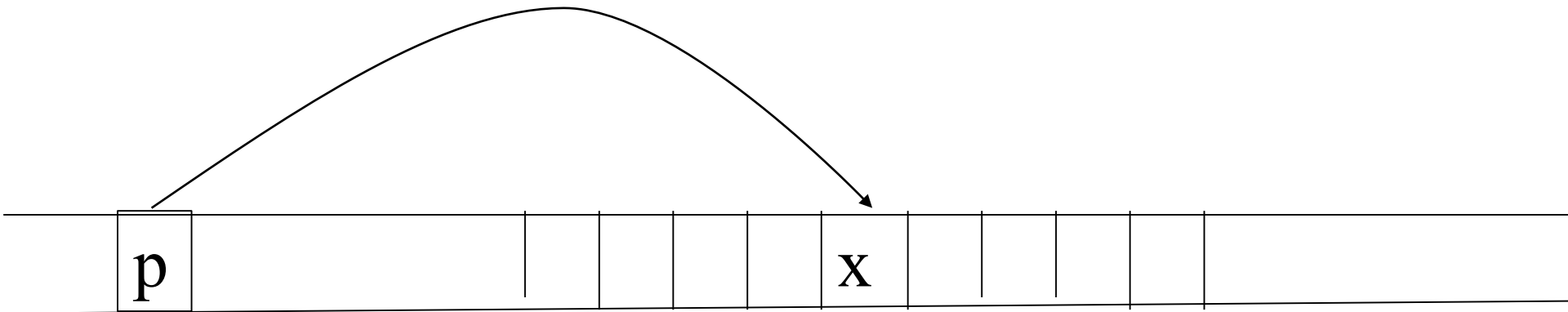


array e aritmetica dei puntatori

5.3 e 5.4 del testo

in C++ array sono puntatori, ma anche puntatori
sono array

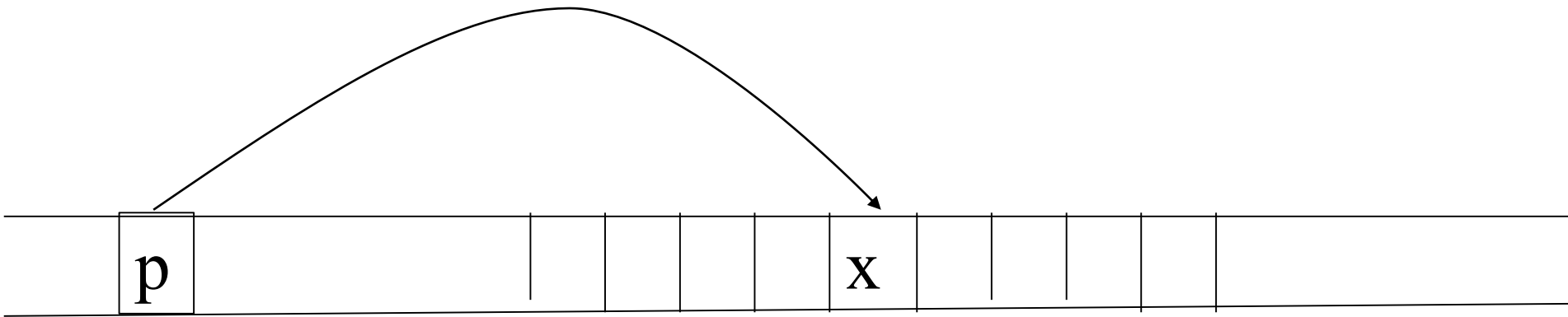
```
double x, *p=&x;
```



è come se x fosse l'elemento di un array
quindi $p[0]$, $p[1]$, $p[2]$, o $*p$, $*(p+1)$, $*(p+2)$...

il programmatore deve sapere cosa fa ...

questa è l'idea dell'aritmetica dei puntatori



...p-2, p-1, p, p+1, p+2,... sposta il puntatore p sui diversi elementi dell'array (immaginario)

