# Puntatori e array

## Puntatori

Nel linguaggio C, **un puntatore è una variabile il cui valore è l'indirizzo in memoria di un altro dato**. Per tale ragione, un puntatore è un "riferimento" ad un altro dato. Per dichiarare una variabile puntatore è necessario indicare il tipo dei dati "puntati", ovvero dei dati il cui indirizzo sarà contenuto nella variabile. Dunque una variabile di nome Punt usata per puntare a dati di tipo T va dichiarata tramite l'istruzione

```
T *Punt;
```

(che può anche essere scritta come `T* Punt;` poichè la posizione dello spazio è indifferente). Si dice che la variabile Punt è *di tipo puntatore a T*, o anche "di tipo T*"[1].
Si noti che

```
T A;      /* dichiara la variabile A, di tipo T */
T *B;     /* dichiara la variabile B, di tipo puntatore a T */
T C[3];   /* dichiara l'array C, avente 3 elementi di tipo T */
T *D[7];  /* dichiara l'array D, avente 7 elementi di tipo puntatore a T */
```

Un puntatore può "non puntare a niente". Questa condizione corrisponde ad assegnare alla variabile puntatore il valore NULL. La costante NULL è definita nella libreria *stdlib*: per usare NULL è dunque necessario inserire nel programma la direttiva `#include <stdlib.h>`. Tale direttiva non è necessaria se nel programma è già presente la `#include` di una libreria (ad es. *stdio*) che a sua volta include *stdlib*.

## Puntatori e array

L'aspetto meno ovvio della gestione dei puntatori in C è il rapporto che esiste tra puntatori e array. Tale rapporto, tuttavia, è molto semplice ed è riassunto da questa frase: **in C l'identificatore di un array è, a tutti gli effetti, un puntatore costante[2] alla prima cella dell'array**.
Si consideri un array Arr avente NUM_ELEMENTI di tipo T, dichiarato cioè con

```
T Arr[NUM_ELEMENTI];
```

Ebbene, nel linguaggio C

| | | |
|---|---|---|
| **l'identificativo Arr** | equivale a tutti gli effetti a | **&(Arr[0])** |

Questa unica relazione racchiude TUTTO quello che c'è da sapere sul rapporto tra array e puntatori. Quando occorre, è possibile farne discendere -applicando gli operatori &, * e [ ] nonché l'aritmetica dei puntatori- altre relazioni utili. Ad esempio (sia k un intero qualsiasi):

| | | |
|---|---|---|
| `*Arr` | equivale a | `Arr[0]` |
| `Arr+k` | equivale a | `&(Arr[k])` |
| `*(Arr+k)` | equivale a | `Arr[k]` |

Supponiamo per esempio che Arr sia un array di interi. Le due istruzioni C

```
printf("%d", Arr[0]);
printf("%d", *Arr);
```

sono esattamente equivalenti: entrambe stampano il valore di `Arr[0]`. Allo stesso modo, le due istruzioni

```
scanf("%d", &(Arr[5]));
scanf("%d", Arr+5);
```

sono tra loro esattamente equivalenti: entrambe leggono da tastiera un intero decimale e lo inseriscono in `Arr[5]`.

1 In genere "T*" è pronunciato "T star".
2 Il fatto che il puntatore sia costante significa che non è consentito cambiare il suo valore.