

Funzioni rientranti

Concetto di funzione rientrante

- Una funzione si dice *rientrante* (*reentrant function* in Inglese) se la sua esecuzione può essere interrotta in qualsiasi punto e la funzione rieseguita senza che questo produca effetti indesiderati (i.e. risultati scorretti o comportamento diverso rispetto all'output ottenuto se la funzione non viene mai interrotta).
- La rientranza è un concetto importante nella programmazione single-threaded (sequenziale). In particolare, è un concetto rilevante per la corretta implementazione di *signal handlers* (in questo caso, come vedremo più avanti nel corso, si parla più correttamente di *asynchronous signal safe functions*).
- Non sono rientranti quelle funzioni che modificano variabili locali statiche (stato interno), oppure che modificano variabili globali, oppure che chiamano funzioni non rientranti.
- Alcune funzioni della libreria standard non sono rientranti, hanno però la corrispondente versione rientrante (indicata con `_r`), ad esempio `strtok` (`strtok_r`), `readdir` (`readdir_r`), `rand` (`rand_r`).

Esempio

- La funzione F non è rientrante. Se viene interrotta dopo aver eseguito l'istruzione alla riga 3 e viene rieseguita produrrà un output non aspettato al termine della prima chiamata.

```
1. int x=0;  
2. int F(int a) {  
3.     x+=a;  
4.     return x;  
5. }
```

- Altro esempio di funzione non rientrante, e la corrispondente versione rientrante

```
1. void swap(int *a, int *b) {  
2.     static int tmp;  
3.     tmp = *a;  
4.     *a = *b;  
5.     *b = tmp;  
6. }
```

```
1. void swap_r(int *a, int *b) {  
2.     int tmp;  
3.     tmp = *a;  
4.     *a = *b;  
5.     *b = tmp;  
6. }
```

Tokenizzazione di stringhe con strtok

- La funzione di libreria strtok, che serve per tokenizzare una stringa, non è rientrante perchè utilizza “uno stato interno” per effettuare la tokenizzazione. Il seguente frammento di codice mostra un suo possibile utilizzo:

```
1. char stringa[] = "Hello crazy world";
2. char *token=strtok(stringa," "); // inizializza lo stato interno, inoltre modifica la stringa
3. while(token) {
4.     printf("%s\n", token);
5.     token=strtok(NULL, " ");
6. }
```

- La versione rientrante è strtok_r che prende un ulteriore parametro utilizzato per memorizzare lo stato che quindi non è più “interno”.

```
1. char stringa[] = "Hello crazy world";
2. char *save = NULL,
3. char *token=strtok_r(stringa," ", &save); // Attenzione, l'argomento stringa viene modificato!
4. while(token) {
5.     printf("%s\n", token);
6.     token=strtok(NULL, " ", &save);
7. }
```

Considerazioni

- La seguente funzione F non è rientrante perchè utilizza uno stato interno (la variabile con modificatore static dichiarata a riga 2). F però può essere eseguita concorrentemente da due thread di controllo distinti. In altri termini la funzione è *thread-safe* perchè accede in scrittura al suo stato (x) in mutua esclusione, cioè la sezione di codice tra le istruzioni 3 e 5 viene eseguita sempre da un solo thread alla volta.

```
1. int incX(int a) {  
2.     static int x=0;  
3.     Lock();  
4.     int t = (x+=a);  
5.     Unlock();  
6.     return t;  
7. }
```

- Attenzione quindi che thread-safe function and reentrant function sono due concetti differenti. Il primo si utilizza in un contesto multi-threaded in cui le funzioni sono eseguite concorrentemente (o in parallelo) da più threads di controllo. Il secondo è un concetto che ha senso per il singolo thread di controllo.