



Informatica B

2016-2017

Esercitazione 10

Ripassiamo Matrici, Struct e Codifica Binaria

Alessandro A. Nacci

alessandro.nacci@polimi.it - www.alessandronacci.it

Codifica binaria - Da decimale a binario

- Esempio: il numero 6_{10} :

$$6/2 = 3 \text{ resto } 0$$

$$3/2 = 1 \text{ resto } 1$$

$$1/2 = 0 \text{ resto } 1$$

- Leggendo i resti dal basso verso l'alto, si ha che la rappresentazione binaria del numero 6_{10} è 110_2
- Per una corretta verifica basta riconvertire il risultato alla base 10
 - Cioè, calcolare $1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

divido per due

Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

divido per due resto

Esercizio 3 - Da decimale a binario

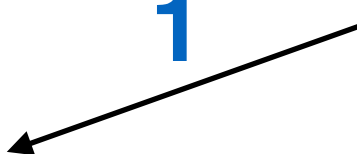
Come si scrive 37 in binario?

divido per due resto risultato

Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

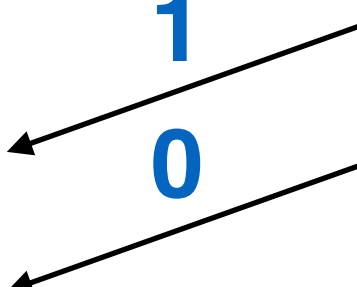
divido per due	resto	risultato
37	1	18



Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

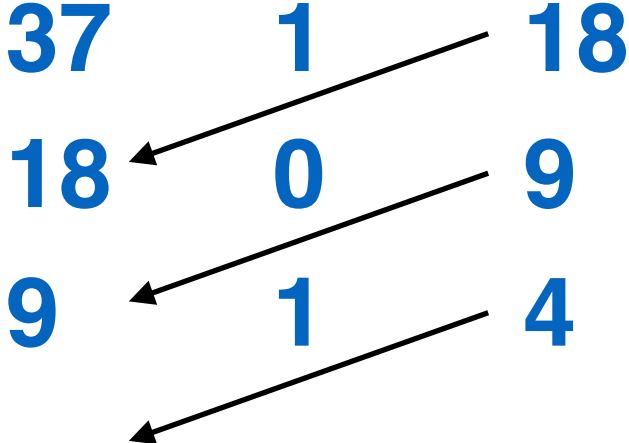
divido per due	resto	risultato
37	1	18
18	0	9



Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

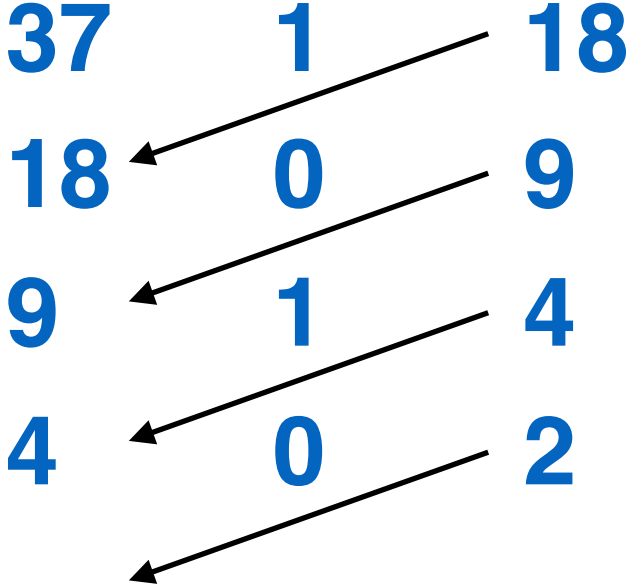
divido per due	resto	risultato
37	1	18
18	0	9
9	1	4



Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

divido per due	resto	risultato
37	1	18
18	0	9
9	1	4
4	0	2



Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

divido per due	resto	risultato
37	1	18
18	0	9
9	1	4
4	0	2
2	0	1

Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

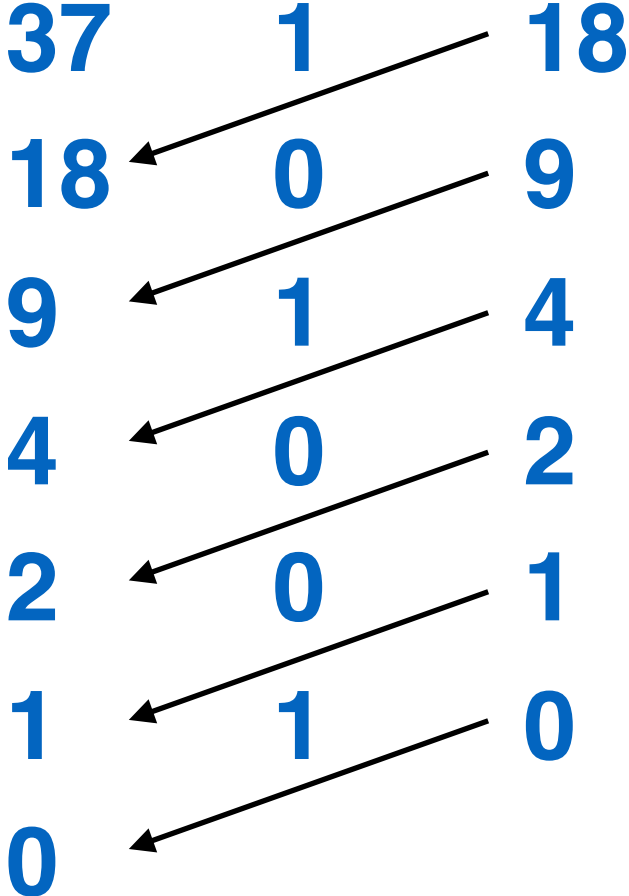
divido per due resto risultato

37	1	18
18	0	9
9	1	4
4	0	2
2	0	1
1	1	0

Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

divido per due	resto	risultato
37	1	18
18	0	9
9	1	4
4	0	2
2	0	1
1	1	0
0		



Esercizio 3 - Da decimale a binario

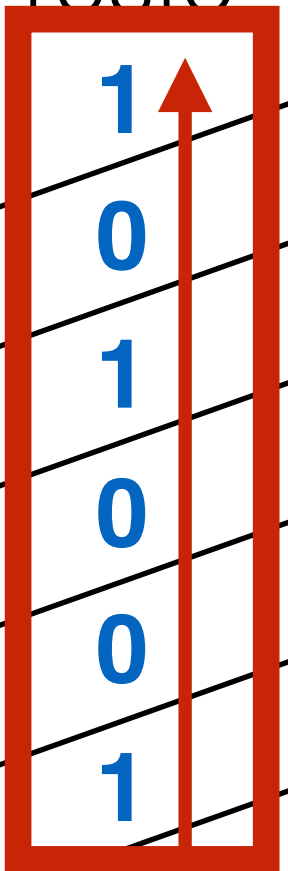
Come si scrive 37 in binario?

