Detail

Map in C++ Standard Template Library

(Training materials for students)

Maps are associative containers that store elements in a mapped fashion. Each element has a key value and a mapped value. No two mapped values can have same key values.

Some basic functions associated with Map:

- **begin**() – Returns an iterator to the first element in the map
- **end**() – Returns an iterator to the theoretical element that follows last element in the map
- **size**() – Returns the number of elements in the map
- **max_size**() – Returns the maximum number of elements that the map can hold
- **empty**() – Returns whether the map is empty
- pair **insert(keyvalue, mapvalue)** – Adds a new element to the map
- **erase**(iterator position) – Removes the element at the position pointed by the iterator
- **erase**(const i)– Removes the key value 'i' from the map
- **clear**() – Removes all the elements from the map.

List of all functions of Map:

- map **insert**() in C++ STL– Insert elements with a particular key in the map container.
- map **count**() function in C++ STL– Returns the number of matches to element with key value 'i' in the map.
- map **equal_range**() in C++ STL– Returns an iterator of pairs. The pair refers to the bounds of a range that includes all the elements in the container which have a key equivalent to k.
- map **erase**() function in C++ STL– Used to erase element from the container.
- map **rend**() function in C++ STL– Returns a reverse iterator pointing to the theoretical element right before the first key-value pair in the map(which is considered its reverse end).
- map **rbegin**() function in C++ STL– Returns a reverse iterator which points to the last element of the map.
- map **find**() function in C++ STL– Returns an iterator to the element with key value 'i' in the map if found, else returns the iterator to end.

- map **crbegin**() and **crend**() function in C++ STL– **crbegin**() returns a constant reverse iterator referring to the last element in the map container. **crend**() returns a constant reverse iterator pointing to the theoretical element before the first element in the map.
- map **cbegin**() and cend() function in C++ STL– **cbegin**() returns a constant iterator referring to the first element in the map container. **cend**() returns a constant iterator pointing to the theoretical element that follows last element in the multimap.
- map **emplace**() in C++ STL– Inserts the key and its element in the map container.
- map **max_size**() in C++ STL– Returns the maximum number of elements a map container can hold.
- map **upper_bound**() function in C++ STL– Returns an iterator to the first element that is equivalent to mapped value with key value 'i' or definitely will go after the element with key value 'i' in the map
- map **operator=** in C++ STL– Assigns contents of a container to a different container, replacing its current content.
- map **lower_bound**() function in C++ STL– Returns an iterator to the first element that is equivalent to mapped value with key value 'i' or definitely will not go before the element with key value 'i' in the map.
- map **emplace_hint**() function in C++ STL– Inserts the key and its element in the map container with a given hint.
- map **value_comp**() in C++ STL– Returns the object that determines how the elements in the map are ordered ('<' by default).
- map **key_comp**() function in C++ STL– Returns the object that determines how the elements in the map are ordered ('<' by default).
- map::**size**() in C++ STL– Returns the number of elements in the map.
- map::**empty**() in C++ STL– Returns whether the map is empty.
- map::**begin**() and end() in C++ STL– begin() returns an iterator to the first element in the map. **end**() returns an iterator to the theoretical element that follows last element in the map
- map::**operator[]** in C++ STL– This operator is used to reference the element present at position given inside the operator.
- map::**clear**() in C++ STL– Removes all the elements from the map.

- map::**at**() and map::swap() in C++ STL– at() function is used to return the reference to the element associated with the key k. swap() function is used to exchange the contents of two maps but the maps must be of same type, although sizes may differ.

--- The End ---