Detail

Multiset in C++ Standard Template Library

(Training materials for students)

Multisets are a type of associative containers similar to set, with an exception that multiple elements can have same values. Some Basic Functions associated with multiset:

- **begin**() – Returns an iterator to the first element in the multiset
- **end**() – Returns an iterator to the theoretical element that follows last element in the multiset
- **size**() – Returns the number of elements in the multiset
- **max_size**() – Returns the maximum number of elements that the multiset can hold
- **empty**() – Returns whether the multiset is empty

List of functions of Multiset:

- **begin**() – Returns an iterator to the first element in the multiset.
- **end**() – Returns an iterator to the theoretical element that follows last element in the multiset.
- **size**() – Returns the number of elements in the multiset.
- **max_size**()– Returns the maximum number of elements that the multiset can hold.
- **empty**() – Returns whether the multiset is empty.
- pair **insert**(const i) – Adds a new element 'i' to the multiset.
- iterator **insert (iterator position,const i)** – Adds a new element 'i' at the position pointed by iterator.
- **erase**(iterator position)– Removes the element at the position pointed by the iterator.
- **erase**(const i)– Removes the value 'i' from the multiset.
- **clear**()– Removes all the elements from the multiset.
- **key_comp**() / **value_comp**()– Returns the object that determines how the elements in the multiset are ordered ('<' by default).
- **find**(const i)– Returns an iterator to the element 'i' in the multiset if found, else returns the iterator to end.
- **count**(const i)– Returns the number of matches to element 'i' in the multiset.
- **lower_bound**(const i)– Returns an iterator to the first element that is equivalent to 'i' or definitely will not go before the element 'i' in the multiset.

- **upper_bound**(const i)– Returns an iterator to the first element that is equivalent to ‘i’ or definitely will go after the element ‘i’ in the multiset.
- **swap**()– This function is used to exchange the contents of two multisets but the sets must be of same type, although sizes may differ.
- **operator**= – This operator is used to assign new contents to the container by replacing the existing contents.
- **emplace**()– This function is used to insert a new element into the multiset container.
- multiset **equal_range**()– Returns an iterator of pairs. The pair refers to the range that includes all the elements in the container which have a key equivalent to k.
- **emplace_hint**() – Inserts a new element in the multiset.
- **rbegin**()– Returns a reverse iterator pointing to the last element in the multiset container.
- **rend**()– Returns a reverse iterator pointing to the theoretical element right before the first element in the multiset container.
- **cbegin**()– Returns a constant iterator pointing to the first element in the container.
- **cend**()– Returns a constant iterator pointing to the position past the last element in the container.
- **crbegin**()– Returns a constant reverse iterator pointing to the last element in the container.
- **crend**()– Returns a constant reverse iterator pointing to the position just before the first element in the container.
- **get_allocator**()– Returns a copy of the allocator object associated with the multiset.

--- The End ---