divido per due	resto	risultato
37	1	18
18	0	9
9	1	4
4	0	2
2	0	1
1	1	0
0		

Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

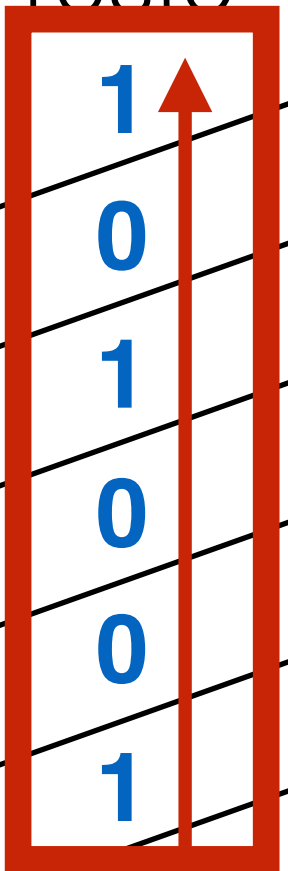
divido per due	resto	risultato
37	1	18
18	0	9
9	1	4
4	0	2
2	0	1
1	1	0
0		



Esercizio 3 - Da decimale a binario

Come si scrive 37 in binario?

divido per due	resto	risultato
37	1	18
18	0	9
9	1	4
4	0	2
2	0	1
1	1	0
0		



37 in binario è **100101**

Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

divido per due

Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

divido per due resto

Esercizio 4 - Da decimale a binario

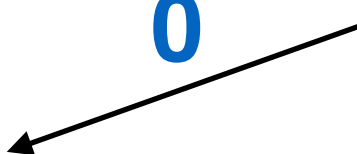
Come si scrive 52 in binario?

divido per due	resto	risultato
-----------------------	-------	-----------

Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

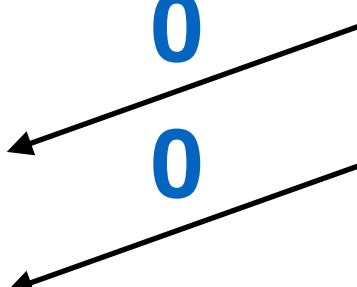
divido per due	resto	risultato
52	0	26



Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

divido per due	resto	risultato
52	0	26
26	0	13



Esercizio 4 - Da decimale a binario

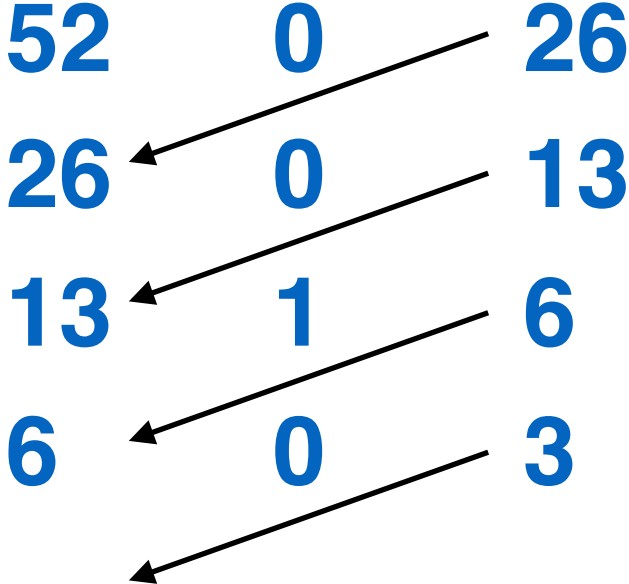
Come si scrive 52 in binario?

divido per due	resto	risultato
52	0	26
26	0	13
13	1	6

Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

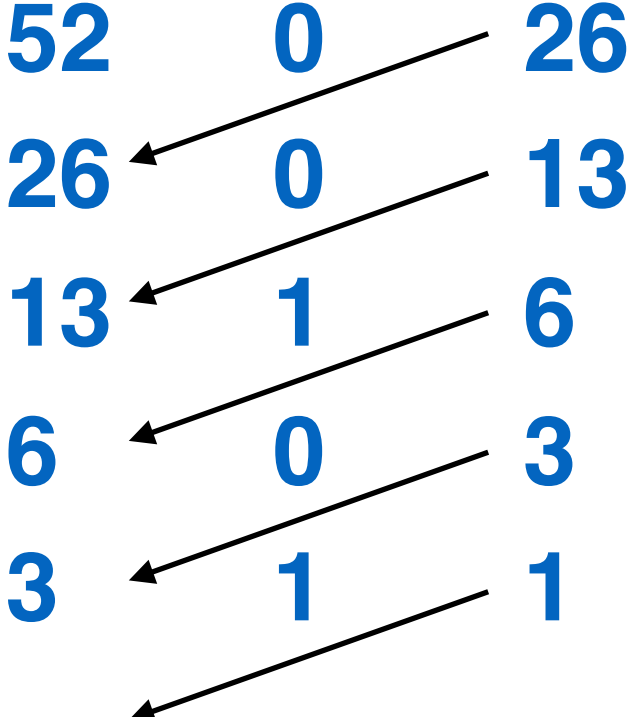
divido per due	resto	risultato
52	0	26
26	0	13
13	1	6
6	0	3



Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

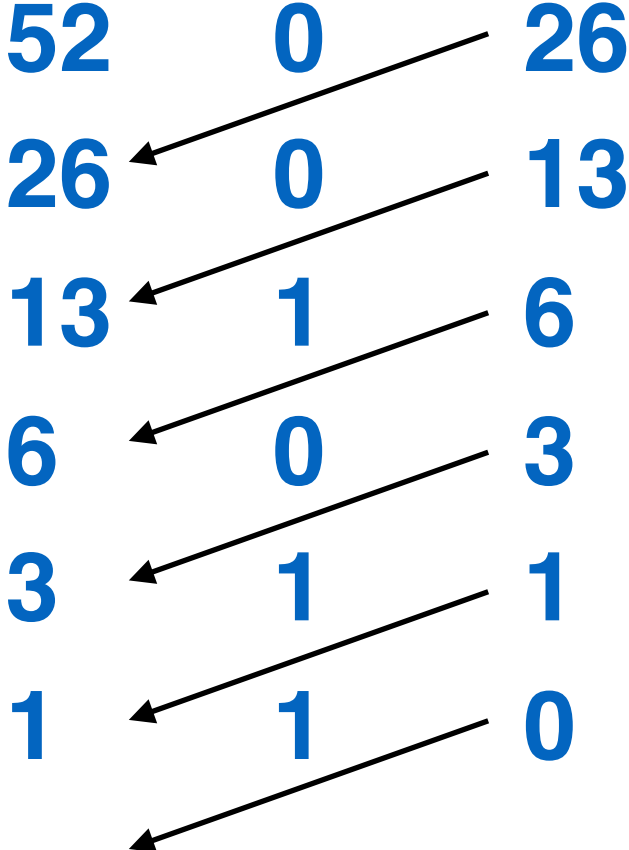
divido per due	resto	risultato
52	0	26
26	0	13
13	1	6
6	0	3
3	1	1



Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

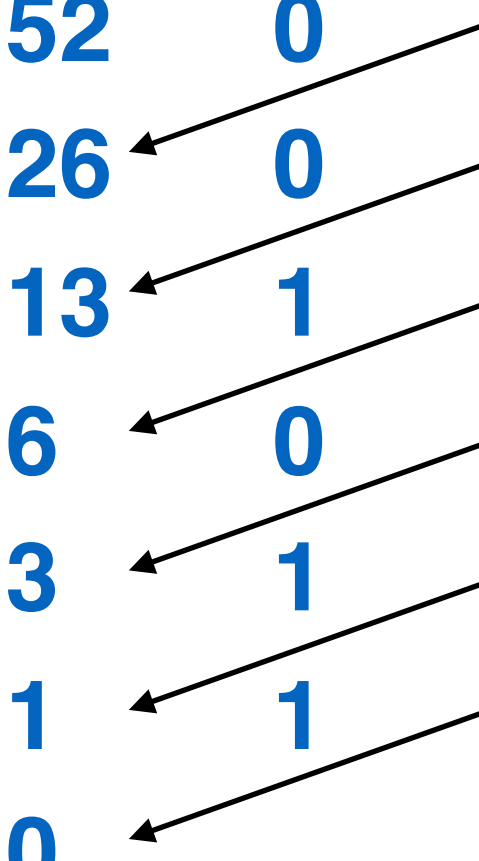
divido per due	resto	risultato
52	0	26
26	0	13
13	1	6
6	0	3
3	1	1
1	1	0



Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

divido per due	resto	risultato
52	0	26
26	0	13
13	1	6
6	0	3
3	1	1
1	1	0
0		



Esercizio 4 - Da decimale a binario

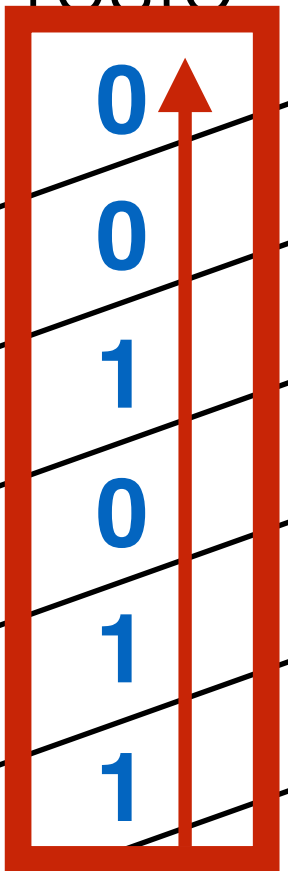
Come si scrive 52 in binario?

divido per due	resto	risultato
52	0	26
26	0	13
13	1	6
6	0	3
3	1	1
1	1	0
0		

Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

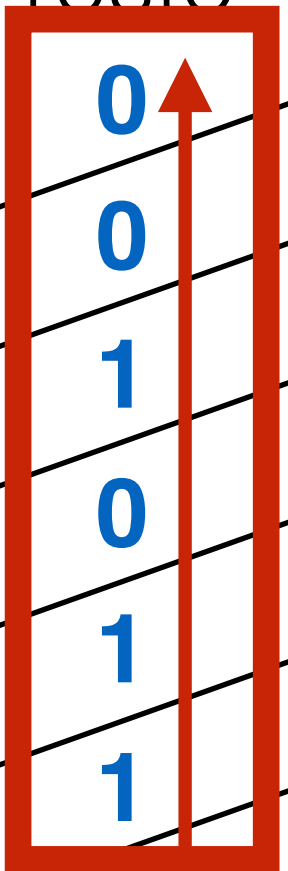
divido per due	resto	risultato
52	0	26
26	0	13
13	1	6
6	0	3
3	1	1
1	1	0
0		



Esercizio 4 - Da decimale a binario

Come si scrive 52 in binario?

divido per due	resto	risultato
52	0	26
26	0	13
13	1	6
6	0	3
3	1	1
1	1	0
0		



52 in binario è **110100**

Codifica binaria - Da decimale a complemento a 2

Codifica binaria - Da decimale a complemento a 2

- Il bit più significativo rappresenta il segno del numero: 0 per i numeri positivi e 1 per i numeri negativi

