



Programmazione modulare nel linguaggio C

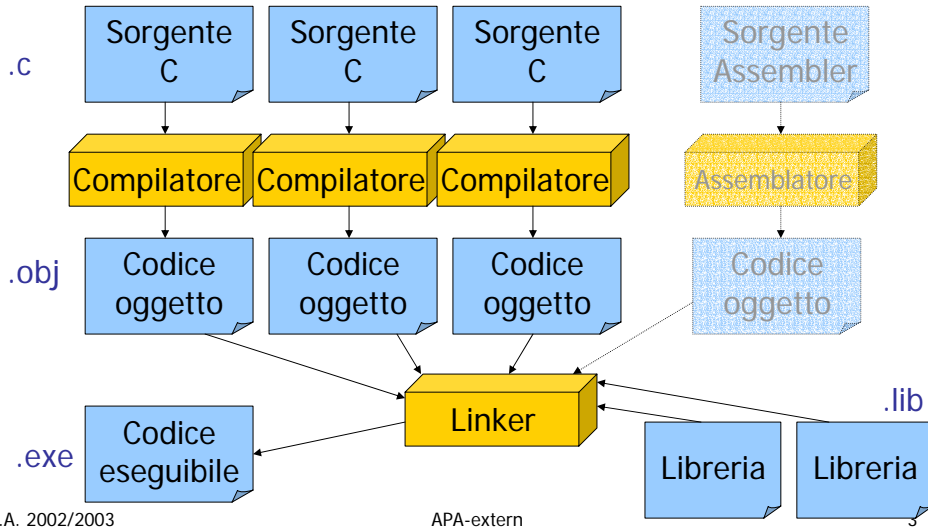


Fulvio CORNO - Matteo SONZA REORDA
Dip. Automatica e Informatica
Politecnico di Torino

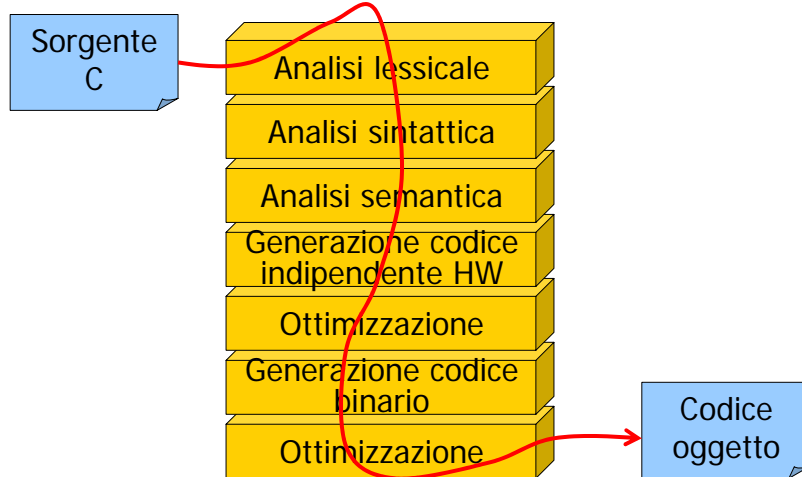
Introduzione

- Non è possibile creare programmi di elevata complessità lavorando con un unico file sorgente
 - Lentezza di ricompilazione
 - Difficile riuso di procedure
 - Impossibile collaborare tra più programmatori
 - Difficile tener traccia delle modifiche
- Occorre realizzare programmi distribuiti su più file sorgenti.

Compilazione e Link



Struttura del compilatore



Modulo di Codice oggetto

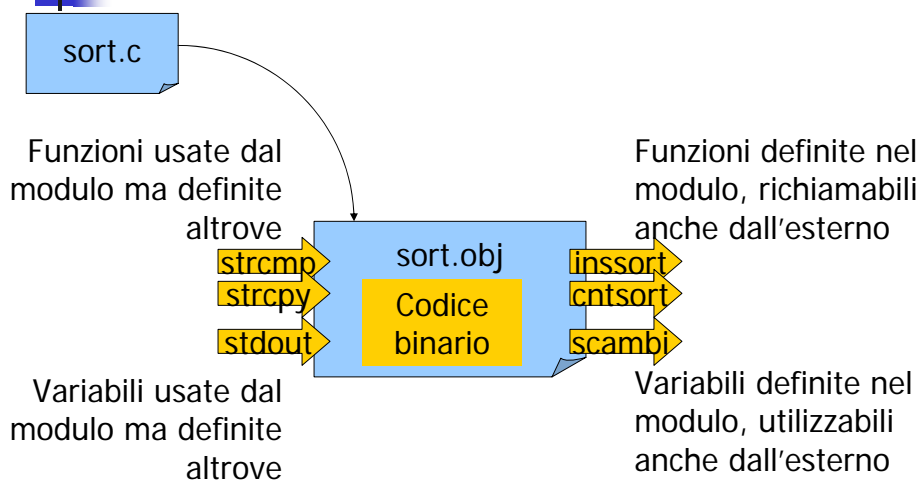
- Contiene il codice binario corrispondente ad un *modulo C*
- Contiene dei *riferimenti esterni* a funzioni e variabili dichiarate in altri moduli
- Contiene funzioni e variabili utilizzabili da altri moduli
- Uno ed un solo modulo contiene main().

