Creazione di Librerie e Compilazione Separata: make

Informatica - CdL Triennale in Fisica

Carlo Mereghetti

`Sito del corso:  mereghetti.di.unimi.it/inf`

`Pagina personale: mereghetti.di.unimi.it`

Creazione di librerie di funzioni

intestazione

```
#include  ...
using namespace std;
struct { ... };

... f1( ... );
... f2( ... );
... f3( ... );
```

main

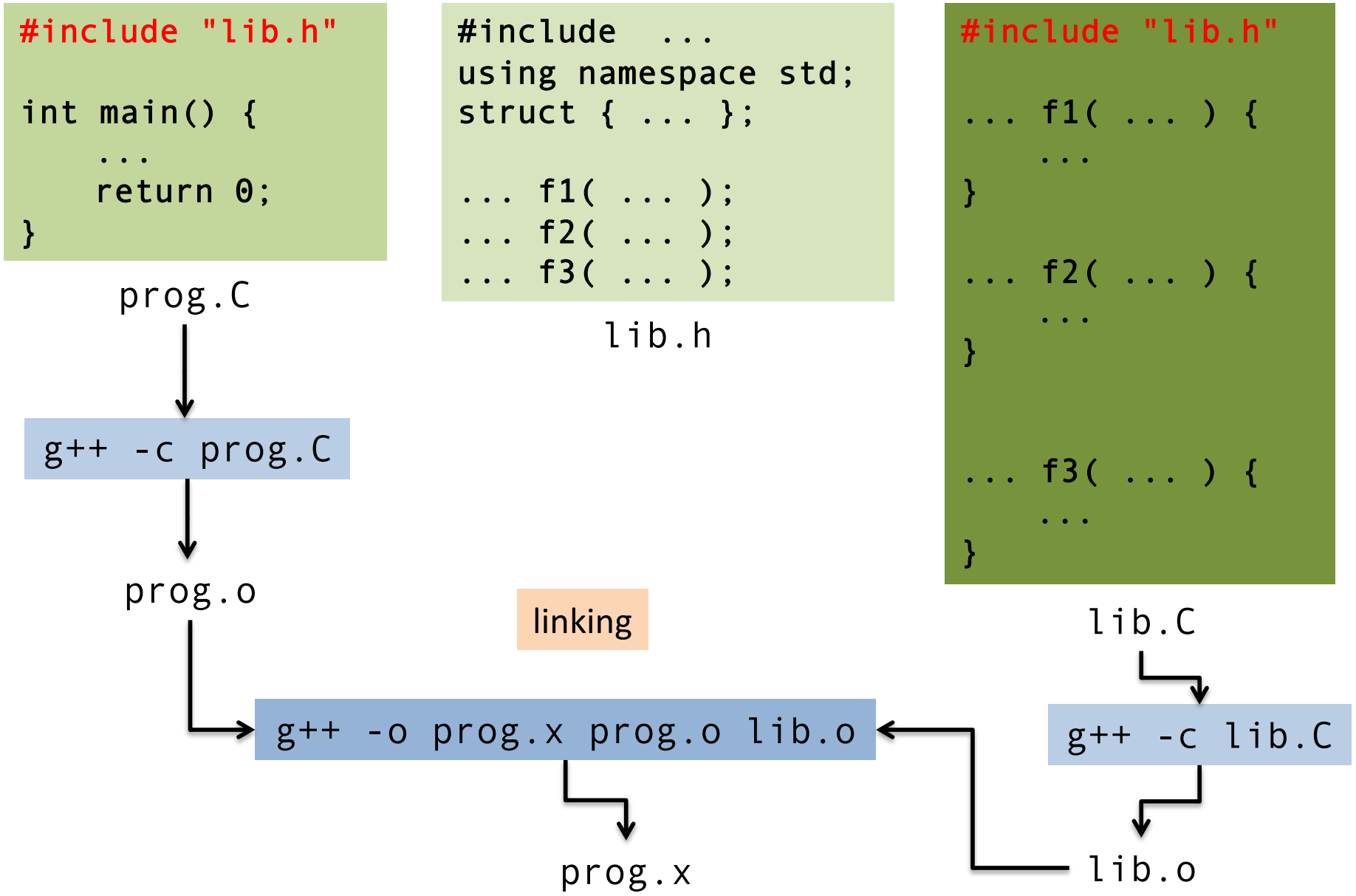
```
int main() {
    ...
    return 0;
}
```

