with *k* being the smallest positive integer such that *l* + 1 + *k* ≡ 448 mod 512, where *l* is the length in bits of the initial message,
* finally, the length *l <* 264 of the initial message is represented with exactly 64 bits, and these bits are added at the end of the message.

The message shall always be padded, even if the initial length is already a multiple of 512.

# Block decomposition

For each block *M* ∈ {0*,*1}512, 64 words of 32 bits each are constructed as follows: • the first 16 are obtained by splitting *M* in 32-bit blocks *M* = *W*1k*W*2k···k*W*15k*W*16 • the remaining 48 are obtained with the formula:

*Wi* = *σ*1(*Wi*−2) + *Wi*−7 + *σ*0(*Wi*−15) + *Wi*−16*,* 17 ≤ *i* ≤ 64*.*

# Hash computation

* First, eight variables are set to their initial values, given by the first 32 bits of the fractional part of the square roots of the first 8 prime numbers:

0xbb67ae85 0x3c6ef372 0xa54ff53a

0x510e527f 0x9b05688c 0x1f83d9ab 0x5be0cd19

* Next, the blocks *M*(1), *M*(2)*,...,M*(*N*) are processed one at a time:

## For *t* =1 to *N*

* construct the 64 blocks *Wi* from *M*(*t*), as explained above
* set



* do 64 rounds consisting of:

*T*1 = *h* + Σ1(*e*) + *Ch*(*e,f,g*) + *Ki* + *Wi*

*T*2 = Σ0(*a*) + *Maj*(*a,b,c*) *h* = *g g* = *f f* = *e*

*e* = *d* + *T*1 *d* = *c c* = *b b* = *a*

*a* = *T*1 + *T*2

* compute the new value of

|  |  |  |
| --- | --- | --- |
|  | = |  |
|  | = |  |
|  | = |  |
|  | = |  |
|  | = |  |
|  | = |  |
|  | = |  |
| End for | = |  |

• The hash of the message is the concatenation of the variables *HiN* after the last block has been processed

*.*

# Implementation: signatures

Implement the cryptographic hash function just described. Define the class sha256 with the method: public static BigInteger hash(byte[] M)

input: M is a chain of bytes of arbitrary length; output: a positive integer in the interval [0*,*2256), the value of the hash of M.

# Test values

To check the implementation, you can use the following values, given in hexadecimal notation.

|  |  |
| --- | --- |
| input | 61 62 63 |
| hash | ba7816bf 8f01cfea 414140de 5dae2223 b00361a3 96177a9c b410ff61 f20015ad |
| input | 61 62 63 64 62 63 64 65 63 64 65 66 64 65 66 67 65 66 67 68 66 67 68 69 67  68 69 6a 68 69 6a 6b 69 6a 6b 6c 6a 6b 6c 6d 6b 6c 6d 6e 6c 6d 6e 6f 6d 6e  6f 70 6e 6f 70 71 |
| hash | 248d6a61 d20638b8 e5c02693 0c3e6039 a33ce459 64ff2167 f6ecedd4 19db06c1 |
| input | One million of 61 |
| hash | cdc76e5c 9914fb92 81a1c7e2 84d73e67 f1809a48 a497200e 046d39cc c7112cd0 |

1. We assume that the length of the message can be represented by a 64-bit integer. [↑](#footnote-ref-1)