Sets

All of us must have dealt with the issue of duplication of elements in a container at some point, and sets solve the issue of duplication pretty efficiently. Sets allow no duplication of elements and the elements in the set are arranged in sorted order. Sets are internally implemented using *Red-Black Trees*, which happen to be self balancing trees, whose height is at most *log(n)*.

Declaring Sets:

Unlike Vectors, Sets have only one type of constructor: *set<T> st;*
The above constructor creates an empty set of Ts.

Inserting elements into a Set:

The *insert()* method of a set takes the element to be inserted as a parameter and adds it to the set. Insertion rearranges the previous elements in the set so that a sorted order is maintained. And as one might guess, *insert()* method takes O(log(n)). Addition of 'n' elements to a set takes O(nlog(n)) . The *insert()* method returns an iterator to the element being inserted once insertion is completed. If the element is already present, the iterator to that element is returned.

Accessing/Traversing a Set:

As random access in a set is not possible, one must use iterators to access the values in a set. *st.begin()* returns an iterator to the first element. **st.begin()* will de-reference the iterator to get the first element of the set, and a similar statement holds for *st.end()*.
Note: Also use of **(st.begin() + i)* to access the i^{th} element is also not possible.

All elements of a set can be traversed by using a regular container traversal method:

```
for(set<T> :: iterator it = st.begin(); it != st.end();
it++)
	cout << *it;
```

Similar to vectors, *set<T> :: iterator* can be replaced by *auto*,

```
for(auto it = st.begin(); it != st.end(); it++)
	cout << *it;

for(auto it : st)
	cout << it;
```

Deleting elements from a Set:

The *erase()* method can be used to delete an element from the set. An iterator or value of the element to be deleted may be specified as the parameter. Ex: *st.erase(st.begin())* or *st.erase(32);*

You can read more about the generic methods related to sets at:
https://www.geeksforgeeks.org/set-in-cpp-stl/