$p - 2 = p - 2 * \text{sizeof}(\text{tipo elemento puntato})$

$p + 2 = p + 2 * \text{sizeof}(\text{tipo elemento puntato})$

quindi $p + 2$ ha valori diversi a seconda di cosa punta p

se

`int * p;` allora $p + 2 = p + 2 * 4$

`double * p;` $p + 2 = p + 2 * 8$

`char * p;` $p + 2 = p + 2$

tutto semplice ? Si, ma ci sono gli array a più dimensioni

che tipo hanno?

che tipo ha un array ?

int A[100] -> int* o int[]

int X[5][10] -> int (*) [10] o int [][][10]

int Y[5][5][10] -> int (*) [5][10] o int [][][5][10]

int Z[6][5][5][10] -> int (*)[5][5][10] o
int [][][5][5][10]

il loro tipo ci dice la dimensione dell'oggetto puntato:
per Z è $5*5*10*4$ (byte)

guardiamo bene

```
double F[3][5][7][9];  
tipo di F = double (*)[5][7][9]  
R-valore di F = &F[0][0][0][0]
```

se dereferenziamo un puntatore otteniamo l'oggetto puntato.

In questo caso, $*F = \text{double } [5][7][9]$ che ha tipo $\text{double } (*) [7][9]$, quindi: il tipo di $*F$ è:
 $\text{double } (*) [7][9]$

ricorda che $*F = F[0]$

ma qual è l'R-valore di *F?

sempre &F[0][0][0][0]

conviene fare una figura: l'array F è una sequenza di 3 torte ognuna formata da 5 strati con 7 righe e 9 colonne.

F punta alla prima torta,

*F punta al primo strato della prima torta

**F punta alla prima riga del primo strato della prima torta

***F punta al primo elemento della prima riga...

****F è il primo elemento della prima riga...

scrivendo i tipi :

F ha tipo double () [7][9]

**F ha tipo double (*)[9]

***F ha tipo double *

ma tutti hanno lo stesso valore = &F[0][0][0][0]

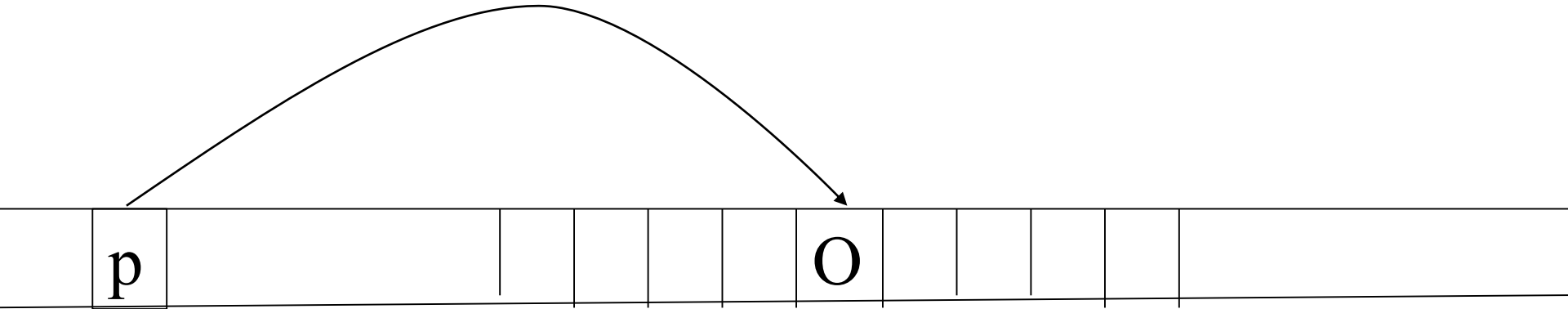
e infine ****F ha tipo double


```
cout<<F<<*F<<**F<<***F ;
```