Codifica binaria - Da decimale a complemento a 2

- Il bit più significativo rappresenta il segno del numero: 0 per i numeri positivi e 1 per i numeri negativi
- La rappresentazione di un numero positivo si ottiene codificando il valore assoluto del numero con i bit restanti

Codifica binaria - Da decimale a complemento a 2

- Il bit più significativo rappresenta il segno del numero: 0 per i numeri positivi e 1 per i numeri negativi
- La rappresentazione di un numero positivo si ottiene codificando il valore assoluto del numero con i bit restanti
- La rappresentazione di un numero negativo si ottiene in tre passi:
 - Si rappresenta in complemento a due il numero positivo con lo stesso valore assoluto del numero negativo da codificare
 - Si invertono tutti i bit in tale rappresentazione ($0 \rightarrow 1, 1 \rightarrow 0$)
 - Si somma uno al risultato ottenuto al passo precedente

Codifica binaria - Operazioni

- **Addizione:**

$$0 + 0 = 0 \text{ con riporto } 0$$

$$0 + 1 = 1 \text{ con riporto } 0$$

$$1 + 0 = 1 \text{ con riporto } 0$$

$$1 + 1 = 0 \text{ con riporto } 1$$

- **Esempi:**

$$\begin{array}{r} 1 + \\ \underline{1} = \end{array}$$

Codifica binaria - Operazioni

- **Addizione:**

$$0 + 0 = 0 \text{ con riporto } 0$$

$$0 + 1 = 1 \text{ con riporto } 0$$

$$1 + 0 = 1 \text{ con riporto } 0$$

$$1 + 1 = 0 \text{ con riporto } 1$$

- **Esempi:**

$$\begin{array}{r} 1 + \\ 1 = \\ \hline 10 \end{array} \quad \begin{array}{r} 101 + \\ 11 = \\ \hline \end{array}$$

Codifica binaria - Operazioni

- **Addizione:**

$$0 + 0 = 0 \text{ con riporto } 0$$

$$0 + 1 = 1 \text{ con riporto } 0$$

$$1 + 0 = 1 \text{ con riporto } 0$$

$$1 + 1 = 0 \text{ con riporto } 1$$

- **Esempi:**

$$\begin{array}{r} 1 + \\ 1 = \\ \hline 10 \end{array}$$

$$\begin{array}{r} 101 + \\ 11 = \\ \hline 1000 \end{array}$$

$$\begin{array}{r} 10110101 + \\ 1000110 = \\ \hline \end{array}$$

Codifica binaria - Operazioni

- **Addizione:**

$$0 + 0 = 0 \text{ con riporto } 0$$

$$0 + 1 = 1 \text{ con riporto } 0$$

$$1 + 0 = 1 \text{ con riporto } 0$$

$$1 + 1 = 0 \text{ con riporto } 1$$

- **Esempi:**

$$\begin{array}{r} 1 + \\ 1 = \\ \hline 10 \end{array}$$

$$\begin{array}{r} 101 + \\ 11 = \\ \hline 1000 \end{array}$$

$$\begin{array}{r} 10110101 + \\ 1000110 = \\ \hline 11111011 \end{array}$$

$$\begin{array}{r} 111 + \\ 11 = \\ \hline \end{array}$$

Codifica binaria - Operazioni

- **Addizione:**

$$0 + 0 = 0 \text{ con riporto } 0$$

$$0 + 1 = 1 \text{ con riporto } 0$$

$$1 + 0 = 1 \text{ con riporto } 0$$

$$1 + 1 = 0 \text{ con riporto } 1$$

- **Esempi:**

$$\begin{array}{r} 1 + \\ 1 = \\ \hline 10 \end{array}$$

$$\begin{array}{r} 101 + \\ 11 = \\ \hline 1000 \end{array}$$

$$\begin{array}{r} 10110101 + \\ 1000110 = \\ \hline 11111011 \end{array}$$

$$\begin{array}{r} 111 + \\ 11 = \\ \hline 1010 \end{array}$$

Esercizio 5 - Da binario a complemento a 2

Come si scrive -37 in complemento a 2?

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione ($0 \rightarrow 1, 1 \rightarrow 0$)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 5 - Da binario a complemento a 2

Come si scrive -37 in complemento a 2?

1) 37 in binario è **100101**

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione ($0 \rightarrow 1, 1 \rightarrow 0$)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 5 - Da binario a complemento a 2

Come si scrive -37 in complemento a 2?

1) 37 in binario è **100101**

2) **100101** $\longrightarrow$ **011010**

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 5 - Da binario a complemento a 2

Come si scrive -37 in complemento a 2?

1) 37 in binario è **100101**

2) **100101** $\longrightarrow$ **011010**

3) **011010** +
 1 =

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 5 - Da binario a complemento a 2

Come si scrive -37 in complemento a 2?

1) 37 in binario è **100101**

2) **100101** $\longrightarrow$ **011010**

$$\begin{array}{r} 3) \text{ **011010** + } \\ \text{ **1** = } \\ \hline \text{ **011011** } \end{array}$$

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 5 - Da binario a complemento a 2

Come si scrive -37 in complemento a 2?

1) 37 in binario è **100101**

2) **100101** $\longrightarrow$ **011010**

$$\begin{array}{r} 3) \text{ **011010** + } \\ \quad \quad \quad \text{1 =} \\ \hline \text{011011} \longrightarrow \end{array}$$

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 5 - Da binario a complemento a 2

Come si scrive -37 in complemento a 2?

1) 37 in binario è **100101**

2) **100101** $\longrightarrow$ **011010**

$$\begin{array}{r} 3) \text{ **011010** + } \\ \quad \quad \quad \text{1 =} \\ \hline \quad \quad \text{011011} \end{array} \longrightarrow \boxed{\text{1011011}}$$

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 6 - Da binario a complemento a 2

Come si scrive -52 in complemento a 2?

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione ($0 \rightarrow 1, 1 \rightarrow 0$)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 6 - Da binario a complemento a 2

Come si scrive -52 in complemento a 2?

