Recitation Outline

- C++ STL associative containers
  - Sets and maps
- Examples
  - Simple example of using maps
  - Word frequency using maps
  - Sorting strings (a programming contest problem)
  - Adding comma into integer to separate groups of three digits
  - Word puzzle using maps
  - Word puzzle using sets

C++ STL Maps/Multimaps

- Sorted associative containers containing key/value pairs
- Sorted by their keys
- Maps: do not allow duplicates
  - Elements in maps must contain different keys
- Multimaps: allow duplicates
  - Two elements in multimaps may contain same key
- They are usually implemented as balanced binary trees

C++ STL Maps/Multimaps

- begin(), end(): iterator support
- clear(), delete all elements
- size(), number of elements
- find(key), return iterator referring to element with key
- count(key), number of elements with key
  - For maps, return value can only be 0 or 1
- empty(), true if map does not contain any elements
- erase(), delete an element or range of elements
- insert(), insert an element or range of elements
- operator[](key), retrieve corresponding value for key
  - Only available for maps
  - Automatically inserted if key is not in map
- For details, see
  - http://www.cppreference.com/wiki/

C++ STL Sets/Multisets

- Sorted associative containers containing elements of certain type
- Sorted by their elements
- Sets: do not allow duplicates
  - Elements in sets must be unique
- Multisets: allow duplicates
  - Elements in multisets may be duplicates
- They are usually implemented as balanced binary trees

C++ STL Sets/Multisets

- begin(), end(): iterator support
- clear(), delete all elements
- size(), number of elements
- find(key), return iterator referring to element with key
- count(key), number of elements with key
  - For sets, return value can only be 0 or 1
- empty(), true if set has no elements
- erase(), delete an element or range of elements
- insert(), insert an element or range of elements
- For details, see
  - http://www.cppreference.com/wiki/

A Simple Map Example

- In this example, we use map to store the number of days in each month
- See examples/r6/map_example.cpp

- Review the code
- A demo of running the program
  - To compile: make map_example.x

Word frequency using Maps

- We compute the frequency of words in a given file using map
  - A word is defined as a sequence of characters separated by spaces
- See examples/r6/word_frequency.cpp

- Review the code
- A demo of running the program with word_frequency.input
  - To compile: make word_frequency.x

Convert Integer Values

- Given an integer value, add comma to separate groups of three digits
- For example
  - 1234 -> 1,234
  - 123456 -> 123,456
  - 1234567 ->1,234,567
  - 1000001 -> 1,000,001
- We want a recursive algorithm
- See examples/r6/comma_recursive.cpp
  - Review code
- Demo
  - To compile: make comma_recursive.x

Sort Strings

- Problem statement
  - sortstring.html
- Solution using a map
  - sortstring.cpp
  - Sample input (sortstring_input)
  - To compile: make sortstring.x
- Code review and demo run

Word Puzzle using Maps

- In this version we store word dictionary in a map
- See examples/r6/word_puzzle_map.cpp, word_puzzle_map.h, rotation.cpp

- Review the code (only discussed the part updated)
- A demo of running the program

Word Puzzle using Sets

- In this version we store word dictionary in a set
- See examples/r6/word_puzzle_set.cpp, word_puzzle_set.h, rotation.cpp

- Review the code (only discuss the part that has updated)
- A demo of running the program