

# Re: map/multimap/wildcards

---

*Source:* <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.stl/2006-06/msg00021.html>

---

- *From:* "Arnaud Debaene" <[adebaene@xxxxxxxxxxxxxxxxxxxx](mailto:adebaene@xxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Fri, 9 Jun 2006 08:13:17 +0200
- 

"Jason S" <[jmsachs@xxxxxxxxxx](mailto:jmsachs@xxxxxxxxxx)> a écrit dans le message de news:  
1149790993.418819.325110@xx

Argh. I keep an index (presently implemented as `map<char *, long, pchar_cmp>` where `pchar_cmp` calls `strcmp()` appropriately; the "char \*"s are stored elsewhere & will exist as long as the map does) for character string lookup in sorted order. Works great for exact matches. Now I need an alternate accessor (need to keep the exact-match lookup around) to use it for string completion, e.g. find all the entries that start with an arbitrary string. I am *\*NOT\** going to expand this to regex searches (which I have no idea how to implement efficiently).

This seems to cry out for a multimap, I think...

No, multimap won't help you there : the only difference between a map and a multimap is that the multimap allow several objects with the same key, which the map forbids. However, the multimap manages only when comparison object and one sorting of elements.

What you are after is either `lower_bound` that was proposed by other, either, if you want a more generic solution, a multi-index container. Boost provides such a beast : see [http://www.boost.org/libs/multi\\_index/doc/index.html](http://www.boost.org/libs/multi_index/doc/index.html)

Arnaud  
MVP – VC