stampa 4 volte lo stesso indirizzo che è l'L-valore
del primo elemento dell'array

aggiungiamo ora l'aritmetica dei puntatori

idea dell'aritmetica dei puntatori



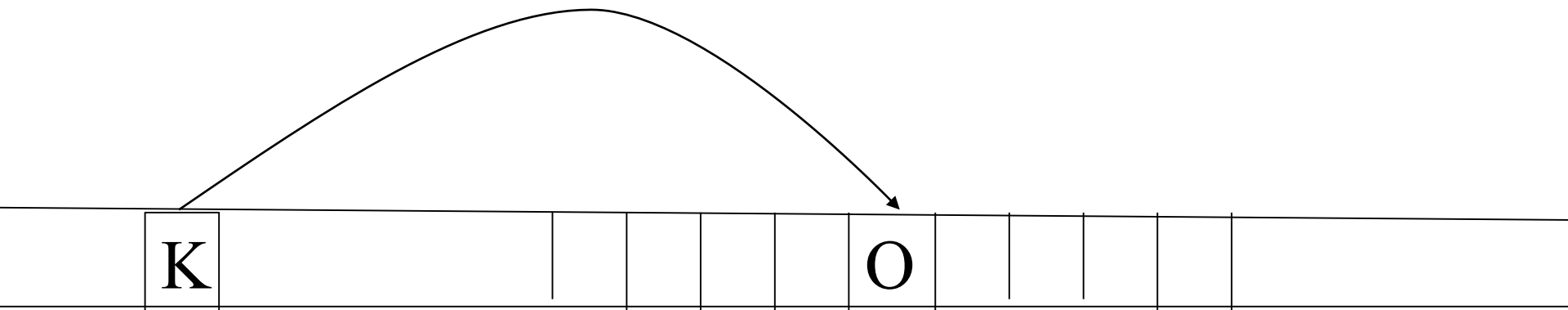
p punta ad un oggetto O , quindi $p+n$ sposta il puntatore di n elementi a destra e $p-n$ lo sposta di n elementi a sinistra

esempi con array

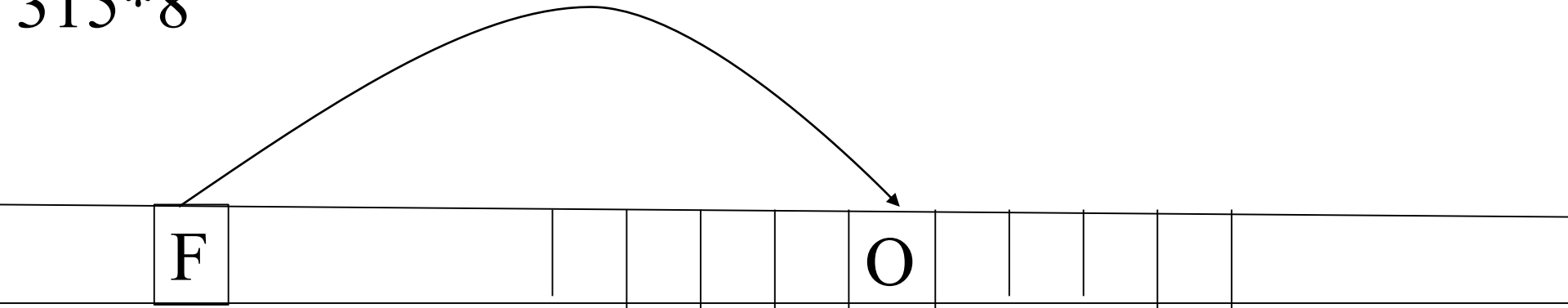
`int K[5][10];` tipo di K = `int (*) [10]`
dimensione=10*4

