Mini Guida ai puntatori in C

1. Cos'è un Puntatore?

Un puntatore è una variabile che, invece di contenere un valore diretto (come un numero o un carattere), contiene l'indirizzo di memoria di un'altra variabile.

In parole semplici: se una variabile normale è come una casa che contiene un oggetto, il puntatore è come un biglietto con scritto l'indirizzo di quella casa.

Dichiarazione e Operatori Fondamentali

Per dichiarare un puntatore si usa l'asterisco *:

```
int *ptr; // ptr è un puntatore a un intero
```

Gli operatori chiave sono:

- & (Operatore "indirizzo di"): Restituisce l'indirizzo di memoria di una variabile.
- * (Operatore di dereferenziazione, si legge “puntato”):
  Accede al valore contenuto nell'indirizzo di memoria puntato.

Esempio base:

```
int x = 10;
int *p = &x; // p ora contiene l'indirizzo di x
printf("%d", *p); // Stampa 10 (il valore di x)
```

2. Puntatori e Stringhe (Array di Caratteri)

In C, una stringa non è un tipo di dato primitivo (come int, char, float, …) , ma un array di caratteri terminato dal carattere nullo '\0' (il cui valore ASCII è 0) . Il nome di un array è già di per sé un puntatore al suo primo elemento.

Dichiarare Stringhe con Puntatori

Ci sono due modi principali per lavorare con le stringhe:

```
char nome[] = "Luca"; // Array di caratteri (modificabile)
char *cognome = "Rossi"; // Puntatore a stringa costante (sola lettura)
```

Scorrere una Stringa con l'Aritmetica dei Puntatori

Invece di usare l'indice (es. nome[0]), si può usare il puntatore per spostarsi di carattere in carattere.

```
#include <stdio.h>
int main() {
   char *ptr = "Ciao";

   while (*ptr != '\0') { // Finché non si raggiunge il terminatore
      printf("%c ", *ptr); // Stampa il carattere corrente
      ptr++; // Sposta il puntatore al carattere successivo
   }
   return 0;
}
```

Output: **C i a o**

Esempio Pratico: Lunghezza di una Stringa (strlen fai-da-te)

```c
int mia_strlen(char *s) {
  char *inizio = s; // Salva il punto di partenza
  while (*s != '\0') {
    s++; // Avanza fino alla fine
  }
  return (s - inizio); //La differenza tra puntatori dà il numero di elementi
}
```

Chiamata: mia_strlen("Ciao"); restituisce 4.

Esempio Pratico: Copia di una Stringa (strcpy fai-da-te)

La funzione copia i caratteri dalla sorgente (src) alla destinazione (dest) usando solo puntatori.

```c
void mia_strcpy(char *dest, char *src) {
  while (*src != '\0') {
    *dest = *src; // Copia il carattere
    dest++; // Avanza il puntatore destinazione
    src++; // Avanza il puntatore sorgente
  }
  *dest = '\0'; // Non dimenticare il terminatore!
}
```

3. Riepilogo

- *ptr significa "vai all'indirizzo in ptr e prendi il valore".
- ptr++ non incrementa il valore, ma fa avanzare il puntatore alla cella di memoria successiva (nel caso di char, avanza di 1 byte, di short di 2 byte, etc.).
- (*ptr)++ non incrementa il puntatore, ma il valore dell'oggetto puntato da ptr
- Le stringhe in C sono attraversabili non solo come vettori ma anche con puntatori a char, e la loro fine è sempre determinata dal carattere '\0'.

```c
// Esempio in RAM, puntatore ad int
// Situazione dopo aver definito:
int x = 10;
int *p;
p = &x;
// abbiamo che l'indirizzo della variabile x = 0x100
// l'indirizzo della variabile p = 0x104
// *p (si legge “puntato p”) restituisce 10
```