SETS

#include <iostream>

#include <iterator>

#include <set>

using namespace std;

int main()

{

// empty set container

set<int, greater<int> > s1;

// insert elements in random order

s1.insert(40);

s1.insert(30);

s1.insert(60);

s1.insert(20);

s1.insert(50);

// only one 50 will be added to the set

s1.insert(50);

s1.insert(10);

// printing set s1

set<int, greater<int> >::iterator itr;

cout << "\nThe set s1 is : \n";

for (itr = s1.begin();

itr != s1.end(); ++itr)

{

cout << ',' << \*itr;

}

cout << endl;

// assigning the elements from s1 to s2

set<int> s2(s1.begin(), s1.end());

// print all elements of the set s2

cout << "\nThe set s2 after assign from s1 is : \n";

for (itr = s2.begin();

itr != s2.end(); ++itr)

{

cout << ',' << \*itr;

}

cout << endl;

// remove all elements up to 30 in s2

cout

<< "\ns2 after removal of elements less than 30 :\n";

s2.erase(s2.begin(), s2.find(30));

for (itr = s2.begin();

itr != s2.end(); ++itr) {

cout << ',' << \*itr;

}

// remove element with value 50 in s2

int num;

num = s2.erase(50);

cout << "\ns2.erase(50) : ";

cout << num << " removed\n";

for (itr = s2.begin();

itr != s2.end(); ++itr)

{

cout << ',' << \*itr;

}

cout << endl;

// lower bound and upper bound for set s1

cout << "s1.lower\_bound(40) : \n"

<< \*s1.lower\_bound(40)

<< endl;

cout << "s1.upper\_bound(40) : \n"

<< \*s1.upper\_bound(40)

<< endl;

// lower bound and upper bound for set s2

cout << "s2.lower\_bound(40) :\n"

<< \*s2.lower\_bound(40)

<< endl;

cout << "s2.upper\_bound(40) : \n"

<< \*s2.upper\_bound(40)

<< endl;

return 0;

}

\*\*\*\*set.size() will print the size of the set.

OUTPUT-

The set s1 is :

,60,50,40,30,20,10

The set s2 after assign from s1 is :

,10,20,30,40,50,60

s2 after removal of elements less than 30 :

,30,40,50,60

s2.erase(50) : 1 removed

,30,40,60

s1.lower\_bound(40) :

40

s1.upper\_bound(40) :

30

s2.lower\_bound(40) :

40

s2.upper\_bound(40) :

60

VECTORS

// C++ program to illustrate the

// iterators in vector

#include <iostream>

#include <vector>

using namespace std;

int main()

{

vector<int> g1;

for (int i = 1; i <= 5; i++)

g1.push\_back(i);

cout << "Output of begin and end: ";

for (auto i = g1.begin(); i != g1.end(); ++i)

cout << \*i << " ";

cout << "\nOutput of cbegin and cend: ";

for (auto i = g1.cbegin(); i != g1.cend(); ++i)

cout << \*i << " ";

cout << "\nOutput of rbegin and rend: ";

for (auto ir = g1.rbegin(); ir != g1.rend(); ++ir)

cout << \*ir << " ";

cout << "\nOutput of crbegin and crend : ";

for (auto ir = g1.crbegin(); ir != g1.crend(); ++ir)

cout << \*ir << " ";

return 0;

}

\*\*\*vectorname.size() will give the number of elements in that vector set.

Output-

Output of begin and end: 1 2 3 4 5

Output of cbegin and cend: 1 2 3 4 5

Output of rbegin and rend: 5 4 3 2 1

Output of crbegin and crend : 5 4 3 2 1

MAPS

#include <iostream>

#include <iterator>

#include <map>

using namespace std;

int main()

{

// empty map container

map<int, int> gquiz1;

// insert elements in random order

gquiz1.insert(pair<int, int>(1, 40));

gquiz1.insert(pair<int, int>(2, 30));

gquiz1.insert(pair<int, int>(3, 60));

gquiz1.insert(pair<int, int>(4, 20));

gquiz1.insert(pair<int, int>(5, 50));

gquiz1.insert(pair<int, int>(6, 50));

gquiz1.insert(pair<int, int>(7, 10));

// printing map gquiz1

map<int, int>::iterator itr;

cout << "\nThe map gquiz1 is : \n";

cout << "\tKEY\tELEMENT\n";

for (itr = gquiz1.begin(); itr != gquiz1.end(); ++itr) {

cout << '\t' << itr->first

<< '\t' << itr->second << '\n';

}

cout << endl;

// assigning the elements from gquiz1 to gquiz2

map<int, int> gquiz2(gquiz1.begin(), gquiz1.end());

// print all elements of the map gquiz2

cout << "\nThe map gquiz2 after"

<< " assign from gquiz1 is : \n";

cout << "\tKEY\tELEMENT\n";

for (itr = gquiz2.begin(); itr != gquiz2.end(); ++itr) {

cout << '\t' << itr->first

<< '\t' << itr->second << '\n';

}

cout << endl;

// remove all elements up to

// element with key=3 in gquiz2

cout << "\ngquiz2 after removal of"

" elements less than key=3 : \n";

cout << "\tKEY\tELEMENT\n";

gquiz2.erase(gquiz2.begin(), gquiz2.find(3));

for (itr = gquiz2.begin(); itr != gquiz2.end(); ++itr) {

cout << '\t' << itr->first

<< '\t' << itr->second << '\n';

}

// remove all elements with key = 4

int num;

num = gquiz2.erase(4);

cout << "\ngquiz2.erase(4) : ";

cout << num << " removed \n";

cout << "\tKEY\tELEMENT\n";

for (itr = gquiz2.begin(); itr != gquiz2.end(); ++itr) {

cout << '\t' << itr->first

<< '\t' << itr->second << '\n';

}

cout << endl;

// lower bound and upper bound for map gquiz1 key = 5

cout << "gquiz1.lower\_bound(5) : "

<< "\tKEY = ";

cout << gquiz1.lower\_bound(5)->first << '\t';

cout << "\tELEMENT = "

<< gquiz1.lower\_bound(5)->second << endl;

cout << "gquiz1.upper\_bound(5) : "

<< "\tKEY = ";

cout << gquiz1.upper\_bound(5)->first << '\t';

cout << "\tELEMENT = "

<< gquiz1.upper\_bound(5)->second << endl;

return 0;

}

OUTPUT------------

The map gquiz1 is :

KEY ELEMENT

1 40

2 30

3 60

4 20

5 50

6 50

7 10

The map gquiz2 after assign from gquiz1 is :

KEY ELEMENT

1 40

2 30

3 60

4 20

5 50

6 50

7 10

gquiz2 after removal of elements less than key=3 :

KEY ELEMENT

3 60

4 20

5 50

6 50

7 10

gquiz2.erase(4) : 1 removed

KEY ELEMENT

3 60

5 50

6 50

7 10

gquiz1.lower\_bound(5) : KEY = 5 ELEMENT = 50

gquiz1.upper\_bound(5) : KEY = 6 ELEMENT = 50

