Set/Map creation (`#include <set>`/`#include <map>`)  

- Make an empty set of integers.
  ```cpp
  set<int> intSet;
  ```
- Make a set of integers containing the given array of numbers.
  ```cpp
  int array[] = {10, 20, 30, 40};
  set<int> intSet(array, array + 4);
  ```
- Make an empty map from string to int.
  ```cpp
  map<string, int> siMap;
  ```
- Make an empty map from C-string to int.
  ```cpp
  struct compareString {
    bool operator()(const char *a, const char *b) const { return strcmp(a, b) < 0; }
  };
  map<const char *, int, compareString> siMap;
- Declare an iterator for a set of integers; declare an iterator for a string-to-int map (a map iterator represents a pair of key and value).
  ```cpp
  set<int>::iterator iSetItr;
  map<string, int>::iterator siMapItr;
  ```

Set/Map access and modification  

- Number of items in a set (also for map).
  ```cpp
  intSet.size();
  ```
- Get an iterator which points to the beginning of the set.
  ```cpp
  iSetItr = intSet.begin();
  ```
- Get an iterator which points to the end of the map (one past the last element).
  ```cpp
  siMapItr = siMap.end();
  ```
- Get the value that is pointed to by the set iterator.
  ```cpp
  *iSetItr
  ```
- Get the key that is pointed to by the map iterator.
  ```cpp
  siMapItr->first
  ```
- Get the value that is pointed to by the map iterator.
  ```cpp
  siMapItr->second
  ```
Finding in Set/Map

- Find an item in a set (returns an iterator).
  \[ \text{intSet.find(3)} \]
- See if an item is in a set.
  \[ \text{if (intSet.find(3) != intSet.end()) ...} \]
- Find an item in a map (returns an iterator).
  \[ \text{siMap.find("hello")} \]
- See if an item is in a set.
  \[ \text{if (siMap.find("hello") != siMap.end()) ...} \]

Set/Map insertion and removal

- Place an item in a set.
  \[ \text{intSet.insert(3)} \]
- Place a key/value in a map.
  \[ \text{siMap["hello"] = 3} \]
- Removing an item from a set.
  \[ \text{intSet.erase(intSet.find(3)); intSet.erase(intSet.begin())} \]
- Removing an item from a map.
  \[ \text{siMap.erase(siMap.find("hello")); siMap.erase(siMap.begin())} \]
- Clearing a set or a map.
  \[ \text{intSet.clear(), siMap.clear()} \]

Getline and String streams (\#include <sstream>)

- Getline can be used to grab a line of input at a time – everything until the next newline.
  \[ \text{string line; getline(cin, line);} \]
- Using an input string stream to parse the words obtained by getline:
  \[ \text{string line, token;}
      \text{while (getline(cin, line)) {}
          \text{istringstream in(line);}
          \text{while (in >> token) { ...}
      \text{}} \]
- Beware of the unintended effects of the following code! If there is a newline immediately following the extracted integer, then the call to getline will obtain just an empty string.
  \[ \text{int n; string line; cin >> n; getline(cin, line);} \]
- Constructing strings with ostringstream:
  \[ \text{ostringstream out;}
      \text{out << 3 << " is less than " << 3.14159 << endl;}
      \text{cout << out.str();} \]