NAME

sha*sum, **b2sum**, **md5sum** — generate or verify cryptographic hashes

SYNOPSIS

```
sha1sum [-ztb] [--tag] [file]...
sha224sum [-ztb] [--tag] [file]...
sha256sum [-ztb] [--tag] [file]...
sha384sum [-ztb] [--tag] [file]...
sha512sum [-ztb] [--tag] [file]...
b2sum [-ztb] [-l BITS] [--tag] [file]...
md5sum [-ztb] [--tag] [file]...

sha1sum -c [-w] [--strict] [--quiet] [--status] [--ignore-missing] [file]...
sha224sum -c [-w] [--strict] [--quiet] [--status] [--ignore-missing] [file]...
sha256sum -c [-w] [--strict] [--quiet] [--status] [--ignore-missing] [file]...
sha384sum -c [-w] [--strict] [--quiet] [--status] [--ignore-missing] [file]...
sha512sum -c [-w] [--strict] [--quiet] [--status] [--ignore-missing] [file]...
b2sum -c [-l BITS] [-w] [--strict] [--quiet] [--status] [--ignore-missing]
        [file]...
md5sum -c [-w] [--strict] [--quiet] [--status] [--ignore-missing] [file]...
```

DESCRIPTION

Without **-c**, hash *file*s (standard input stream if "**-**", the default), writing the hashes to the standard output stream. With **-c**, verify hashes from *file*s (likewise), against on-disk state.

Program	Digest	Standard	Length	
sha1sum	SHA-1	FIPS PUB 180	160 bits	avoid use in security applications
sha224sum, **sha256sum**, **sha384sum**, **sha512sum**	SHA-2	FIPS PUB 180-2	respective	
b2sum	BLAKE2b	RFC 7693	8-512 bits	
md5sum	MD5	RFC 1321	128 bits	*NOT* suitable for any application

Generation

When run without **-c** nor **--tag**, a listing is produced in the following form:

```
$ sha1sum README.md
22eb73bd30d47540a4e05781f0f6e07640857cae  README.md
```

With **-b**, an asterisk replaces the second space:

```
$ echo "POSIX.2" | sha1sum -b
33210013f52127d6ada425601f16bbb62e85f3be *-
```

With **--tag**, the output is in the form

```
$ echo "POSIX.2" | sha1sum --tag LICENCE -
SHA1 (LICENCE) = bd25664d6e803060dcb31bfdd9642ba9d8a3f1b9
SHA1 (-) = 33210013f52127d6ada425601f16bbb62e85f3be
```

A dash and width in bits is appended for non-default digest lengths:

```
$ echo "POSIX.2" | b2sum --tag -l 96
BLAKE2b-96 (-) = 386b90bea2a1e566249cdb96
```

If the filename contains a backslash or newline characters, they're replaced with "\\" and "**\n**", respectively, and a backslash is prepended to the line:

```
$ echo 'trademark of AT&T' > "$(printf 'UNIX\nregistered')"
$ sha1sum "$(printf 'UNIX\nregistered')"
\7390a4a0bfb7c6da55d6f5f3db4e42c534271363  UNIX\nregistered
$ sha1sum --tag "$(printf 'UNIX\nregistered')"
\SHA1 (UNIX\nregistered) = 7390a4a0bfb7c6da55d6f5f3db4e42c534271363
```

With **-z**, lines are separated by NUL bytes and escaping does not occur.

Verification

With **-c**, the *file*s are instead the output: either in the default format with lines consisting of an even non-zero amount of hexadecimal digits (any case), a space, a space or an asterisk and a filename; or in the **--tag** format, with the algorithm to be used, parenthesised filename, **=**, and hexadecimal digits, each separated with a space. If the first character of a line is a backslash, it's skipped and the filename is de-escaped. With **-w**, an error is written to the standard error stream for each invalid line. With **--strict**, invalid lines yield a non-zero exit code. Be wary of using "**-**" *file*s and "**-**" hashed files.

For each valid line, the file is hashed, compared to the listed hash, and a verdict is issued to the standard output stream:

```
      file: verdict comment
```

Where *verdict* is either **OK** or **FAILED**, and the *comment* is **(I/O)** if the file couldn't be hashed (a diagnostic is also issued in this case). If **--ignore-missing** is specified and the error is ENOENT, the file is silently ignored. With **--quiet**, *verdict*s of **OK** are skipped.

After each *file*, a summary is written to the standard error stream if it had any errors.

With **--status**, all output, except for diagnostics, is suppressed.

