Contents

Puntatori a funzioni

Una funzione, viene identificata dall'indirizzo in memoria dove è iniziano le istruzioni che la funzione eseguirà. Le funzioni quindi possono essere passate come puntatori.

Sintassi

```
int (*compare)(int x, int y);
```

Una funzione del genere ha 3 caratteristiche:

- Si chiama compare
- Ritorna un int
- Prende come parametri x e y

Può essere richiamata come `compare(x,y)` oppure `(*compare)(x,y)`

Puntatori void *

Un puntatore di tipo `void *` è un puntatore ad uso generico, può rappresentare qualsiasi tipo di puntatore. Lo spazio occupato da questo puntatore è lo stesso di un qualsiasi puntatore(contiene solo l'indirizzo, quindi 4 byte per 32-bit e 8 byte per 64-bit). Tuttavia il compilatore non sa cosa c'è dentro a `void *` e quindi neanchè la dimensione degli elementi.

Questi tipi di puntatori non si possono dereferenziare, infatti vanno trasformati nel tipo desiderato attraverso il casting e poi usati come dei normali puntatori a tipo.

Questi tipi di puntatori sono utili per accettare funzioni di qualsiasi tipo, ad esempio:

```
bool sort((*int compare)(void *arr, void *x, void *y)) {
    for(int i = 0; i < size-1; i++) {
        if(!compare(arr[i], arr[i+1])) {
            return 0;
        };
    }
};
```

Ad esempio una funzione del genere permette di stabilire se un array di qualsiasi tipo è ordinato. Basta cambiare la funzione `compare` che compara tipi di qualsiasi.

Funzioni variadiche

Le funzioni variadiche servono per passare un numero indefinito di parametri ad una funzione.

Sintassi delle funzioni variadiche

```
int printf(char *format, ...);
```

Dove i puntini di sospensione indicano un numero indefinito di parametri di qualsiasi tipo.

I puntini di sospensione vanno usati come ultimo argomento.

Utilizzo delle funzioni variadiche

Per prendere gli argomenti passati alla funzione dovremo fare le seguenti cose:

- Creare la lista dei parametri di tipo `va_list ap`
- Chiamare la funzione `va_start(ap, num_parametri)` per inializzare la lista di argomenti
- Prendere gli argomenti con `va_arg(ap, tipo_variabile)`. In questo modo prenderemo un argomento dalla lista e verrà fatto il casting al tipo specificato

```
double average(int i, ...) {
    double total = 0; // initialize total
    va_list ap; // stores information needed by va start and va end
    va_start(ap, i); // initializes the va list object
    // process variable-length argument list
    for (int j = 1; j <= i; ++j) {
        total += va_arg(ap, double);
    }
    va_end(ap); // clean up variable-length argument list
```

```
    return total / i; // calculate average
}
```