funzioni

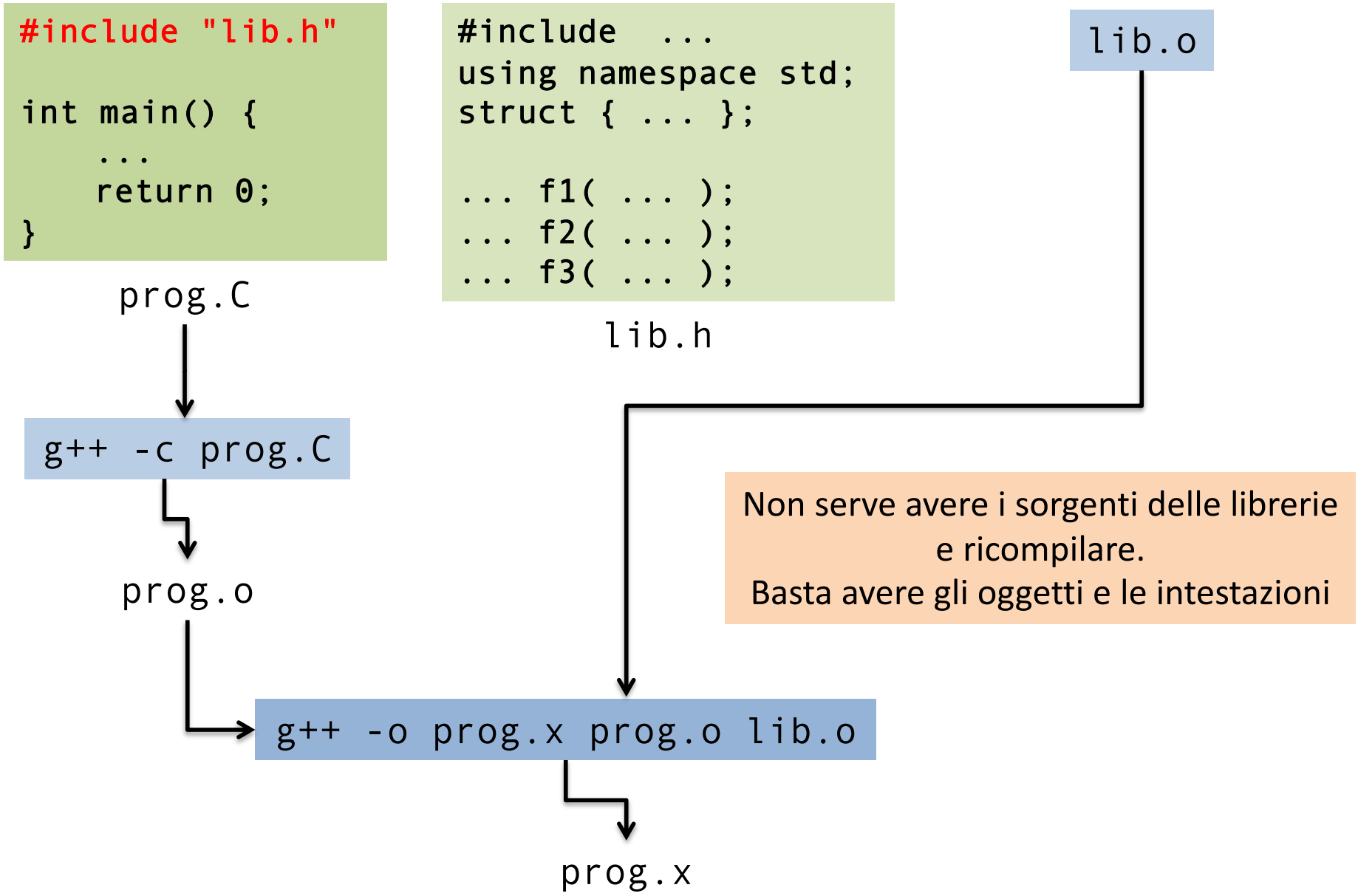
```
... f1( ... ) {
    ...
}

... f2( ... ) {
    ...
}


... f3( ... ) {
    ...
}
```

- Collezione di funzioni che potrebbe essere interessante anche per altri programmi. Esportabile.
- Perché non creare una libreria separata, riutilizzabile senza doverla riscrivere, ma solo richiamandone le funzioni nei main?

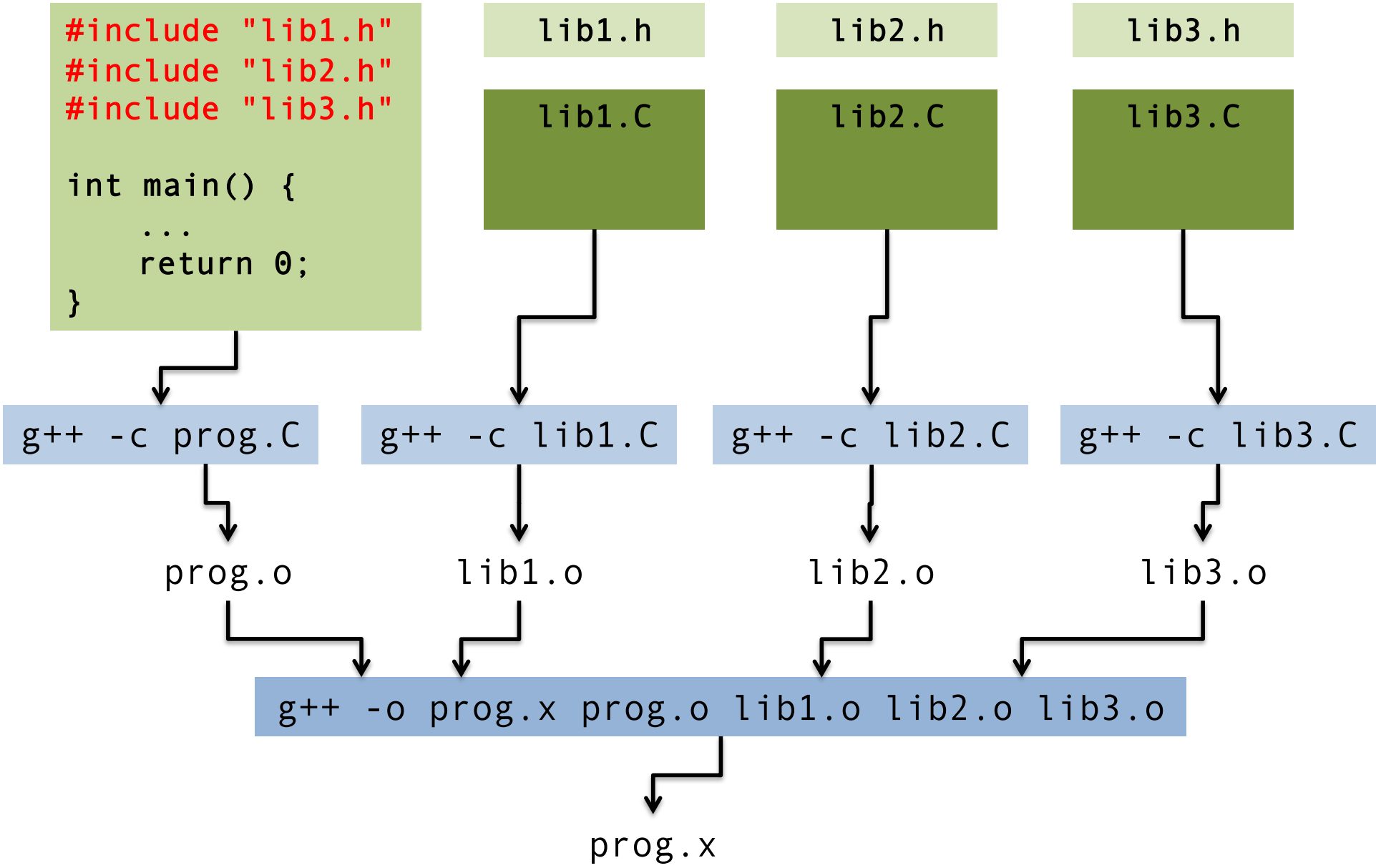
Creazione e utilizzo di librerie di funzioni



Utilizzo di librerie di funzioni oggetto



Strutturazione in varie librerie e loro utilizzo



Il nostro progetto sui **complessi**

```
#include <fstream>
#include <iostream>
#include <cmath>

using namespace std;

struct complesso {
	float r;
	float i;
};

complesso coniugato( complesso );
complesso somma( complesso, complesso );
double modulo( complesso );
void stampa( complesso );
complesso prodotto( complesso, complesso );

int main() {
	...
}

// Funzioni sui complessi

complesso coniugato( complesso a ) {
	...
}

complesso somma( complesso a, complesso b ) {
	...
}

complesso prodotto( complesso a, complesso b ) {
	...
}

double modulo( complesso a ) {
	...
}

void stampa( complesso a ) {
	...
}
```

complessi.C

```
#include "cpl.h"

int main() {
	...
}
```

cpl.h

```
#include <fstream>
#include <iostream>
#include <cmath>

using namespace std;

struct complesso {
	float r;
	float i;
};

complesso coniugato( complesso );
complesso somma( complesso, complesso );
double modulo( complesso );
void stampa( complesso );
complesso prodotto( complesso, complesso );
```

cpl.C

```
#include "cpl.h"

complesso coniugato( complesso a ) {
	...
}

complesso somma( complesso a, complesso b ) {
	...
}

complesso prodotto( complesso a, complesso b ) {
	...
}

double modulo( complesso a ) {
	...
}

void stampa( complesso a ) {
	...
}
```

Gestione di progetti complessi: **make** e **makefile**

Automatizzare la compilazione di programmi che usano librerie: `make`

Poniamo di avere un programma con main in `prog.C` e che usa le librerie `lib1.C, lib2.C, lib3.C` con relativi file di intestazione `lib1.h, lib2.h, lib3.h`

<TAB>

```
prog.x: prog.o lib1.o lib2.o lib3.o
     g++ -o prog.x prog.o lib1.o lib2.o lib3.o

prog.o: prog.C lib1.h lib2.h lib3.h
     g++ -c prog.C

lib1.o: lib1.C lib1.h
     g++ -c lib1.C

lib2.o: lib2.C lib2.h
     g++ -c lib2.C

lib3.o: lib3.C lib3.h
     g++ -c lib3.C

compila: prog.x

esegui:
     ./prog.x

clean:
     rm *.o
```

`makefile`

Esplicito le dipendenze tra i file che costituiranno il programma finale e le azioni per ottenerli

Lanciando il comando

```
$> make
```

compilo solamente i file modificati

```
$> make <etichetta>
```

Lancio `make` a partire dall'etichetta