Consider the following shell transcript:

```
$ echo "POSIX.2" | sha1sum LICENCE - "$(printf 'UNIX\nregistered')" |
                   tee sum
22eb73bd30d47540a4e05781f0f6e07640857cae  LICENCE
33210013f52127d6ada425601f16bbb62e85f3be  -
\7390a4a0bfb7c6da55d6f5f3db4e42c534271363  UNIX\nregistered

$ echo "POSIX.2" | sha1sum -c sum; echo $?
LICENCE: OK
-: OK
UNIX\nregistered: OK
0

$ rm UNIX* LICENCE
$ ln -s LICENCE LICENCE
$ echo "POSIX.2" | sha1sum -c sum; echo $?
sha1sum: sum: LICENCE: Too many levels of symbolic links
LICENCE: FAILED (I/O)
-: OK
sha1sum: sum: UNIX\nregistered: No such file or directory
UNIX\nregistered: FAILED (I/O)
sha1sum: sum: 1/3 files OK (2 missing, 0 bad lines)
1

$ echo "POSIX.2" | sha1sum -c --ignore-missing sum; echo $?
sha1sum: sum: LICENCE: Too many levels of symbolic links
LICENCE: FAILED (I/O)
-: OK
sha1sum: sum: 2/3 files OK (2 missing, 0 bad lines)
1

$ sed -iB 's/22eb73bd/22eb73bd /' sum
$ echo "POSIX.2" | sha1sum -c --ignore-missing sum; echo $?
-: OK
sha1sum: sum: 2/2 files OK (1 missing, 1 bad lines)
0

$ echo "POSIX.2" | sha1sum -c --ignore-missing --quiet sum; echo $?
```

```
sha1sum: sum: 2/2 files OK (1 missing, 1 bad lines)
0

$ echo "POSIX.2" | sha1sum -cw --ignore-missing --quiet sum; echo $?
sha1sum: sum: 1: invalid line
sha1sum: sum: 2/2 files OK (1 missing, 1 bad lines)
0

$ echo "POSIX.2" | sha1sum -cw --ignore-missing --quiet --strict sum; echo $?
sha1sum: sum: 1: invalid line
sha1sum: sum: 2/2 files OK (1 missing, 1 bad lines)
1
```

OPTIONS

-l, **--length**=*BITS*	For variable-length hashes (BLAKE2b), select the digest length. Must be a positive multiple of **8** and fall within the algorithm's domain. Default: maximum.

For generation:

-z, **--zero**	Separate lines with the NUL byte and suppress filename escaping.
-t, **--text**	Prefix filename with a space in the default output format. This is the default.
-b, **--binary**	Prefix filename with an asterisk in the default output format.
--tag	Use the alternative output format.

For verification:

-c, **--check**	Verify *file*s. Implies **-b**.
-w, **--warn**	Write errors for invalid lines.
--strict	Exit non-zero for invalid lines.
--quiet	Skip writing **OK** verifications.
--status	Suppress all output (the exit status remains unchanged).
--ignore-missing	Completely ignore files that don't exist.

ENVIRONMENT

SHASUM_NOMMAP	If available, *file*s may be mmap(2)ed to improve performance. This is observable in one way: truncating a mmap(2)ed *file* will produce an error (ENODATA), but truncating it otherwise will just produce an undetectably indeterminate result (it's impossible to predict at which point the file was truncated). This may acually lead to a *decrease* in performance in some pathological scenarios (for example, it may reduce the I/O size to single pages when reading files/devices over the network as each one is faulted in, against better posix_madvise(3)).

EXIT STATUS

1 if one of the *file*s didn't exist, an invalid line was encountered with **--strict**, or a file failed verification (its hash was different than listed).

SEE ALSO

BLAKE2	
	RFC 7693: **http://tools.ietf.org/html/7693**
MD5	*RFC 1321*: **http://tools.ietf.org/html/1321**
SHA	*FIPS PUB 180-4*: **https://csrc.nist.gov/publications/detail/fips/180/4/final**

HISTORY

Originates from the GNU system. **b2sum** appeared in coreutils 8.26.

The handling of the **-c** output options is more fine-grained than on the GNU system, which has a tendency to only use the final one.

The **`--tag`** format, indicating the algorithm, appeared in coreutils 8.20 as a "BSD-style" checksum. It's not clear what that means.

Do not rely on the format of *`comment`* or the end-of-*`file`* summary across systems.

`-tb` have no effect on UNIX, as it doesn't have file (description) types. The distinction is purely cosmetic.