ublic member function

std::istream::get

<istream> <iostream>

single character (1)	`int get();` `istream& get (char& c);`
c-string (2)	`istream& get (char* s, streamsize n);` `istream& get (char* s, streamsize n, char delim);`
stream buffer (3)	`istream& get (streambuf& sb);` `istream& get (streambuf& sb, char delim);`

Get characters

Extracts characters from the stream, as *unformatted input*:

(1) single character

Extracts a single character from the stream.
The character is either returned (first signature), or set as the value of its argument (second signature).

(2) c-string

Extracts characters from the stream and stores them in *s* as a c-string, until either `(n-1)` characters have been extracted or the *delimiting character* is encountered: the *delimiting character* being either the *newline character* (`'\n'`) or *delim* (if this argument is specified).
The *delimiting character* is **not** extracted from the input sequence if found, and remains there as the next character to be extracted from the stream (see `getline` for an alternative that *does* discard the *delimiting character*).
A *null character* (`'\0'`) is automatically appended to the written sequence if *n* is greater than zero, even if an empty string is extracted.

(3) stream buffer

Extracts characters from the stream and inserts them into the output sequence controlled by the *stream buffer* object *sb*, stopping either as soon as such an insertion fails or as soon as the *delimiting character* is encountered in the input sequence (the *delimiting character* being either the newline character, `'\n'`, or*delim*, if this argument is specified).
Only the characters successfully inserted into *sb* are extracted from the stream: Neither the *delimiting character*, nor eventually the character that failed to be inserted at *sb*, are extracted from the input sequence and remain there as the next character to be extracted from the stream.

The function also stops extracting characters if the *end-of-file* is reached. If this is reached prematurely (before meeting the conditions described above), the function sets the `eofbit` flag.

Internally, the function accesses the input sequence by first constructing a `sentry` object (with *noskipws* set to`true`). Then (if `good`), it extracts characters from its associated *stream buffer* object as if calling its member functions`sbumpc` or `sgetc`, and finally destroys the `sentry` object before returning.

The number of characters successfully read and stored by this function can be accessed by calling member `gcount`.

Parameters

`c`
The reference to a character where the extracted value is stored.

`s`
Pointer to an array of characters where extracted characters are stored as a c-string.
If the function does not extract any characters (or if the first character extracted is the *delimiter character*) and *n* is greater than zero, this is set to an empty c-string.

`n`
Maximum number of characters to write to *s* (including the terminating null character).
If this is less than `2`, the function does not extract any characters and sets `failbit`.
`streamsize` is a signed integral type.

`delim`
Explicit *delimiting character*: The operation of extracting successive characters stops as soon as the next character to extract compares equal to this.

`sb`
A `streambuf` object on whose controlled output sequence the characters are copied.

Return Value

The first signature returns the character read, or the *end-of-file* value (`EOF`) if no characters are available in the stream (note that in this case, the `failbit` flag is also set).

All other signatures always return `*this`. Note that this return value can be checked for the state of the stream (see *casting a stream to bool* for more info).

Errors are signaled by modifying the *internal state flags*:

flag	error
`eofbit`	The function stopped extracting characters because the input sequence has no more characters available (*end-of-file* reached).
`failbit`	Either no characters were written or an empty c-string was stored in *s*.
`badbit`	Error on stream (such as when this function catches an exception thrown by an internal operation). When set, the integrity of the stream may have been affected.

Multiple flags may be set by a single operation.

If the operation sets an *internal state flag* that was registered with member `exceptions`, the function throws an exception of member type `failure`.

Example

```
// istream::get example
#include <iostream>      // std::cin, std::cout
#include <fstream>       // std::ifstream

int main () {
  char str[256];

  std::cout << "Enter the name of an existing text file: ";
  std::cin.get (str,256);     // get c-string

  std::ifstream is(str);      // open file

  char c;
  while (is.get(c))           // loop getting single characters
    std::cout << c;

  is.close();                 // close file

  return 0;
}
```

This example prompts for the name of an existing text file and prints its content on the screen, using `cin.get` both to get individual characters and c-strings.

Data races

Modifies *c*, *sb* or the elements in the array pointed by *s*.
Modifies the stream object.
Concurrent access to the same stream object may cause data races, except for the standard stream object `cin`when this is *synchronized with stdio* (in this case, no data races are initiated, although no guarantees are given on the order in which extracted characters are attributed to threads).

Exception safety

Basic guarantee: if an exception is thrown, the object is in a valid state.
It throws an exception of member type `failure` if the resulting *error state flag* is not `goodbit` and member`exceptions` was set to throw for that state.
Any exception thrown by an internal operation is caught and handled by the function, setting `badbit`.
If `badbit` was set on the last call to `exceptions`, the function rethrows the caught exception.

See also

istream::getline	Get line (public member function)
istream::ignore	Extract and discard characters (public member function)
istream::gcount	Get character count (public member function)