A.A. 2002/2003

APA-extern

5

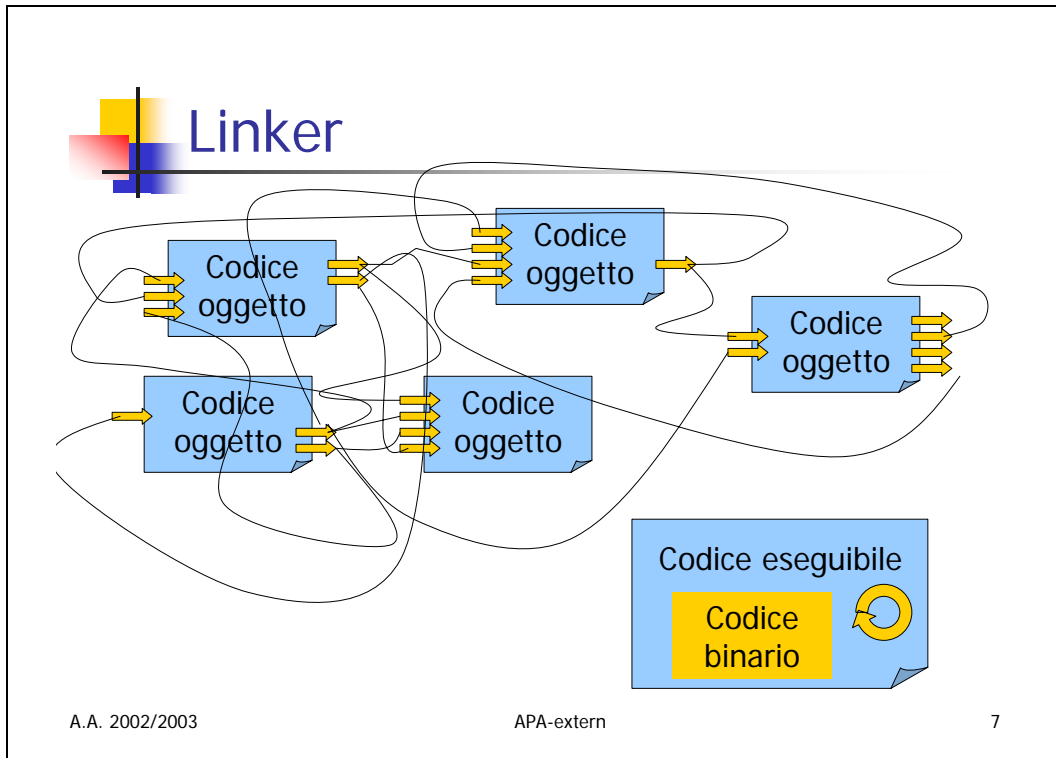
Modulo oggetto



A.A. 2002/2003

APA-extern

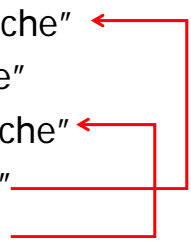
6



Modularità

Ciascun modulo in C:

- Definisce alcune funzioni "private"
- Definisce alcune funzioni "pubbliche"
- Definisce alcune variabili "private"
- Definisce alcune variabili "pubbliche"
- Chiama alcune funzioni "esterne"
- Usa alcune variabili "esterne".



A.A. 2002/2003

APA-extern

8

Chiamata di funzioni (I)

```
void InsertionSort(int A[], int n) ;  
void InsertionSort(int A[], int n)  
{  
    int i, j, key ;  
    /* ... */  
}
```

```
extern void InsertionSort(int A[], int n) ;  
/* ... */  
main()  
{  
    int v[10] ;  
    /* ... */  
    InsertionSort(v, 10) ;  
    /* ... */  
}
```

A.A. 2002

9

Chiamata di funzioni (II)

- Il modulo che *esporta* una funzione non deve fare nulla di particolare: in C, tutte le funzioni sono pubbliche (per default)
- Il modulo che utilizza la funzione deve dichiararne il *prototipo* (la parola chiave **extern** è opzionale)
- Il linker farà il collegamento.