`char K[4][6][8];` tipo = `char (*) [6][8]` dimensione=6*8

`double K[3][5][7][9];` tipo = `double (*)[5][7][9]`
dimensione=5*7*9*8



partiamo da F, ha tipo, double (*) [5][7][9]
punta ad un oggetto O di dimensione $= (5*7*9)*8 = 315*8$

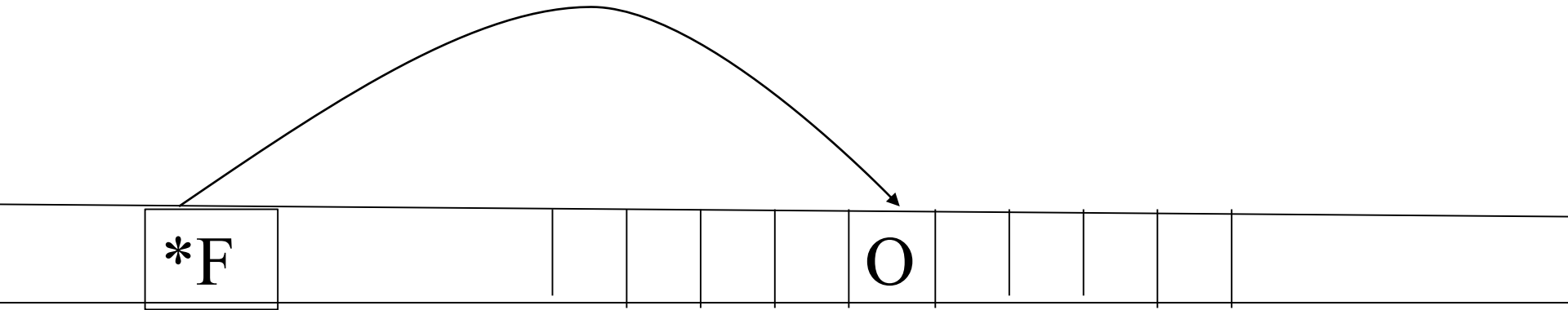


$F+1 = F + (315*8)$
 $F+2 = F + (2*315*8)$
 $F-5 = F - (5*315*8)$
eccetera

} sono tutti valori di tipo
double (*) [5][7][9]

F ha tipo double () [7][9]

punta ad un oggetto O di dimensione = $(7*9)*8=63*8$

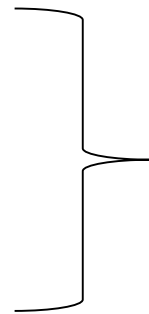


$$*F + 1 = F + 63 * 8$$

$$*F + 2 = F + 2 * 63 * 8$$

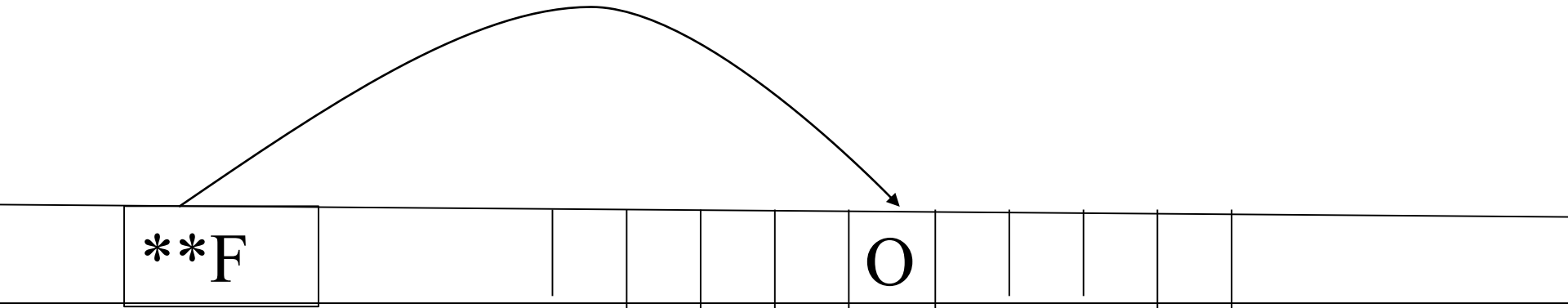
$$*F - 5 = F - 5 * 63 * 8$$

eccetera



tutti valori di tipo
double (*) [7][9]

