Member Functions of the set Class

constructors	Create sets
operator=	Copy the contents of a set
begin	Return the iterator pointing to the beginning of a set
clear	Erase all elements of a set
count	Count the number of elements of a set with a specific key
empty	Test whether a set is empty
end	Return the iterator pointing to the end of a set
equal_range	Find a range of a set containing all elements with a specific key
erase	Erase elements of a set
find	Find the element of a set with a specific key
insert	Insert elements into a set
key_comp	Compare two keys of a set for ordering
lower_bound	Find the first element of set whose key is not less than a specific key
max_size	Return the largest possible size of a set
rbegin	Return the reverse_iterator to the beginning of a reversed set
rend	Return the reverse_iterator pointing to the end of a reversed set
size	Return the size of a set
swap	Swap the contents of two sets
upper_bound	Find the first element of a set whose key is greater than a specific key
value_comp	Compare two values of a set for ordering

Function Prototypes

constructors	Create sets
	set (const C& cmp =less<T>) – construct an empty set, using cmp as the comparison object for the values in the set, which can either be a class implementing a function call operator or a pointer to a function
	template <class II> set (II first, II last, const C& cmp =less<T>) – construct a set from a copy of the elements, starting from the element referred by the input iterator first to the element right before the one referred by the input iterator last, using cmp as the comparison object for the values in the set, which can either be a class implementing a function call operator or a pointer to a function
	set (const set<T, C>& s) – construct a copy of the set s
destructor	Destroy a set
	~set () – deallocate all the storage capacity allocated by a set
operator=	Copy the contents of a set
	set<T, C> operator= (const set<T, C>& s) – assign a copy of the set s to a set
begin	Return the iterator pointing to the beginning of a set
	iterator begin () – return an iterator to the first element of a set
	const_iterator begin () const – const version of the iterator
clear	Erase all elements of a set
	void clear () – remove all elements of a set
count	Count the number of elements of a set with a specific key
	size_type count (const T& x) const – search a set for an element with the value x and return the number of elements with that value, which is either 1 or 0
empty	Test whether a set is empty
	bool empty () const – return whether a set is empty
end	Return the iterator pointing to the end of a set
	iterator end () – return an iterator referring to the element past the end of a set
	const_iterator end () – const version of the iterator
equal_range	Find a range of a set containing all elements with a specific key
	pair<iterator, iterator> equal_range (const T& x) const – return the bounds of a range that includes all the elements in a set whose values compare equal to x, which includes one element at most
erase	Erase elements of a set
	void erase (iterator i) – erase the element of a set at the position referred by the iterator i
	size_type erase (const T& x) – erase the element with the value x in a set and return the number of elements erased, which is 1 if the element exists; otherwise, it's 0
	void erase (iterator first, iterator last) – erase all the elements of a set between the positions referred by the iterators first and last
find	Find the element of a set with a specific key
	iterator find (const T& x) const – return an iterator to the position of the element with the value x in a set if it's found; otherwise, return the iterator end
insert	Insert elements into a set
	pair<iterator, bool> insert (const T& x) – insert a copy of the element x into a set, but the element is not inserted if another element exists in the set with the same value of x, and it returns a pair, whose first element is referring to either the newly inserted element or to the existing element, and its second element is set to

	true if the new element is inserted; otherwise, it's set to false
	iterator insert (iterator i, const T& x) – insert a copy of the element x into a set, where i is referring to the position in the set, but x will go into the correct position, and returns an iterator referring to the position of x.
	template <class II> void insert (II first, II last) – insert a copy of the elements, between the elements referred by the input iterators first and last, into a set
key_comp	Compare two keys of a set for ordering
	key_compare key_comp () const – return a comparison object associated with a set, which can be used to compare the values of two elements in the set, where key_compare is a member type of the class
lower_bound	Find the first element of set whose key is not less than a specific key
	iterator lower_bound (const T& x) const – return an iterator to the first element of a set whose value does not compares less than x, using the set's comparison object
max_size	Return the largest possible size of a set
	size_type max_size () const – return the maximum number of elements that a set can hold
rbegin	Return the reverse_iterator to the beginning of a reversed set
	reverse_iterator rbegin () – return a reverse iterator referring to the last element of a set
	const_reverse_iterator rbegin () const – const version of the reverse iterator
rend	Return the reverse_iterator pointing to the end of a reversed set
	reverse_iterator rend () – return a reverse iterator referring to the element right before the first element of a set
	const_reverse_iterator rend () const – const version of the reverse iterator
size	Return the size of a set
	size_type size () const – return the number of elements in a set
swap	Swap the contents of two sets
	void swap (set<T, C>& s) – swap the contents of a set with the set s
upper_bound	Find the first element of a set whose key is greater than a specific key
	iterator upper_bound (const T& x) const – return an iterator referring to the first element of a set whose value compares greater than x, using the set's comparison object
value_comp	Compare two values of a set for ordering
	value_compare value_comp () const – return a comparison object associated with a set, which can be used to compare the values of two elements in a set, where value_compare is a member type of the class