SFU CMPT-212 2008-1 Topic: Generic Algorithms

Ján Maňuch

E-mail: jmanuch@sfu.ca

Sunday 13th April, 2008

Generic algorithms

- work with any container which provides *suitable* iterators
  Example:

```
1 void sort(RandomAccessIterator first,
2           RandomAccessIterator last);
```

  requires *random access iterators* — hence it can be used with `vector` and `deque` containers, but not with `list` containers (recall, the `list` template provides its own member function for sorting)

- provided in header files `<algorithm>` and `<numeric>`
- see Appendix G of the textbook for details on parameters to the following template functions

Algorithm Groups

- *nonmodifying sequence algorithms*
- *modifying sequence algorithms*
- *sorting and related algorithms*
- *numeric algorithms* (those are defined in header file `<numeric>`)

Nonmodifying sequence algorithms

- counting: `count`, `count_if`
- searching: `find`, `find_if`, `find_first_of`, `adjacent_find`
- searching subsequences: `search`, `search_n`, `find_end`
- sequence equality: `equal`, `mismatch`
- others: `for_each`

Modifying sequence algorithms

- copying: `copy`, `copy_backward`
- swapping: `swap`, `iter_swap`, `swap_ranges`
- filling: `fill`, `fill_n`
- generating: `generate`, `generate_n`
- replacing: `replace`, `replace_if`, `replace_copy`, `replace_copy_if`
- transforming: `transform`
- others: `unique`, `unique_copy`, `reverse`, `reverse_copy`, `rotate`, `rotate_copy`, `random_shuffle`, `partition`

Example:

```
1 random_shuffle(v.begin(),v.end());
2 // permutes the content of container v
```

Sorting and related algorithms

- sorting: `sort`, `stable_sort` and partial sorts, `merge`, heap algorithms
- searching: `binary_search`, `lower_bound`, `upper_bound`, `equal_range`
- set algorithms (assume that elements are sorted): `includes`, `set_union`, `set_intersection`, `set_difference`
- minmax: `min`, `max`, `min_element`, `max_element`
- others: `lexicographical_compare`, permutations (generating all)

Example (implementation of Assignment 2 using STL): set.h, set.cpp

Numeric algorithms

- defined in the header file `numeric`
- `accumulate()`, `inner_product()`, `partial_sum()`, `adjacent_difference()`

Online references:

- `www.sgi.com/tech/stl`
- `www.cppreference.com/cpp_stl.html`

Sample questions

- Write a code for a class template `accumulate`:

```
1 template <typename InputIterator, typename T,
2           typename BinaryOperation>
3 T accumulate(InputIterator first, InputIterator last,
4 BinaryOperation binary_op)
```

  which returns the sum of init and all elements in the range using operation `binary_op`.

- List four groups of the generic algorithms of STL. Give one example of generic algorithm for each group and explain what it does.