12.9 Tipi derivati composti tramite puntatore

Il puntatore può essere abbinato a qualsiasi tipo, compreso se stesso. In effetti il puntatore a puntatore – o puntatore di puntatore – è un tipo derivato composto tra i più usati in C. Per esempio:

```
char **pp;
```

è la dichiarazione di un puntatore di puntatore a carattere. Un puntatore di puntatore è una variabile abilitata a mantenere l'indirizzo di una variabile puntatore. Per esempio, nel programma

```
# include <stdio.h>
     void main(void)
     {
        int **ppi;
        int a = 3;
        int *pi;
        char *pausa;

        pi = &a;
        ppi = &pi;
        printf("%d", **ppi);
        gets(pausa);
     }
```

la variabile `pi` contiene l'indirizzo della variabile intera `a`, e `ppi` contiene l'indirizzo di `pi`. Conseguentemente `*ppi` corrisponde al contenuto di `pi`, cioè all'indirizzo di `a`, e `**pp1` corrisponde al contenuto di `a`. Infatti l'istruzione

```
printf("%d", **ppi);
```

visualizza il numero 3 (Figura 12.2).

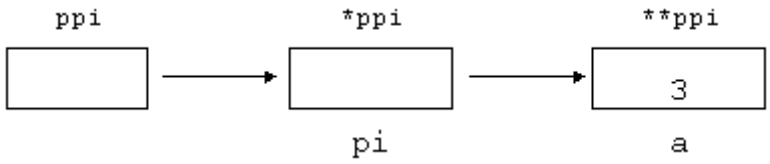

Figura 12.2 Puntatore di puntatore a intero

È naturalmente possibile dichiarare puntatori di puntatori di puntatori. Il programmatore deve però prestare molta attenzione perché una doppia, o peggio ancora, tripla indirezione è difficile da controllare in C; pertanto i puntatori debbono essere usati con parsimonia. Esistono comunque casi in cui è necessario usare una doppia indirezione, cioè un puntatore di puntatore: un tipo derivato composto comunemente usato in C è l'array di puntatori, e per scandire un array di puntatori si fa generalmente uso di un puntatore di puntatore (Listato 12.4).

```
#include <stdio.h>

char *menu[] = {
  "1. Voce di menu 1\n",
  "2. Voce di menu 2\n",
  "3. Voce di menu 3\n",
  ...
  "N. Voce di menu N\n",
  NULL
};

char **ppc = menu;
```

```
main()
{
char *pausa;
while(*ppc!=NULL)
  printf("%s", *ppc++);
gets(pausa);
}
```

Listato 12.4 Un array di puntatori scandito da un puntatore di puntatore

Eseguendo tale programma si ottiene:

```
1.  Voce di Menu 1
     2.  Voce di Menu 2
     3.  Voce di Menu 3
         ...
     N.  Voce di Menu N
```

Come si può osservare, `menu` è un array di puntatori a carattere. Infatti il lettore ricordi che ogni costante stringa riportata tra doppi apici corrisponde all'indirizzo del primo carattere della stringa. Per contrassegnare la fine dell'array di puntatori a carattere `menu` si è usato il puntatore `NULL`. Per visualizzare le varie voci di menu basta passare a `printf` l'indirizzo delle voci di menu raggiungibile tramite `*ppc`. Inoltre `ppc` deve essere incrementato per scandire gli elementi dell'array (Figura 12.3).

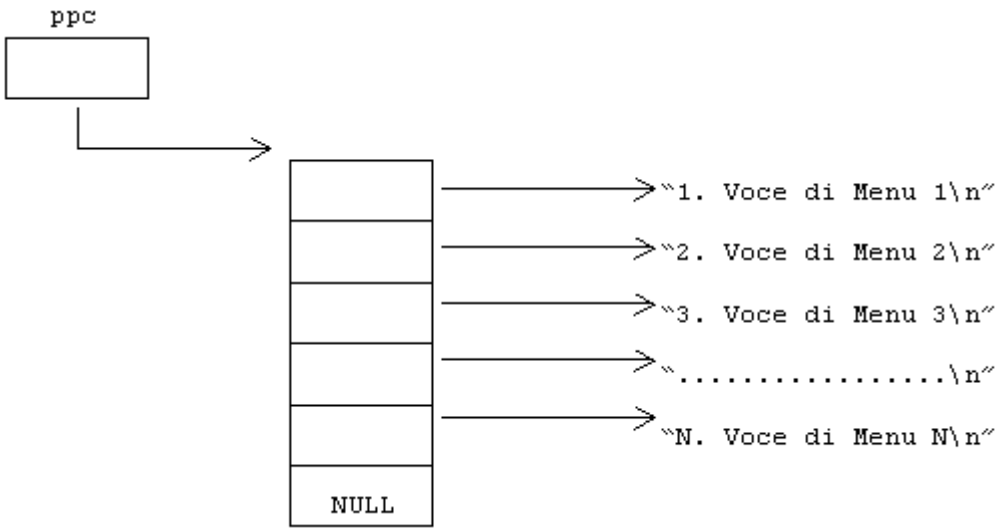

Figura 12.3 Rappresentazione grafica di un array di puntatori

La variabile `menu` è un array di puntatori. La sintassi degli array di puntatori è

`tipo_puntato *nome_array[dim_array];`

Questo tipo derivato composto è particolarmente utile quando si voglia collezionare un insieme di oggetti – in generale – a dimensione variabile, come le stringhe di caratteri. Accanto a un array di puntatori si può trovare un puntatore di puntatore, usato per scandire l'array di puntatori al posto dell'indice dell'array.
Un array di puntatori viene usato anche per passare un numero variabile di parametri alla funzione `main`. Infatti, anche il `main` è una funzione, come abbiamo più volte osservato, e il C mette a disposizione del programmatore un meccanismo attraverso cui passare parametri. Osserviamo il semplice programma:

```
/* prova.c : esempio di programma con parametri sulla linea di
comando                                                          */

     main(int argc, char *argv[])
     {
```

```
    int i;
    for(i=1; i<argc; i++)
       printf(argv[i]);
}
```

I due parametri formali `argc` e `argv` sono parametri standard e sono gli unici ammessi per la funzione `main`. Essi hanno un preciso significato: `argc` è un `int` che contiene il numero di parametri presenti sulla linea di comando, incluso il nome del programma. Per esempio, lanciando il programma `prova` seguito da un insieme di parametri:

```
C:\>prova uno due tre
```

la variabile standard `argc` vale 4 (infatti sono quattro le stringhe presenti sulla linea di comando); `argv` è un array di puntatori a carattere che contiene gli indirizzi delle quattro stringhe `prova`, `uno`, `due`, `tre` (Figura 12.4).

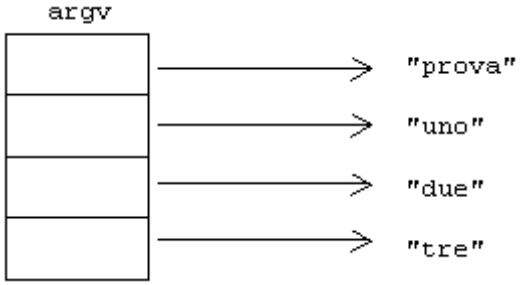

Figura 12.4 Rappresentazione del passaggio di parametri a `main`

Nel programma **`prova.c`** attraverso l'indice `i` si scandisce l'array standard **`argc`** e l'indirizzo delle stringhe `uno`, `due` e `tre` viene iterativamente passato alla funzione `printf`. Dunque lanciando `prova`:

```
C:\>prova uno due tre
```

si ottiene

```
unoduetre
```