COL100 Lecture 22

Note Title 25-10-2018

Review: Set (ADT)

$\log N + 1 + 1 + \ldots N$ times
$= N + \log N$
$= O(N)$

```
for-each loop
string e;
for(e : s)
{
   // do something
   // with e
}
```

s.add(value)	O(log N)
s.remove(value)	O(log N)
s.contains(value)	O(log N)
s.first()	O(log N)
s.size()	O(1)
s.clear()	O(N)
s.equals(set)	O(N)
s.isEmpty()	O(1)
s.isSubsetOf(set)	O(N)
s.toString()	O(N)

Map

ADT

(just like in Set)

Key → unique

eg. entry number

each key is associated with a value

eg. name

key $\xrightarrow{\text{maps to}}$ value

associations

each key is associated with its value

entry number → name

key's type

value's type

map

map

```
Map < string , string > m1;

map < int , string > m2;

Map < int, vector<string>> m3;
```

```
#include "map.h"
```

```
#include <map>
```

Comparison with Vectors

1. Vectors look-up by index
   Maps look-up by key
2. Need to have two types in Map
3. Ordered by key, not by index

Example:

Add a key-value mapping

```
map.put(key, value);
```

→ `map[key] = value;`

similar to vector syntax except index replaced by key

1. Adds the key if it did not already exist
2. Overwrites the value for the key

How to lookup?

map.at(key)

```
v = map.get(key);   // Stanford
V = map[key];       // STL & Stanford
```

raises exception

slightly different

Default value

What happens if key is not present in map?

adds key → default value to map
returns default value

Removing an element

```
map.remove(key);
```

1. No effect if key does not exist in map
2. Otherwise removes the key-value pair from the map

Map Member functions

Stanford	STL	Time	Description
m.put(key,value); m[key]=value;	m[key]=value	O(log N)	inserts key-value pair; overwrites value if key already present
m.get(key)	m.at(key)	O(log N)	if key exists, returns corresponding value; else exception
m[key]	m[key]	O(log N)	if key exists, returns corresponding value; else adds a default value for key to m, and returns the default value

m.remove(key)	m.erase(key)	O(log N)	If key exists, removes it and associated value; no effect otherwise
m.containsKey(key)	m.find(key) != m.end()	O(log N)	Returns true if key exists in m

Examples using Map : Dichonary

file: dichonary.txt

course
a series of lectures
tree
full grown plant
.
.
.

need not be ordered

```
ifstream ifs;
promptUserForFile (ifs, "Dictionary? ");
Map <string, vector<string>> reverse;
Map < string, string > dictionary;
string word;
while (getline (ifs, word))
{
        string meaning;
        getline (ifs, meaning);
        dictionary [word] = meaning;
        reverse[meaning].add (word);
}
// populated the dictionary
```

```
while (true)
{
        string query;
        cin >> query;
        if (dictionary.containsKey(query))
        {
            cout << "meaning: " << dictionary[query]
                 << endl;
        } else {
            cout << "I don't know that
                 word!" << endl;
        }
}
```

Ex. What if I want to know the word(s) for a given meaning?

meaning → Map<string, → list of words Vector<string>>

reverse;

e.g.:

course-id

Map < string ,

entry number

student name

Map <int , string >>

or: