Compilazione separata in C

Barbara Guidi & Roberta Gori

Dipartimento di Informatica
Università di Pisa

Corso Informatica I - 2012/2013

Suddivisione del codice

Tutti i programmi che abbiamo visto finora erano composti da un unico file .c
Nel caso di programmi più grandi è conveniente suddividere il codice in più file:

- migliore suddivisione e organizzazione del codice
- facilità di manutenzione e correzione degli errori
- possibilità di riutilizzare il codice per progetti diversi

Il linguaggio C e l'ambiente Unix forniscono tre strade per suddividere il codice:

1. inclusione diretta di file sorgente e compilazione unica;
2. inclusione di header file e compilazione separata;
3. creazione di librerie statiche o dinamiche.

Header file e compilazione separata

Anziché includere tutto il sorgente si può includere un header file:

- è un file con estensione .h
- contiene solamente: inclusioni di altri header file definizione di strutture e di tipi (typedef) prototipi di funzioni
- si usa la direttiva `#include "mylib.h"`

Il codice delle funzioni va scritto in un file .c separato:

- che deve includere l'header!

Compilazione e linking devono essere fatti separatamente:

- `gcc -c mylib.c`
- `gcc -c main.c`
- `gcc -o main main.o mylib.o`

La modifica di uno dei file richiede di ricompilare il file oggetto corrispondente e di rifare il linking.

Compilazione separata (1)

La compilazione è infatti formata da (almeno) due fasi diverse:

- la compilazione vera e propria e il linkaggio.
- Il risultato della compilazione è un file oggetto, binario ma non ancora eseguibile
  - Eseguendo: `gcc -c esempio_comp.c` si ha come risultato l'ouput del preprocessore (**"modulo oggetto"** o **"object file"**) che tipicamente ha estensione .o
- Il linkaggio è l'operazione che "aggangia" a quel file oggetto le librerie di sistema che lo rendono eseguibile sul pc
  - Si collegano insieme piu moduli oggetto per creare un le eseguibile;
  - Eseguendo `gcc esempio_comp.o` il modulo oggetto verra trasformato in un file eseguibile;

Compilazione separata (2)

File: somma.c

```
#include "somma.h"
int somma (int addendo1, int addendo2)
{
    return addendo1 + addendo2;
}
```

File: somma.h

```
int somma (int addendo1, int addendo2);
```

File: main.c

```
#include "somma.h"
main() {
    int a, b, c;
    scanf("%d %d", &a, &b);
    c = somma(a,b);
    printf("%d\n", c);
}
```

Compilazione separata (3)

Come compiliamo?

- `gcc -c somma.c`
- `gcc -c main.c`
- `gcc -o main main.o somma.o`