COMP2004 Programming Practice 2002 Summer School

Kevin Pulo
School of Information Technologies
University of Sydney

C++ STL library

- Basic concepts:
  - Assignable
    - Read and write value
  - Default Constructable
    - No args needed to construct
  - Equality Comparable
    - operator==() and operator!=()
  - LessThan Comparable
    - operator<() and operator>()

Iterators

- Restricted pointers
- Input / Output Iterator
  - Equality Comparable, Assignable, Dereference read / write, Increment
- Forward Iterator
  - Input and Output, no alternating
- Bidirectional Iterator
  - Forward, Decrement
- Random Access Iterator
  - Bidirectional, random access

Iterator Ranges

- begin and end iterators
- Range is [begin, end)
  - includes begin
  - but not end

Common output iterators

- ostream_iterator<T>(cout)
  - Outputs to ostream
- istream_iterator<T>(cin)
  - Inputs from istream
- istream_iterator<T>()
  - End-of-input iterator
- back_inserter(c), front_inserter(c)
  - Requires Front / Back Insertion Sequence

Containers

- Container
  - One Input Iterator, begin(), end()
- Forward Container = Container + Forward Iterators
- Reversible Container = Forward Container + Bidirectional Iterators, rbegin(), rend()
- Random Access Container = Reversible Container + Random Access Iterators

STL Container definitions

- Typedefs:
  - value_type
  - reference, const_reference
  - pointer, const_pointer
  - iterator, const_iterator
  - difference_type, size_type
- Methods:
  - a.size(), a.max_size()
  - a.empty(), a.swap(b)
  - a.begin(), a.end()

Container abstractions

- Sequence
  - Forward container, no reordering, add / delete anywhere
  - Front Insertion Sequence
    - push_front(), pop_front(), insert / access front quickly
  - Back Insertion Sequence
    - push_back(), pop_back(), insert / access last element quickly

Sequence types

- vector
  - Random Access Container
  - Back Insertion Sequence
  - Insertion can invalidate iterators
- list
  - Reversible Container
  - Front and Back Insertion Sequence
- deque
  - Random Access Container
  - Front and Back Insertion Sequence

Container abstractions

- Associative Container
  - Elements add / delete / access via keys
  - Unique vs Multiple
  - Simple vs Pair
  - Sorted vs Hashed

Associative Container types

- For all, deletion invalidates only deleted
- set
  - Unique, Simple, Sorted
- multiset
  - Multiple, Simple, Sorted
- map
  - Unique, Pair, Sorted
- multimap
  - Multiple, Pair, Sorted

Adapter containers

- stack
  - LIFO: only use top element
  - Use Back Insertion Sequence
- queue
  - FIFO: push back, pop front
  - Use Front and Back Insertion Seq
- priority_queue
  - Access top (largest) element
  - Use Random Access Container
  - Use LessThan Comparable elems

Function Objects

- Can be called like a function
  - Class overloads operator()
- Generator: 0 args
- Unary Function: 1 arg
- Unary Predicate: 1 arg, return bool
- Binary Function: 2 args
- Binary Predicate: 2 args, return bool
- Strict Weak Ordering: eg. less than

STL Function Objects

- Binary functions: plus, minus, ...
- Binary predicates: logical_and, logical_or, less, greater, equal_to, ...
- Unary predicates: logical_not, ...
- Unary functions: negate, ...

Adapter Function Objects

- Convert function objects, uses helpers
- bind1st(), bind2nd()
  - Convert binary function to unary
  - Allow constant value for one arg
- not1(), not2()
  - Logical not unary / binary predicate
- compose1(), compose2()
  - Composes unary / binary functions
- mem_fun_ref(), mem_fun()
  - Uses member function

Algorithms

- find - linear search for value
- find_if - linear search using predicate
- adjacent_find - linear search for adj
- find_first_of - first of possible values
- search - linear search for subrange
- find_end - like search but backwards
- search_n - first consecutive n of value
- count - counts occurances of value
- count_if - counts matches

Algorithms

- for_each - apply unary function
- accumulate - sum values
- equal - compare two ranges
- mismatch - find first difference
- lexicographical_compare -
- max_element - finds largest element
- min_element - finds smallest element

Mutating Algorithms

- Ranges are fixed
- Variants _copy, _if where appropriate
- Notables:
  - copy
  - transform
    - like for_each(), saves return values
  - replace
  - remove, unique
    - only rearranges, returns new end iter
    - use c.erase() to actually remove

Mutating Algorithms

- Notables:
  - reverse
  - random_shuffle
  - sort, partial_sort, is_sorted, merge

Exam information

- Location: MacLaurin Hall, Main Quadrangle
- Duration: 2 hrs (10 mins reading)
- Date: Friday 15 February 2002
- Time: 9:20am to 11:30am
- Format: Closed book

- Clash => see or email me ASAP

- Good luck!