**Multimap:**

Multimaps are associative containers that store elements formed by a combination of a key value and mapped value, following a specific order and multiple elements can have equivalent key.

i.e,

Multimap is similar to map with addition that muliple elements can have same keys. Rather than each element being unique, the key value and mapped value pair has to be unique in this case.

In general, Multimap allows duplicate keys.

**Program:**

#include<iostream>

#include<map>

#include<iterator>

#include<string>

int main()

{

std::multimap<std::string, int> m;

m.insert(std::pair<std::string, int> ("a", 1) );

m.insert(std::pair<std::string, int> ("c", 2) );

m.insert(std::pair<std::string, int> ("b", 3) );

m.insert(std::pair<std::string, int> ("b", 4) );

m.insert(std::pair<std::string, int> ("a", 5) );

m.insert(std::pair<std::string, int> ("b", 6) );

std::cout << "Number of elements with key a: " << m.count("a") << std::endl;

std::cout << "Number of elements with key b: " << m.count("b") << std::endl;

std::cout << "Number of elements with key c: " << m.count("c") << std::endl;

std::cout << "Elements in m: " << std::endl;

for(std::multimap <std::string, int>::iterator it = m.begin(); it != m.end(); it++)

std::cout << " [" << it->first << ", " << it->second << "] "<< std::endl;

std::pair<std::multimap <std::string, int>::iterator, std::multimap<std::string, int>::iterator> ppp;

ppp= m.equal\_range("b");

std::cout << std::endl << "Range of 'b' elements:" << std::endl;

for(std::multimap<std::string, int>::iterator it2 = ppp.first; it2 != ppp.second; ++it2)

std::cout << " [" << it2->first << ", " << it2->second << "]" << std::endl;

m.clear();

}