A.A. 2002/2003

APA-extern

10

Funzioni private

- Nel caso in cui un modulo definisca delle funzioni che NON vuole esportare, nella definizione (non nel prototipo) deve comparire la parola chiave **static**.

```
void InsertionSort(int A[], int n) ;  
void CancellaVettore(int A[], int n) ;  
  
void InsertionSort(int A[], int n)  
{ . . . }  
static void CancellaVettore(int A[], int n)  
{ . . . }
```

A.A. 2002/2003

APA-extern

11

Variabili globali (I)

```
int n_elem ;  
double vett[MAX] ;  
  
static int temp ;
```

```
extern int n_elem ;  
extern double vett[MAX] ;  
/* ... */  
{ for(i=0; i<n_elem; ++i)  
    vett[i]++ ;
```

A.A. 2002/2003

APA-extern

12

Variabili globali (II)

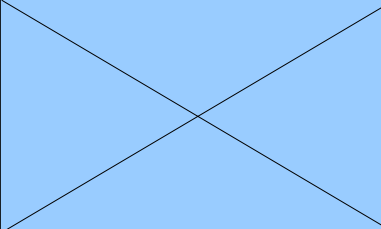
- Il modulo che *esporta* una variabile globale non deve fare nulla di particolare: in C, tutte le variabili globali sono pubbliche (per default)
- Il modulo che utilizza la variabile deve ridefinirla con la parola chiave **extern**
- Se non si vuole esportare una variabile occorre definirla con la parola chiave **static**
- Le variabili locali (alle procedure) non possono essere condivise.

A.A. 2002/2003

APA-extern

13

Riassunto: funzioni


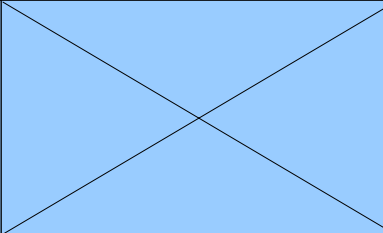
<pre>int f(int x) ; int f(int x) { . . . }</pre>	<pre>int f(int x) ; { . . . y = f (z) ; . . . }</pre>
<pre>int f(int x) ; static int f(int x) { . . . }</pre>	

A.A. 2002/2003

APA-extern

14

Riassunto: variabili

<pre>int x ; { . . . x = 18 ; . . . }</pre>	 <pre>extern int x ; { . . . x = x - 38 ; . . . }</pre>
<pre>static int x ; { . . . x = 18 ; . . . }</pre>	

A.A. 2002/2003


APA-extern

15

Problemi

- Le due dichiarazioni (nel modulo che esporta e nei moduli che usano la funzione/variabile) devono coincidere al 100%

<pre>int f(int x) ; int f(int x) { . . . }</pre>	<pre>int f(double x) ; { . . . x = f (z) ; }</pre>
---	---



A.A. 2002/2003

APA-extern

16

Il ruolo degli header file

- Si possono utilizzare degli header file (file .h) per raccogliere le definizioni "extern" di funzioni e variabili pubbliche
- Il creatore del modulo che esporta le funzioni o variabili crea un file .h
- Tutti i moduli che usano tali risorse esportate includono i file .h (#include).

A.A. 2002/2003

APA-extern

17

Esempio di header file (I)

```
void InsertionSort(int A[], int n) ;  
extern int n_elem ;  
extern double vett[MAX] ;
```

← sort.h

sort.c ↓

```
#include "sort.h" /* per controllo coerenza */  
int n_elem ;  
double vett[MAX] ;  
static int temp ;  
void InsertionSort(int A[], int n)  
{ . . . }  
static void CancellaVettore(int A[], int n) ;  
{ . . . }
```

A.A. 2002/2003

APA-extern

18

Esempio di header file (II)

```
void InsertionSort(int A[], int n) ;  
extern int n_elem ;  
extern double vett[MAX] ;
```

← sort.h

altro.c ↓

```
#include "sort.h" /* per importare dichiarazioni */  
main()  
{  
    int v[10] ;  
    /* ... */  
    InsertionSort(v, 10) ;  
    /* ... */  
    for(i=0; i<n_elem; ++i)  
        vett[i]++ ;  
}
```

A.A. 2002/2003

APA-extern

19

Note

- Ciascun modulo solitamente esporta alcune funzioni o variabili, e ne importa altre
- È sempre bene che ciascun modulo includa anche il *proprio* header file
- Conviene limitare al minimo il numero di variabili condivise
- Includere anche le #define nei file .h
- Gruppi di moduli correlati tra loro possono condividere un solo file .h.

A.A. 2002/2003

APA-extern

20