Standard Template Library (STL)

Luca Lista

INFN, Sezione di Napoli

La Standard Library

La libreria standard STL e' una libreria di classi di contenitori, algoritmi ed iteratori.
STL e' una libreria *generica:* tutti i suoi componenti sono parametrizzati mediante l'utilizzo dei template

puntatori *intelligenti*

vettori, liste, mappe, ….

find, replace, reverse, sort, ….

Iteratori (puntatori intelligenti)

Gli iteratori sono dei puntatori agli elementi di un contenitore e ci permettono di muoverci all'interno di esso:

Iteratori monodirezionali :	Permettono di accedere all'elemento successivo o al precedente
Iteratori bidirezionali :	Permettono di accedere sia all'elemento successivo che al precedente
Iteratori ad accesso casuale :	Permettono di accedere ad un qualunque elemento del contenitore

Contenitori

Un contenitore e' un oggetto capace di immagazzinare altri oggetti (i suoi elementi) e che possiede metodi per accedere ai suoi elementi. Ogni contenitore ha un iteratore associato che permette di *muoversi* tra gli elementi del contenitore.

Sequenze:
Una sequenza e' un contenitore di lunghezza variabile i cui elementi sono organizzati linearmente. E' possibile aggiungere e rimuovere elementi

Contenitori associativi:
Una sequenza e' un contenitore di lunghezza variabile che permette un efficiente accesso ai suoi elementi basato su una *chiave*.

Sequenze

vector :

- Tempo costante di inserimento e cancellazione di elementi all'inizio e alla fine del vettore.
- Tempo lineare con il numero di elementi per inserimento e cancellazione di elementi all'interno del vettore
- Iteratore ad accesso casuale

list :

- Tempo costante di inserimento e cancellazione di elementi in ogni punto della lista
- Iteratore bidirezionale

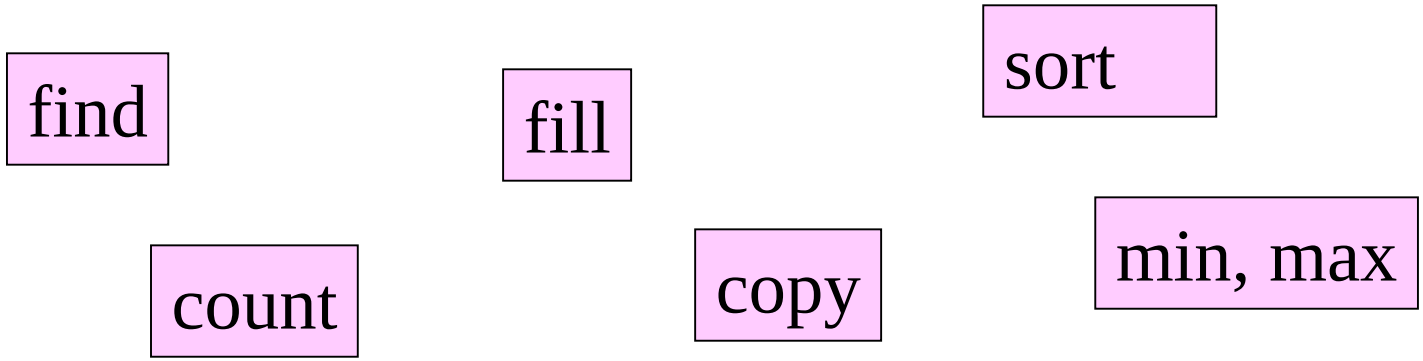
Contenitori associativi

Sono contenitore di coppie *(key, value)* e possiedono un iteratore bidirezionale

- map :
  - Viene richiesto l'operatore < per la chiave
  - Gli elementi sono ordinati secondo la chiave
- hash_map :
  - Viene richiesta una funzione di *hash* per la chiave
  - Non vi e' alcun ordinamento

Algoritmi

Gli algoritmi sono delle funzioni globali capaci di agire su contenitori differenti



Sono incluse operazioni di ordinamento (sort, merge, min, max...), di ricerca (find, count, equal...), di trasformazione (transform, replace, fill, rotate, shuffle...), e generiche operazioni numeriche (accumulate, adjacent difference...).

Esempio uso sequenze

```
#include <vector>
#include <algorithm>
#include <iostream>

int main() {

vector<int> container;
int val;
for (int i=0; i<10; i++) {
   val = (int)((float)rand()/RAND_MAX*10);
   container.push_back(val); }
vector<int>::iterator it1;
for ( it1=container.begin(); it1!=container.end(); it1++)
   cout << "vector : " << *it1 << endl;
find ( container.begin(), container.end(), 3 );
sort ( container.begin(), container.end() );
for ( it1=container.begin(); it1!=container.end(); it1++)
   cout << "vector : " << *it1 << endl;
min_element ( container.begin(), container.end());
return 0;
}
```

Esempio uso contenitori associativi

```
#include <map>
#include <algorithm>
#include <iostream>

int main() {

map<char*,int> amap;

amap.insert( pair<char*,int>("Primo",1));
amap.insert( pair<char*,int>("Secondo",2));
cout << "Size : " << amap.size() << endl;
amap["Terzo"]=3;
amap["Quarto"]=4;
cout << "Size : " << amap.size() << endl;

map<char*,int>::iterator it;

for ( it=amap.begin(); it!=amap.end(); it++)
  cout << "map : " << it->first << " " << it->second << endl;
cout << amap.find("Terzo")->second << endl;

return 0;
}
```