**Assignment-48: A Job Ready Bootcamp in c++,DSA and IOT**

**multimap**

1. Write a c++ program to demonstrate the implementation of multimap

Sol – 1.

#include<iostream>

#include<map>

using namespace std;

int main()

{

multimap<int,int>mm;

mm.insert(make\_pair(1,5));

mm.insert({5,50});

mm.insert(pair<int,int>(5,10));

mm.insert({2,45});

cout<<"Size of multimap : "<<mm.size()<<endl;

cout<<"Key\tValue"<<endl;

for(auto it=mm.begin();it!=mm.end();it++)

{

cout<<it->first<<"\t"<<(\*it).second<<endl;

}

cout<<"Enter a no to find the freq of a key : ";

int n;

cin>>n;

cout<<mm.count(n)<<" times"<<endl;

cout<<"Enter a key to remove : ";

cin>>n;

auto it=mm.find(n);

if(it!=mm.end())

{

mm.erase(it);

cout<<"Key\tValue"<<endl;

for(auto it=mm.begin();it!=mm.end();it++)

{

cout<<it->first<<"\t"<<(\*it).second<<endl;

}

}

else

{

cout<<"Key not found!"<<endl;

}

return 0;

}

2. Declare a multimap m1 of key-value pairs of integer type and then insert some pair

type data. Now print the multimap values of m1, and also create another multimap

m2 of the same type as m1 using m1.begin() and m1.end() as parameters.

Sol – 2.

#include<iostream>

#include<map>

using namespace std;

int main()

{

multimap<int,int>mm;

mm.insert({1,5});

mm.insert({5,50});

mm.insert(pair<int,int>(5,10));

mm.insert({2,45});

cout<<"Key\tValue"<<endl;

for(pair<int,int>it:mm)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

multimap<int,int>mm2(mm.begin(),mm.end());

cout<<"Multimap 2\nKey\tValue"<<endl;

for(auto it:mm2)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

return 0;

}

3. Write a c++ code for illustration of multimap::swap() function.

Sol – 3.

#include<iostream>

#include<map>

using namespace std;

int main()

{

multimap<int,int>mm;

mm.insert({1,5});

mm.insert({5,50});

mm.insert(pair<int,int>(5,10));

mm.insert({2,45});

cout<<"Multimap 1\nKey\tValue"<<endl;

for(pair<int,int>it:mm)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

multimap<int,int>mm2{{1,10},{5,12},{2,14}};

cout<<"Multimap 2\nKey\tValue"<<endl;

for(auto it:mm2)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

mm2.swap(mm);

cout<<"After Swapping"<<endl;

cout<<"Multimap 1\nKey\tValue"<<endl;

for(pair<int,int>it:mm)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

cout<<"Multimap 2\nKey\tValue"<<endl;

for(auto it:mm2)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

return 0;

}

4. Write a program to erase all the entries of the key.

Sol – 4.

#include<iostream>

#include<map>

using namespace std;

int main()

{

multimap<int,int>mm;

mm.insert({1,5});

mm.insert({5,50});

mm.insert(pair<int,int>(5,10));

mm.insert({2,45});

cout<<"Multimap\nKey\tValue"<<endl;

for(auto it:mm)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

mm.erase(5);

cout<<"After removing key 5"<<endl;

cout<<"Multimap\nKey\tValue"<<endl;

for(auto it:mm)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

return 0;

}

5. Write a program to erase only a single value based on position.

Sol – 5.

#include<iostream>

#include<map>

using namespace std;

int main()

{

multimap<int,int>mm;

mm.insert({1,5});

mm.insert({5,50});

mm.insert(pair<int,int>(5,10));

mm.insert({2,45});

cout<<"Multimap\nKey\tValue"<<endl;

for(auto it:mm)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

mm.erase(mm.find(5));

cout<<"After removing a key of 5"<<endl;

cout<<"Multimap\nKey\tValue"<<endl;

for(auto it:mm)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

return 0;

}

6. Write a program to find some key value pairs and print on the console.

Sol – 6.

