

# Creazione di librerie statiche e dinamiche in programmi C

# Tipi di librerie

- Una libreria è un file archivio contenente codice compilato che verrà agganciato al programma nella fase di linking
- Le librerie sono utilizzate prevalentemente per:
  - Ridurre i tempi di compilazione
  - Diminuire la size dei programmi
  - Diminuire l'occupazione di memoria in programmi che utilizzano la stessa libreria (per librerie dinamiche)
- Ci sono 2 tipi di librerie:
  - Statiche (estensione “.a”)
  - Condivise o Dinamiche (estensione “.so”)

# Librerie statiche (.a)

- La libreria è agganciata al programma all'atto della compilazione per formare un eseguibile monolitico.
  - Creazione dei file oggetto:

```
gcc -std=c99 -Wall myfunc.c -c -o myfunc1.o
```

```
gcc -std=c99 -Wall myfunc2.c -c -o myfunc2.o
```


```
gcc -std=c99 -Wall errfunc.c -c -o errfunc.o
```
  - Creazione della libreria statica:

```
ar rvs libmyf.a myfunc1.o myfunc2.o errfunc.o
```
  - Linking e creazione dell'eseguibile:

```
gcc main.c -o myprog -L . -lmyf
```

oppure

```
gcc main.c -o myprog $pwd/libmyf.a
```



# Librerie Condivise (.so)

- La libreria è agganciata al programma in 2 fasi. A tempo di compilazione il *linker* verifica che tutti i simboli siano definiti; nella fase di caricamento il *loader* carica le librerie agganciate al programma (quelle necessarie).
  - Creazione dei file oggetto:  
gcc -std=c99 -Wall myfunc.c -c -fPIC -o myfunc1.o  
...  
• Creazione della libreria dinamica:  
gcc -shared -o libmyf.so myfunc1.o ....  
• Linking e creazione dell'eseguibile:  
gcc main.c -o myprog -L . -lmyf

# -fPIC

- L'opzione PIC (anche -fpic) dice al compilatore di generare Position Independent Code
- Il codice generato è indipendente dall'indirizzo virtuale in cui verrà caricato in memoria.
- E' necessario per costruire librerie condivise
- L'indirizzo iniziale dove viene caricata la libreria condivisa dipende da molti fattori che non possono essere conosciuti staticamente (memoria utilizzata, numero di altre librerie condivise presenti, ...)

# Loading di librerie condivise

- Lanciamo il programma `$> ./myprog`

`./myprog: error while loading shared libraries: libmyf.so:  
cannot open shared object file: No such file or directory`

- `$> ldd ./myprog` **(vedere man 1 ldd)**

`linux-vdso.so.1 => (0x00007fff597fe000)`

`libmyf.so => not found`

`libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f4cef03b000)`

`/lib64/ld-linux-x86-64.so.2 (0x00007f4cef42a000)`

- Per dire al dynamic linker/loader dove trovare le librerie si può usare

`export LD_LIBRARY_PATH="./:${LD_LIBRARY_PATH}"`

di default il dynamic linker cerca nei path specificati in  
`/etc/ld.so.conf*` ed in `/lib /usr/lib` **(vedere man ldconfig)**

- Si può anche usare l'opzione del loader **-rpath**

`gcc main.c -o main -Wl,-rpath,./ -L. -lmyf`

Qui in generale metto  
il path dove saranno  
le mie librerie