1) 52 in binario è **110100**

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione ($0 \rightarrow 1, 1 \rightarrow 0$)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 6 - Da binario a complemento a 2

Come si scrive -52 in complemento a 2?

1) 52 in binario è **110100**

2) **110100** $\longrightarrow$ **001011**

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 6 - Da binario a complemento a 2

Come si scrive -52 in complemento a 2?

1) 52 in binario è **110100**

2) **110100** $\longrightarrow$ **001011**

3) **001011** +
 1 =

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 6 - Da binario a complemento a 2

Come si scrive -52 in complemento a 2?

1) 52 in binario è **110100**

2) **110100** $\longrightarrow$ **001011**

$$\begin{array}{r} 3) \text{ **001011** + } \\ \text{ **1** = } \\ \hline \text{ **001100** } \end{array}$$

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 6 - Da binario a complemento a 2

Come si scrive -52 in complemento a 2?

1) 52 in binario è **110100**

2) **110100** $\longrightarrow$ **001011**

$$\begin{array}{r} 3) \text{ **001011** + } \\ \text{ **1** = } \\ \hline \text{ **001100** } \longrightarrow \end{array}$$

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 6 - Da binario a complemento a 2

Come si scrive -52 in complemento a 2?

1) 52 in binario è **110100**

2) **110100** $\longrightarrow$ **001011**

$$\begin{array}{r} 3) \text{ **001011** + } \\ \quad \quad \quad \text{1 =} \\ \hline \quad \quad \text{001100} \end{array} \longrightarrow \boxed{\text{1001100}}$$

- Si rappresenta in complemento a due il numeri positivo con lo stesso valore assoluto del numero negativo da codificare
- Si invertono tutti i bit in tale rappresentazione (0 $\rightarrow$ 1, 1 $\rightarrow$ 0)
- Si somma uno al risultato ottenuto al passo precedente

Esercizio 7 - Operazioni in complemento a 2

Quanto fa **37 + 52** ?

Esercizio 7 - Operazioni in complemento a 2

Quanto fa **37 + 52** ?

$$\begin{array}{r} 100101 + \\ 110100 = \\ \hline \end{array}$$

Esercizio 7 - Operazioni in complemento a 2

Quanto fa **37 + 52** ?

$$\begin{array}{r} 100101 + \\ 110100 = \\ \hline 1011001 \end{array}$$

Esercizio 8 - Operazioni in complemento a 2

Quanto fa **37 - 52** ?

Esercizio 8 - Operazioni in complemento a 2

Quanto fa **37 - 52** ?

$$\begin{array}{r} 0100101 + \\ 1001100 = \\ \hline \end{array}$$

Esercizio 8 - Operazioni in complemento a 2

Quanto fa **37 - 52** ?

$$\begin{array}{r} 0100101 + \\ 1001100 = \\ \hline 1110001 \end{array}$$

Codifica binaria - Da complemento a 2 a decimale

- Per ottenere un numero con segno data la sua rappresentazione in complemento a due:
 - Se il primo bit è 0 il numero è positivo: per calcolarne il valore assoluto si esegue la conversione da binario a decimale

Codifica binaria - Da complemento a 2 a decimale

- Per ottenere un numero con segno data la sua rappresentazione in complemento a due:
 - Se il primo bit è 0 il numero è positivo: per calcolarne il valore assoluto si esegue la conversione da binario a decimale
 - Se il primo bit è 1 il numero è negativo:

Codifica binaria - Da complemento a 2 a decimale

- Per ottenere un numero con segno data la sua rappresentazione in complemento a due:
 - Se il primo bit è 0 il numero è positivo: per calcolarne il valore assoluto si esegue la conversione da binario a decimale
 - Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit

Codifica binaria - Da complemento a 2 a decimale

- Per ottenere un numero con segno data la sua rappresentazione in complemento a due:
 - Se il primo bit è 0 il numero è positivo: per calcolarne il valore assoluto si esegue la conversione da binario a decimale
 - Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit

Codifica binaria - Da complemento a 2 a decimale

- Per ottenere un numero con segno data la sua rappresentazione in complemento a due:
 - Se il primo bit è 0 il numero è positivo: per calcolarne il valore assoluto si esegue la conversione da binario a decimale
 - Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit
 - Si converte il numero da binario a decimale

Esercizio 8 - Operazioni in complemento a 2

37 - 52 : come si torna in decimale?

- Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit
 - Si converte il numero da binario a decimale
 - Si somma uno al numero ottenuto per ottenere il valore assoluto del numero negativo

Esercizio 8 - Operazioni in complemento a 2

37 - 52 : come si torna in decimale?

1110001

- Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit
 - Si converte il numero da binario a decimale
 - Si somma uno al numero ottenuto per ottenere il valore assoluto del numero negativo

Esercizio 8 - Operazioni in complemento a 2

37 - 52 : come si torna in decimale?

1110001

1) **1110001**

- Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit
 - Si converte il numero da binario a decimale
 - Si somma uno al numero ottenuto per ottenere il valore assoluto del numero negativo

Esercizio 8 - Operazioni in complemento a 2

37 - 52 : come si torna in decimale?

1110001

1) **1110001**

2) **110001** $\longrightarrow$ **001110**

- Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit
 - Si converte il numero da binario a decimale
 - Si somma uno al numero ottenuto per ottenere il valore assoluto del numero negativo

Esercizio 8 - Operazioni in complemento a 2

37 - 52 : come si torna in decimale?

1110001

1) **1110001**

2) **110001** $\longrightarrow$ **001110**

3) $0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 14$

- Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit
 - Si converte il numero da binario a decimale
 - Si somma uno al numero ottenuto per ottenere il valore assoluto del numero negativo

Esercizio 8 - Operazioni in complemento a 2

37 - 52 : come si torna in decimale?

1110001

1) **1110001**

2) **110001** $\longrightarrow$ **001110**

3) $0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 14$

4) $14 + 1 = 15$

- Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit
 - Si converte il numero da binario a decimale
 - Si somma uno al numero ottenuto per ottenere il valore assoluto del numero negativo

Esercizio 8 - Operazioni in complemento a 2

37 - 52 : come si torna in decimale?

