

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Standard Template Library (STL)

Intro, sequenze e iteratori

Stefano Ghidoni



Agenda

- Gestire serie di dati
- STL: filosofia di progettazione
- Sequenze
- Iteratori
- Algoritmi e collezioni di dati



Operazioni sui dati

- Gestire una serie di dati implica numerose operazioni
- Inserire i dati in contenitori
 - vector, list, array, ...
- Leggere i dati
 - Stampa, accesso
- Trovare i dati
 - Da un indice, da un valore, da una proprietà
- Modificare un contenitore
 - Aggiungere, rimuovere, ordinare i dati



Operazioni sui dati

- Esempi:
 - Trovare la temperatura più alta
 - Trovare i valori più grandi di 8800
 - Trovare le differenze tra due sequenze
 - Calcolare la somma degli elementi
- Queste operazioni prescindono dalla descrizione di come i dati sono memorizzati

Un algoritmo è concepibile senza conoscere la struttura dati dove sono memorizzati i dati su cui operare. Ciò è fondamentale.



- STL: una parte della libreria standard che fornisce:
 - Contenitori
 - vector
 - list
 - map
 - ...
 - Algoritmi generici
 - sort
 - find
 - accumulate



STL e dati

- I contenitori STL sono molti, ma hanno **un accesso unificato** *L'accesso a ogni contenitore avviene allo stesso modo.*
 - Modificare il tipo di un contenitore non deve avere impatto sull'algoritmo
- STL ci permette di scrivere codice aggiuntivo solo quando vogliamo implementare funzioni aggiuntive
 - Non quando vogliamo semplicemente cambiare contenitori
- Cosa serve per ottenere questo?



STL – caratteristiche

- Accesso unificato ai dati
 - Indipendente da come sono salvati
 - Indipendente dal tipo usato
- Accesso safe
- Facilità di visita alle strutture dati
- Salvataggio compatto dei dati
- Aggiunta, rimozione e ricerca veloci
- Disponibilità degli algoritmi più comuni

Sequenze e iteratori



Sequenze

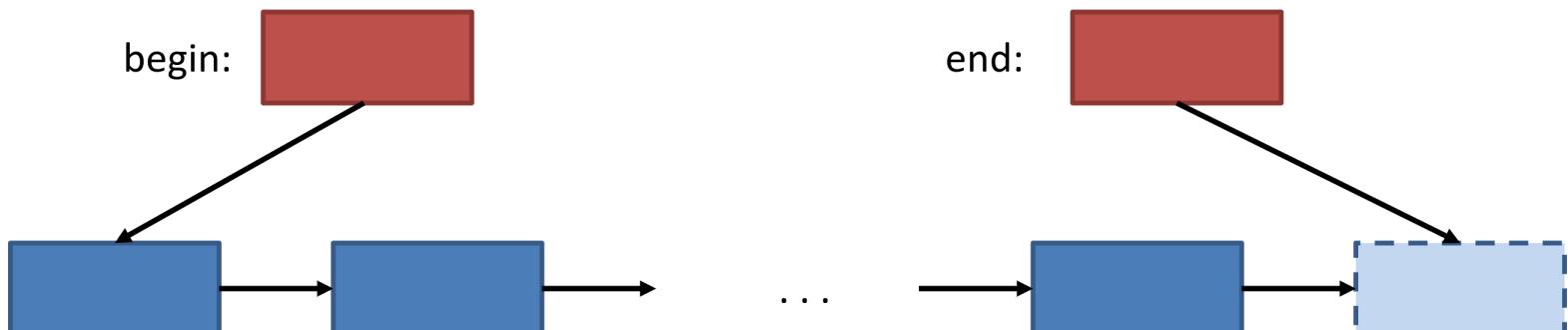
- Un concetto fondamentale di STL è la *sequenza*
- Una sequenza ha un inizio e una fine
- Una sequenza è gestita tramite iteratori
 - Un iteratore è un oggetto che identifica un elemento di una sequenza
 - Cosa ricorda?
 - puntatori
 - indice (es: di array)

→ Viene definito all'interno del relativo contenitore
Viene usato un pattern di polimorfismo statico (spiegato in seguito)



Iteratori

- Iteratore begin: primo elemento
- Iteratore end: primo elemento dopo l'ultimo
- Sequenza: [begin, end)
Questa scelta permette di effettuare cicli in maniera più semplice





Iteratori

- Un iteratore punta a un elemento della sequenza (o a uno dopo l'ultimo)
- È possibile confrontare due iteratori
 - Operatori: == e !=
- È possibile riferirsi agli oggetti puntati usando l'operatore unario * (dereference)
Funziona in maniera analoga rispetto ai puntatori
- È possibile ottenere l'elemento successivo usando ++

Disaccoppiamento
dati/algoritmi



Algoritmi e collezioni di dati

- Per scrivere software di valenza generale è opportuno separare due elementi fondamentali:
 - Collezioni di dati
 - Algoritmi
- Questi elementi **interagiscono spesso**
- L'interazione deve essere gestita con un'**interfaccia** il più possibile **standard**
- STL fornisce questa interfaccia standard



- Disaccoppiamento lato algoritmo: non è necessario conoscere i dettagli della struttura dati; è sufficiente usare i relativi iteratori
 - Es: un algoritmo di ordinamento può operare pur senza conoscere i dettagli della struttura dati
- Disaccoppiamento lato struttura dati: non è necessario esporre tutti i dettagli e gestire tutti i modi d'uso; è sufficiente implementare ed esporre gli iteratori
 - Es: un vector non ha necessità di conoscere l'algoritmo che opera su di esso per permettergli di funzionare; è sufficiente che esponga gli iteratori

Questo non vuol dire che non ci siano avere altre tipologie di accesso a un costruttore (se presenti, non sono unificate).



Esempio

```
// Restituisce l'elemento in [first, last) che ha
// il valore più alto

template<typename Iterator>
Iterator high(Iterator first, Iterator last)
{
    Iterator high = first;
    for (Iterator p = first; p != last; ++p)
        if (*high < *p)
            high = p;

    return high;
}
```

Non significa più "spostati a destra di uno slot", ma "percorri il puntatore al prossimo slot".

È un algoritmo semplice, che permette di implementare un algoritmo (stile array) in maniera generale.

Funziona in maniera disaccoppiata dal suo iteratore. Ogni operatore implementato, è implementato in maniera unbounded.



UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Standard Template Library (STL) Intro, sequenze e iteratori

Stefano Ghidoni