STL Cheat Sheet 2 – set, map

Creation

- Make an empty set of integers.

```
set<int> intSet1;
```

- Make a set of integers containing the given array of numbers.

```
int array[] = {10, 20, 30, 40};
set<int> intSet2(array, array + 4);
```

- Make an empty map from string to int.

```
map<string, int> siMap1;
```

- Make an empty map from C-string to int.

```
struct compareString {
    bool operator()(const char *s1, const char *s2) const {
        return strcmp(s1, s2) < 0;
    }
}

map<const char *, int, compareString> siMap2;
```

- Declare an iterator for a set of integers.

```
set<int>::iterator iSetItr;
```

- Declare an iterator for a string to int map (a map iterator represents a pair of key and value).

```
map<string, int>::iterator siMapItr;
```

Access and modification

- Number of items in a set (also for map).

```
intSet1.size();
```

- Get an iterator which points to the beginning of the set.

```
iSetItr = intSet1.begin();
```

- Get an iterator which points to the end of the map (one past the last element).

```
siMapItr = siMap1.end();
```

- Get the value that is pointed to by the set iterator.

```
*iSetItr
```

- Get the key that is pointed to by the map iterator.

```
siMapItr->first
```

- Get the value that is pointed to by the map iterator.

```
siMapItr->second
```

Finding

- Find an item in a set (returns an iterator).

```
intSet1.find(3)
```

- See if an item is in a set.

```
if (intSet1.find(3) != intSet1.end()) ...
```

- Find an item in a map (returns an iterator).

```
siMap1.find("hello")
```

- See if an item is in a set.

```
if (siMap1.find("hello") != siMap1.end()) ...
```

Insertion and removal

- Place an item in a set.

```
intSet1.insert(3)
```

- Place a key/value in a map.

```
siMap1["hello"] = 3
```

- Removing an item from a set.

```
intSet1.erase(intSet1.find(3))
intSet1.erase(intSet1.begin())
```

- Removing an item from a map.

```
siMap1.erase(siMap1.find("hello"))
siMap1.erase(siMap1.begin())
```

- Clearing a set or a map.

```
intSet1.clear(), siMap1.clear()
```