1110001

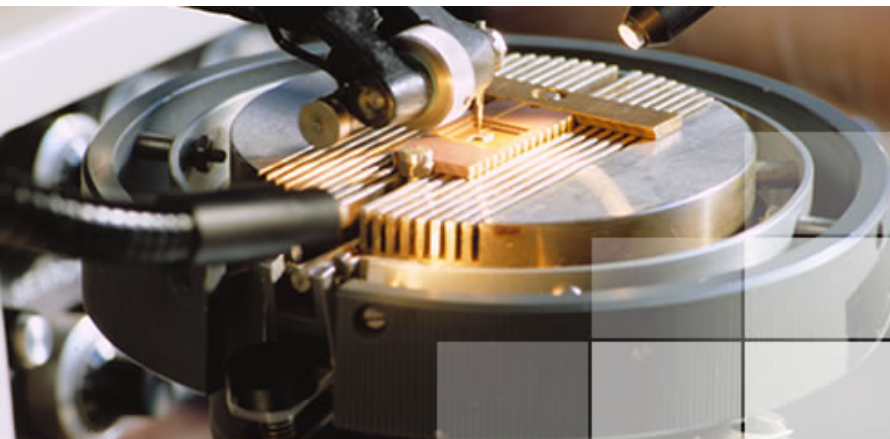
1) **1110001**

2) **110001** $\longrightarrow$ **001110**

3) $0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 14$

4) $14 + 1 = 15 \longrightarrow$ **-15**

- Se il primo bit è 1 il numero è negativo:
 - Si ignora il primo bit
 - Si invertono i restanti bit
 - Si converte il numero da binario a decimale
 - Si somma uno al numero ottenuto per ottenere il valore assoluto del numero negativo



 POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

Array, matrici e struct

Thanks to Matteo Ferroni



POLITECNICO
DI MILANO



Esercizio: Matrici

- Scrivere un programma che legge una matrice quadrata di dimensioni specificate dall'utente (al massimo 10 righe e 10 colonne):
 - Calcolare la somma dei valori sulle **righe**
 - Calcolare la somma dei valori sulle **colonne**
 - calcolare la somma dei valori sulla **diagonale** principale
 - calcolare la somma dei valori **sopra la diagonale** principale
 - calcolare la somma dei valori **sotto la diagonale** principale