****F** ha tipo `double (*) [9]`
punta ad un oggetto **O** di dimensione $= 9 * 8 = 72$



$**F + 1 = F + 72$
 $**F + 2 = F + 2 * 72$
 $**F - 5 = F - 5 * 72$
eccetera

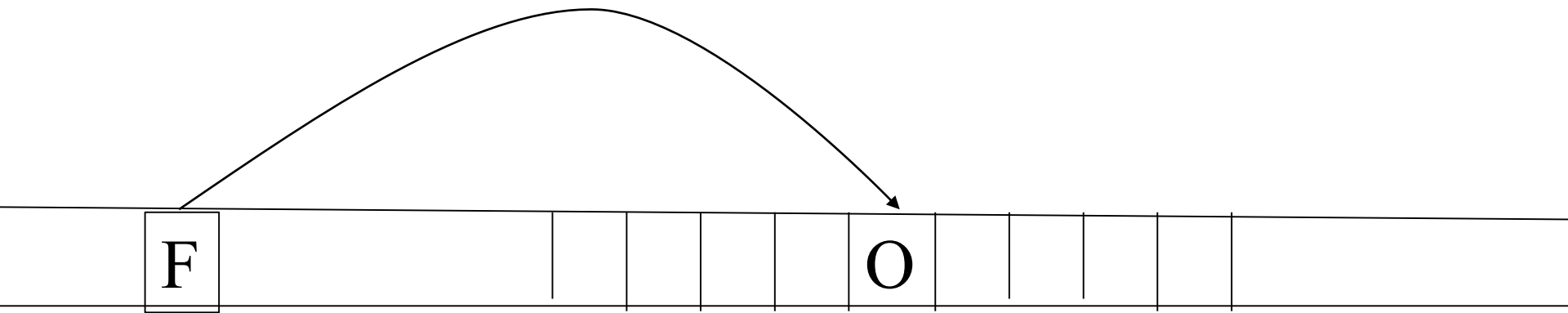
} tutti valori di tipo
`double (*) [9]`

possiamo usare subscripting ?

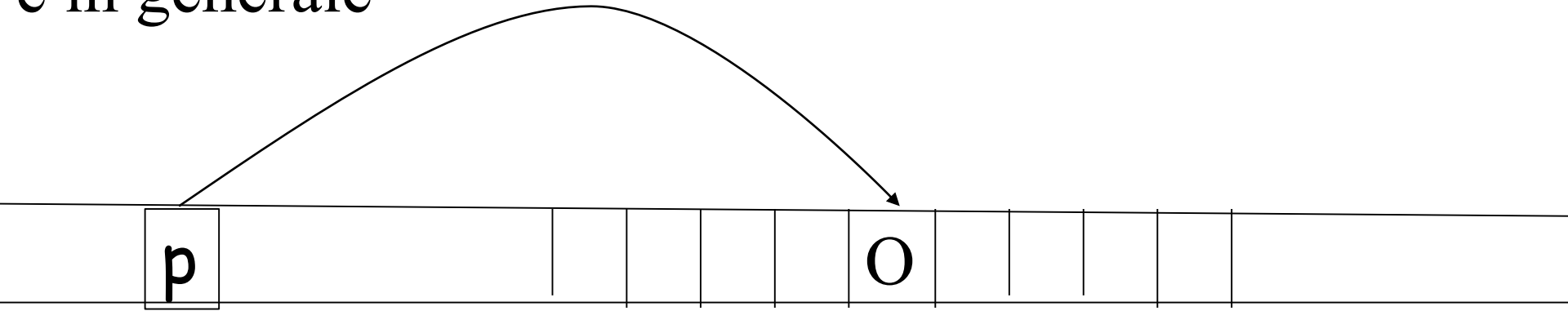
`double F[3][5][7][9];` tipo = `double (*)[5][7][9]`

`F[3] = *(F+3)` `F[-2]=*(F-2)` c'è sempre * !!

