Map

- An abstract data type for associating keys with values
  - Keys must be unique, value can be anything
  - Similar to sets (and often built on them)
    - The map stores sets of pairs or associations
    - The pair first value is the key, determines uniqueness
- Also known as a Dictionary

Example

A map of strings to strings, storing words \(\rightarrow\) definitions.

<table>
<thead>
<tr>
<th>Word</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>individual facts, statistics, or items of information</td>
</tr>
<tr>
<td>structure</td>
<td>anything composed of parts arranged together in some way; an organization</td>
</tr>
<tr>
<td>algorithm</td>
<td>a set of rules for solving a problem in a finite number of steps, as for finding the greatest common divisor</td>
</tr>
</tbody>
</table>

Example

Product database: a map of strings to tuples, storing SKUs (product id codes) \(\rightarrow\) product descriptions, prices, etc:

<table>
<thead>
<tr>
<th>SKU</th>
<th>Description</th>
<th>Color</th>
<th>Price</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>427-WHT-100-A</td>
<td>Widget, white</td>
<td>White</td>
<td>47.99</td>
<td>box</td>
</tr>
<tr>
<td>437-RED-100-A</td>
<td>Thingamajigs, red</td>
<td>Red</td>
<td>47.99</td>
<td>box</td>
</tr>
<tr>
<td>5190-FOO-66X</td>
<td>Misc. doodads</td>
<td>Black</td>
<td>12.49</td>
<td>pack of 6</td>
</tr>
</tbody>
</table>

Types of Maps

Just like sets, we have two kinds:
- Ordered maps
  - Items are stored in key order, retrievable in key order
  - Keys must be comparable
  - Typically implemented using binary search trees
- Unordered maps
  - Items are stored in no particular order
  - Typically faster than ordered maps
  - Implemented using hashtables

The Map ADT

A Map does all of these efficiently:
- Get a value associated with a key (if in map)
- Put a key/value pair into map
- Remove a key/value pair from map
- Update the value associated with a key
- Determine if the map contains a key
Stanford CPP Lib Maps (Ordered)

```
#include "map.h"

class Map<KeyType, ValueType>

(Some) operations on Map objects:

• get(key)
• put(key, value)
• remove(key)
• containsKey(key)
• operator[](key)
• size()
• isEmpty()
• clear()
• toString()
```

Map Example

```
Map<string, int> lengths;
lengths.put("apple", 5);
lengths.put("orange", 6);
lengths.put("pear", 4);
...
```

```
cout << "Ask me a word: " << endl;
string s;
cin >> s;
if (lengths.containsKey(s))
   cout << s << " has " << lengths.get(s) << " letters." << endl;
else
   cout << "I do not know the word \\
   " << s << ".";
cout << endl;
```

Another Map Example

```
Map<string, int> frequencies;
// loop through all words w in some text

frequencies[w]++;
for (string key: frequencies) {
   cout << key << ": " << frequencies[key];
}
```