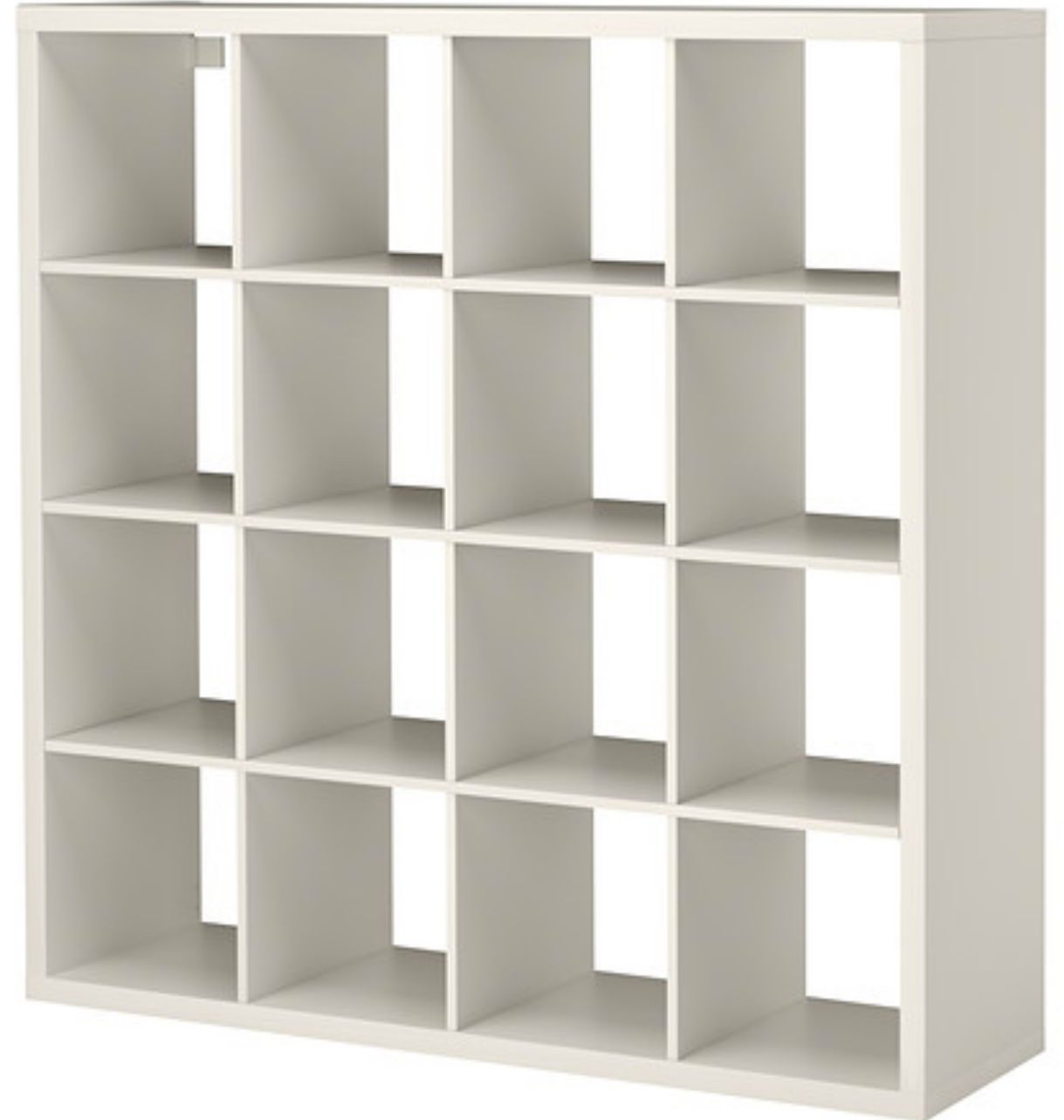
Matrici nel mondo reale

Matrici nel mondo reale

```
int matrice[4][4];
```

Matrici nel mondo reale

```
int matrice[4][4];
```



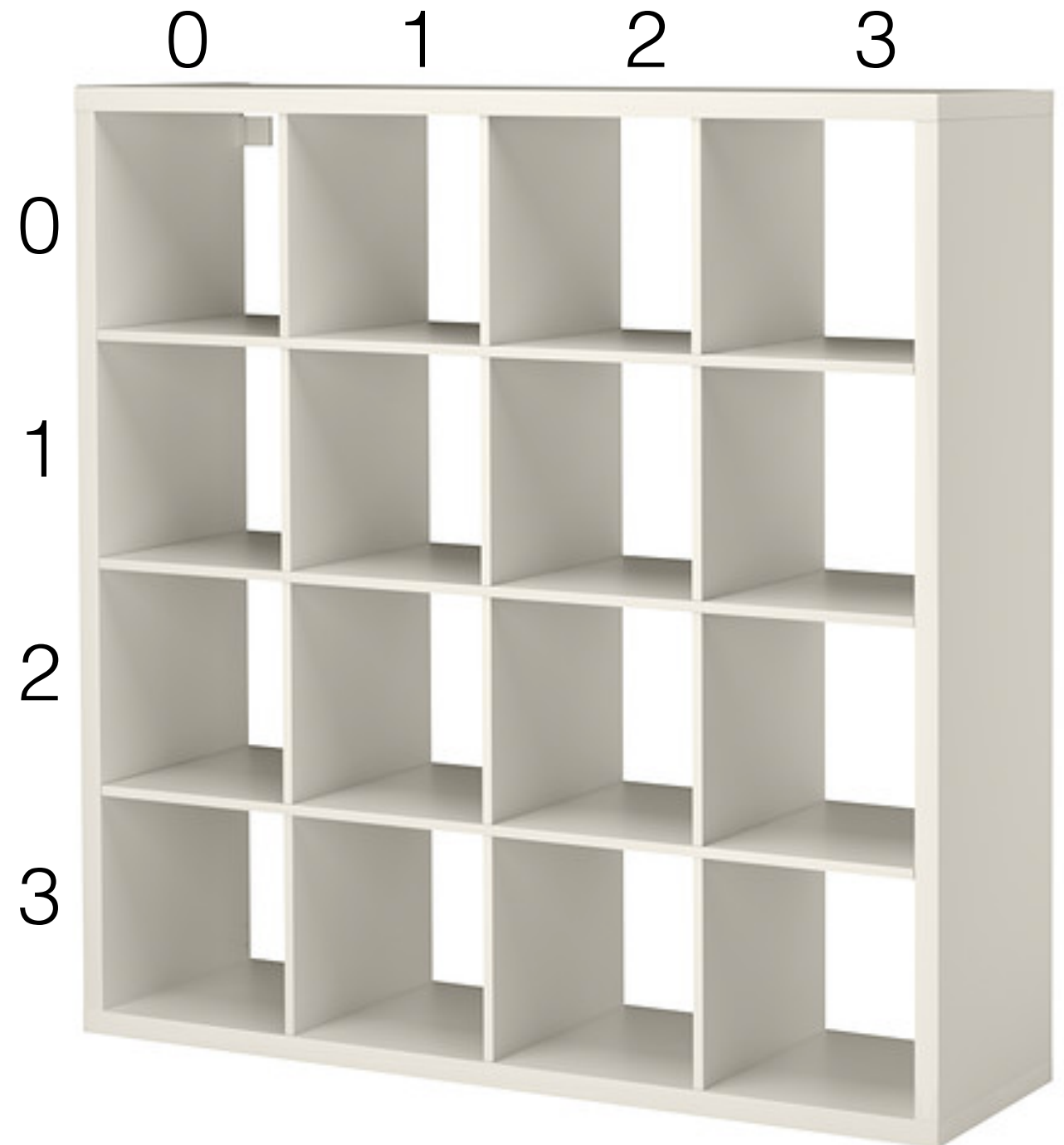
Matrici nel mondo reale

```
int matrice[4][4];
```



