Sets

Sets are a type of associative container in which each element has to be **unique** because the value of the element identifies it. The values are stored in a specific **sorted order** i.e. either ascending or descending.

1. To use sets, you have to include the 'set' header file

```
#include <set>
```

2. The syntax to define a set is:

```
std::set<data_type> setName;
```

3. To define a set sorted in descending order, the syntax is:

```
std::set<data_type, greater<data_type>> set_name;
```

Properties

1. **Storing order –** The set stores the elements in ***sorted*** order.
2. **Values Characteristics** – All the elements in a set have ***unique*** values.
3. **Values Nature** – The value of the element cannot be modified once it is added to the set, though it is possible to remove and then add the modified value of that element. Thus, the values are ***immutable**.
4. **Search Technique** – Sets follow the ***Binary search tree**** implementation.
5. **Arranging order –** The values in a set are ***unindexed***.

Functions

- **begin()** - Returns an iterator to the first element in the set.
- **end()** - Returns an iterator to the theoretical element that follows the last element in the set.
- **size()** - Returns the number of elements in the set.

- **empty()** - Returns whether the set is empty.
- **insert(const g)** - Adds a new element 'g' to the set.
- **insert(iterator position, const g)** - Adds a new element 'g' at the position pointed by the iterator.
- **erase(Iterator position)** - Removes the element at the position pointed by the iterator.
- **erase(const g)** - Removes the value 'g' from the set.
- **clear()** - Removes all the elements from the set.
- **find(const g)** - Returns an iterator to the element 'g' in the set if found, else returns the iterator to the end.
- **count(const g)** - Returns 1 or 0 based on whether element 'g' is present in the set or not.
- **lower_bound(const g)** - Returns an iterator to the first element that is equivalent to 'g' or definitely will not go before the element 'g' in the set.
- **upper_bound(const g)** - Returns an iterator the first element that will go after the element 'g' in the set.
- **swap()** - This function is used to exchange the contents of two sets but the sets must be of the same type, although sizes may differ.

The time complexities for doing various operations on sets are:

- Insertion of Elements – ***O(log N)***
- Deletion of Elements – ***O(log N)***