**cin and strings.**

The extraction operator can be used on *cin* to get strings of characters in the same way as with fundamental data types:

|  |  |  |
| --- | --- | --- |
| 1 2 | string mystring;  cin >> mystring; |  |

However, *cin* extraction always considers spaces (whitespaces, tabs, new-line...) as terminating the value being extracted, and thus extracting a string means to always extract a single word, not a phrase or an entire sentence.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 | // cin with strings  #include <iostream>  #include <string>  using namespace std;  int main ()  {  string mystr;  cout << "What's your name? ";  getline (cin, mystr);  cout << "Hello " << mystr << ".\n";  cout << "What is your favorite team? ";  getline (cin, mystr);  cout << "I like " << mystr << " too!\n";  return 0;  } | What's your name? Homer Simpson  Hello Homer Simpson.  What is your favorite team? The Isotopes  I like The Isotopes too! | [Edit & Run](https://cplusplus.com/doc/tutorial/basic_io/) |

To get an entire line from *cin*, there exists a function, called *getline*, that takes the stream (*cin*) as first argument, and the string variable as second. For example:

Notice how in both calls to *getline*, we used the same string identifier (*mystr*). What the program does in the second call is simply replace the previous content with the new one that is introduced.

The standard behavior that most users expect from a console program is that each time the program queries the user for input, the user introduces the field, and then presses ENTER (or RETURN). That is to say, input is generally expected to happen in terms of lines on console programs, and this can be achieved by using *getline* to obtain input from the user. Therefore, unless you have a strong reason not to, you should always use *getline* to get input in your console programs instead of extracting from *cin*.