**C++ Maps**

**A. What are maps?**

A map in C++ STL is an associative container that stores key-value pairs. To elaborate, a map stores a key of some data type and its corresponding values of some data type. All keys in a map are of a single data type and all values in a map are of a single data type. A map always sorts these key-value pairs by the key elements in ascending order.

**B. Using a map in our programs**

To be able to use maps in our code, the header file <map> must be included. And the syntax to define a map is

map<data\_type\_of\_key, data\_type\_of\_value> map\_name;

Copy

The data\_type could be replaced by any data type and they could be different for both key and value.

Maps provide certain methods to be used to access and utilize the elements of a map. A few of them are discussed. To access all the methods and member functions in detail, one can visit this site, [std::map](https://cplusplus.com/reference/map/map/).

**C. Initialising a map**

A map could be initialized in a similar way we used to initialize a vector or a list. It is just that it would be pairs of elements and not just single elements while we initialize a map.  A map could be initialized with all the key-value pairs inserted in the map at the time it is defined.

map<string, int> m = {{“Munawar”, 2}, {“Kamal”, 4}};

**D. Inserting elements in a map**

We can insert elements in a map in different ways. The insertion of any element in a map takes logarithmic time, making it faster than insertion in vectors or arrays but still slower than lists.

Insertion in a map could be done using the index operators [ ]. The key is put inside the operator and the value is assigned to it. This key-value pair then gets inserted into the map.

#include <iostream>

#include <map>

using namespace std;

void display(map<string, int> m)

{

cout << "The elements are: " << endl;

for (auto it : m)

{

cout << it.first << " -> " << it.second << endl;

}

cout << endl;

}

int main()

{

map<string, int> m = {{"Munawar ", 2}, {"Kamal ", 4}};

display(m);

m["Coder"] = 3;

display(m);

}

Output:

The elements are:

Munawar-> 2

Kamal-> 4

The elements are:

Coder -> 3

Munawar-> 2

Kamal-> 4

Another method to insert an element is using the insert() method and this allows us to insert as many key-value pairs as we want. The syntax for using the insert method is

map\_name.insert({key-value pair});

An example that demonstrates the use of the insert() method is

map<string, int> m = {{"Munawar ", 2}, {"Kamal ", 4}};

display(m);

m.insert({{"Coder", 3}, {"Ali ", 5}});

display(m);

Output:

The elements are:

Munawar-> 2

Kamal-> 4

The elements are:

Coder -> 3

Munawar-> 2

Ali-> 5

Kamal-> 4

**E. Accessing/Changing elements in a map**

Changing or accessing any element in a map is very easy. We just have to use the same key we inserted the value with.

map<string, int> m = {{"Munawar ", 2}, {"Kamal ", 4}};

display(m);

m["Munawar"] = 1;

display(m);

Output:

The elements are:

Munawar-> 2

Kamal-> 4

The elements are:

Munawar-> 1

Kamal-> 4

**F. Removing elements from a map**

We can remove elements from a map in different ways. The removal of any element in a map takes logarithmic time, making it faster than removal in vectors or arrays but slower than lists.

Removal of elements from a map could be done using the erase() method. The erase() method takes both the keys or an iterator to delete elements. The syntax for using the erase method is

map\_name.erase(iterator);

//OR

map\_name.erase(key);

Here, the iterator is the pointer to that position where the element gets erased from and the key is the key of that key-value pair that needs to be removed.

An example that demonstrates the use of the erase() method using a key is

map<string, int> m = {{"Munawar ", 2}, {"Kamal ", 4}};

display(m);

m.erase("Harry");

display(m);

Output:

The elements are:

Munawar-> 2

Kaml-> 4

The elements are:

Kamal-> 4

An example that demonstrates the use of the erase() method using a pointer is

map<string, int> m = {{"Munawar ", 2}, {"Kamal ", 4}};

display(m);

m.erase(m.begin()); //deletes the first element

display(m);