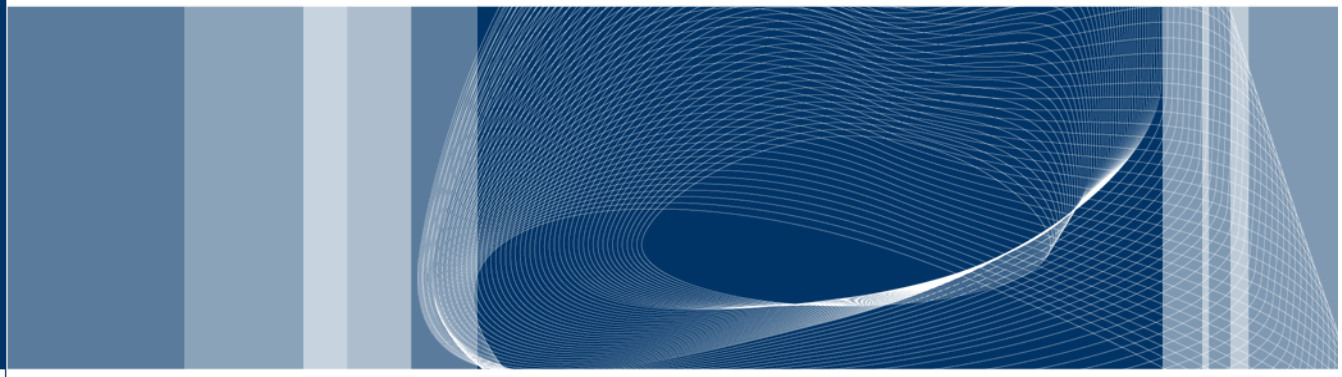
POLITECNICO DI MILANO



Algoritmi e Strutture Dati

Laboratorio 24/11/2008

Daniele Loiacono

STL: list

- La classe **list** è un template che consente di memorizzare un insieme di elementi in una lista puntata (linked list)

```
list<tipo> the_list;
```

- Funzioni principali
  - `void clear()`: elimina tutti gli elementi della lista
  - `bool empty()`: ritorna true se la lista è vuota
  - `void push_back( const tipo& val )`: aggiunge un elemento in fondo
  - `void push_front( const tipo& val )`: aggiunge un elemento in testa
  - `void pop_back()`: rimuove l'elemento in fondo
  - `void pop_front()`: rimuove l'elemento in testa
  - `size_type size()`: restituisce la dimensione del vettore

STL: list (2)

```
#include <cstdlib>
#include <iostream>
#include <list>
using namespace std;
int main(int argc, char *argv[])
{
    list<int> l;
    list<int>::iterator i;
    int j;
    for (j=0; j<10; j++)
        l.push_back(j);
    for (i=l.begin(); i!=l.end(); i++)
        cout << *i << endl;
}
```

Esercizio 1

- Scrivere il template di una classe che implementi con una **list** una coda derivata dalla seguente classe virtuale astratta

```
template <class Elem> class CodaBase {
public:
   virtual bool enque(Elem) = 0;
   virtual bool deque() = 0;
   virtual Elem head() = 0;
   virtual bool isempty() = 0;
};
```

Esercizio 1

- Verificare la classe implementata usando il seguente frammento di codice:

```
CodaList<int> c;

for(i=0;i<10;i++) c.enqueue(rand());

for(i=0;i<10;i++) {
  cout << c.head();
  c.dequeue();
}
```

Code con priorità

- Nelle code con priorità gli elementi sono associati ad un valore di priorità
- Nell'operazione di rimozione viene data la precedenza agli elementi con maggiore priorità
- Tra gli elementi con medesimo valore di priorità il funzionamento è identico alla politca FIFO utilizzata per le code senza priorità

Esercizio 2

- Scrivere il template di una classe che implementi con una una coda con priorità derivata dalla seguente classe virtuale astratta

```
template <class Elem> class CodaBasePriorita {
public:
   virtual bool enque(Elem e, int priorita) = 0;
   virtual bool deque() = 0;
   virtual Elem head() = 0;
   virtual bool isempty() = 0;
};
```

- Utilizzare una list per rappresentare i dati
- Suggerimento: implementare il metodo enqueue in modo che l'inserimento sia ordinato rispetto alla priorità