O ha dimensione = $(5*7*9)*8=315*8$



e in generale



$$p[k] = *(p+k)$$

dereferenziazione cambia il tipo
invece + cambia il valore

CAPIRE :

```
float K[3][5][7][10];
```

```
K[1]= *(K+1)
```

```
K[3][2]=*(* (K+3)+2)
```

```
K[2][1][4][1]= *(*(*(* (K+2)+1)+4)+1)
```

Esercizio 1:

float K[3][5][7][10];

K[-1][-2] = ?

K[-1] = K - (5*7*10)*4 = L1

K[-1][-2] = L1 - 2*(7*10)*4

K[-1][-2][5] = K - (5*7*10)*4 - 2*(7*10)*4 +
5*10*4

Esercizio 2

Data la seguente dichiarazione, `int A[5][6][8][5]`
che tipo e che valore ha la seguente espressione:

`((*A)-3)[2] -4`

indicare con A l'R-valore di A.

Esercizio 3

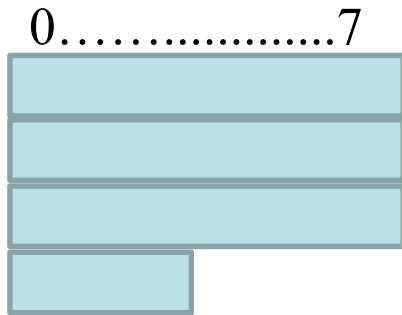
Data la seguente dichiarazione, `int A[5][6][8][5]`
che tipo e che valore ha la seguente espressione:
 $((A[-3]) + 3) [2]$

indicare con A l'R-valore di A.

Esercizio 4

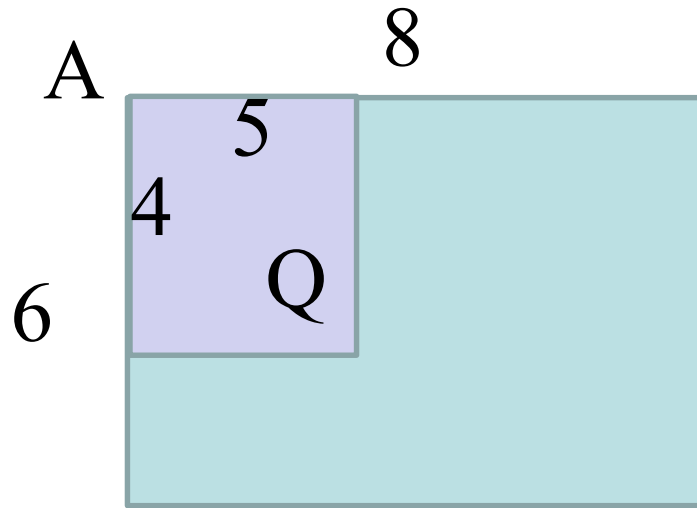
```
sia int A[6][8];  
leggi(*A, 28);
```

avremo 3 righe piene e la quarta con 4 valori



e se volessimo calcolare la matrice B di bool
come nell'esercizio 1 della settimana scorsa?

Esercizio 5



basta usare la porzione $A[4][5]=Q$ di A

Pensate che la seguente funzione sia corretta o sbagliata? Spiegate perché.

```
int *& f (int *& x, int & y) {x=&y;  
return x;}
```

Pensate che la seguente funzione sia corretta o sbagliata? Spiegate perché.

```
int *& f (int ** x) {int b=5; *x=&b; return *x;}
```