#include<iostream>

#include<map>

using namespace std;

int main()

{

multimap<int,int>mm;

mm.insert(make\_pair(1,5));

mm.insert({5,50});

mm.insert(pair<int,int>(5,10));

mm.insert({2,45});

int key=5,flag=0;

for(auto it:mm)

{

if(it.first==key)

{

if(flag==0)

cout<<"Key\tValue"<<endl;

flag=1;

cout<<it.first<<"\t"<<it.second<<endl;

}

}

if(flag==0)

cout<<"Key not found!"<<endl;

return 0;

}

7. Write a program to find a lower bound.

Sol – 7.

#include<iostream>

#include<map>

using namespace std;

int main()

{

multimap<int,int>mm;

mm.insert(make\_pair(1,5));

mm.insert({5,50});

mm.insert(pair<int,int>(5,10));

mm.insert({2,45});

auto it=mm.lower\_bound(1);

cout<<"Lower bound of 1 is "<<it->first<<" "<<it->second<<endl;

it=mm.lower\_bound(3);

cout<<"Lower bound of 3 is "<<it->first<<" "<<it->second<<endl;

it=mm.lower\_bound(6);

cout<<"Lower bound of 6 is "<<it->first<<" "<<it->second<<endl;

//this is pointing after end position

return 0;

}

8. Write a program to find the upper bound.

Sol – 8.

#include<iostream>

#include<map>

using namespace std;

int main()

{

multimap<int,int>mm;

mm.insert({5,50});

mm.insert(make\_pair(1,5));

mm.insert(pair<int,int>(5,10));

mm.insert({2,45});

auto it=mm.upper\_bound(1);

cout<<"Upper bound of 1 is "<<it->first<<" "<<it->second<<endl;

it=mm.upper\_bound(3);

cout<<"Upper bound of 3 is "<<it->first<<" "<<it->second<<endl;

it=mm.upper\_bound(6);

cout<<"Upper bound of 6 is "<<it->first<<" "<<it->second<<endl;

//this is pointing after end position

return 0;

}

9. You are given an array A of size N. You need to insert the elements of A into a

multimap(element as key and index as value) and display the results. Also, you need

to erase a given element x from the multimap and print "erased x" if successfully

erased, else print "not found".

Sol – 9.

#include<iostream>

#include<map>

using namespace std;

multimap<int,int>multimapInsert(int a[],int n)

{

multimap<int,int>mp;

for(int i=0;i<n;i++)

{

mp.insert({a[i],i});

}

return mp;

}

void multimapDisplay(multimap<int,int>mp)

{

cout<<"Multimap"<<endl<<"Key\tValue"<<endl;

for(auto it:mp)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

}

void multimapErase(multimap<int,int>&mp,int x)

{

if(mp.count(x))

{

cout<<"Erased "<<x<<endl;

mp.erase(x);

}

else

cout<<x<<" not found"<<endl;

}

int main()

{

int arr[10]={1,2,3,54,45,78,56,9,59,32};

multimap<int,int>mm=multimapInsert(arr,10);

multimapDisplay(mm);

multimapErase(mm,54);

multimapDisplay(mm);

return 0;

}

10. Write a program that checks whether a given multimap is empty or not.

Sol – 10.

#include<iostream>

#include<map>

using namespace std;

multimap<int,int>multimapInsert(int a[],int n)

{

multimap<int,int>mp;

for(int i=0;i<n;i++)

{

mp.insert({a[i],i});

}

return mp;

}

void multimapDisplay(multimap<int,int>mp)

{

cout<<"Multimap"<<endl<<"Key\tValue"<<endl;

for(auto it:mp)

{

cout<<it.first<<"\t"<<it.second<<endl;

}

}

int main()

{

int arr[10]={1,2,3,54,45,78,56,9,59,32};

multimap<int,int>mm=multimapInsert(arr,10);

if(!mm.empty())

cout<<"Multimap is not empty"<<endl;

else

cout<<"Multimap is empty"<<endl;

return 0;

}