Matrici nel mondo reale

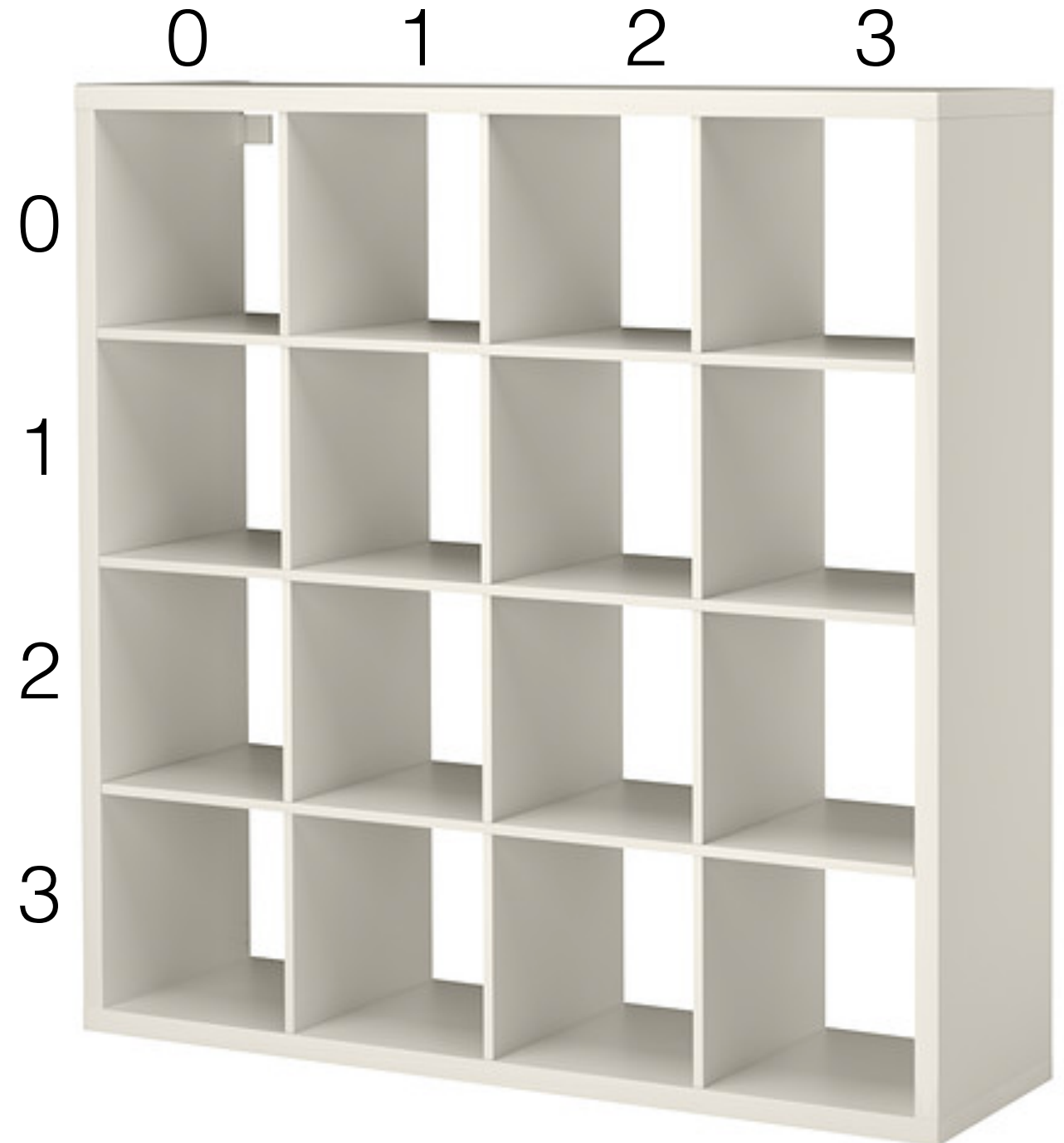
```
int matrice[4][4];
```



Matrici nel mondo reale

```
int matrice[4][4];
```

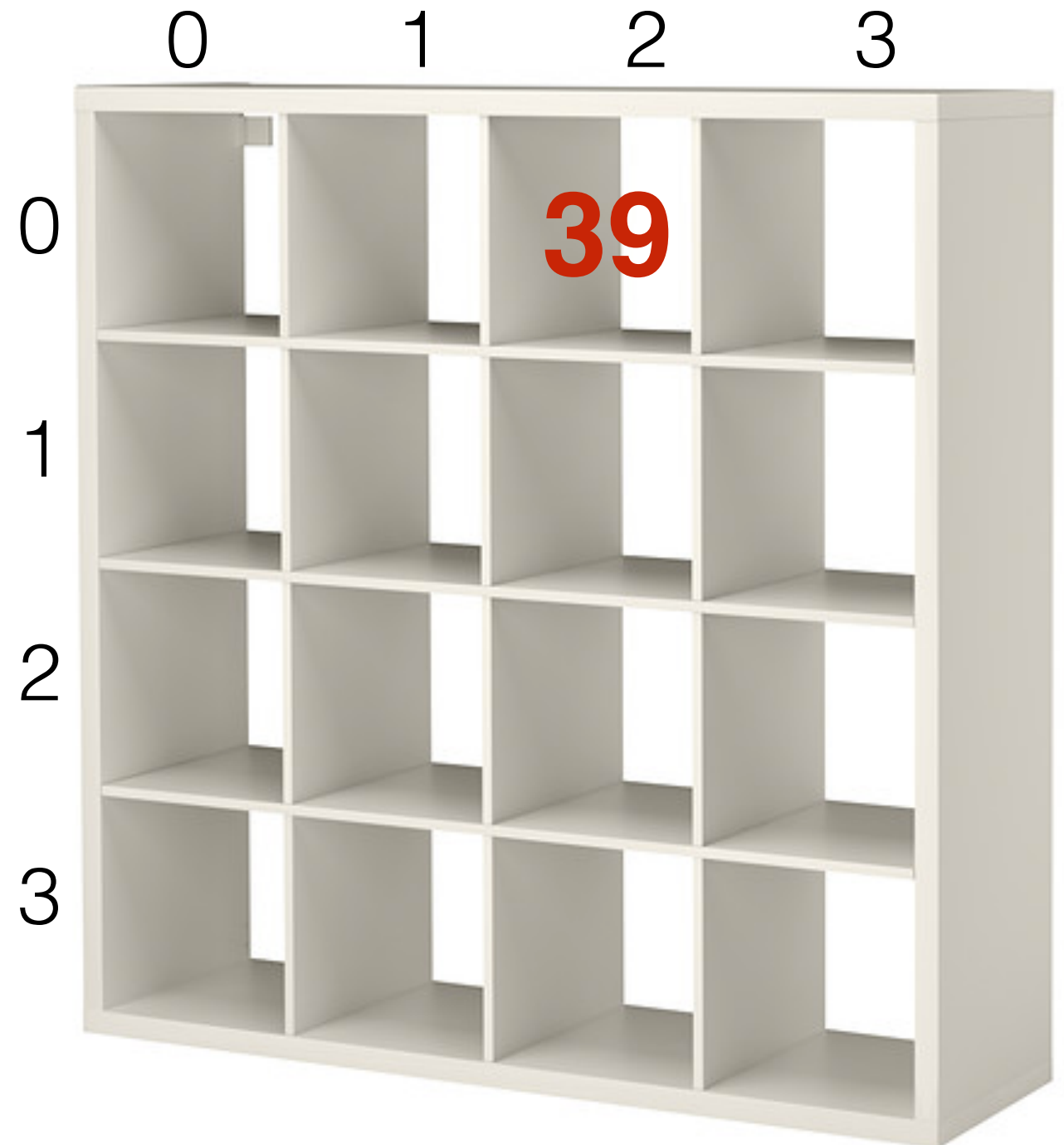
```
matrice[0][2] = 39;
```



Matrici nel mondo reale

```
int matrice[4][4];
```

```
matrice[0][2] = 39;
```

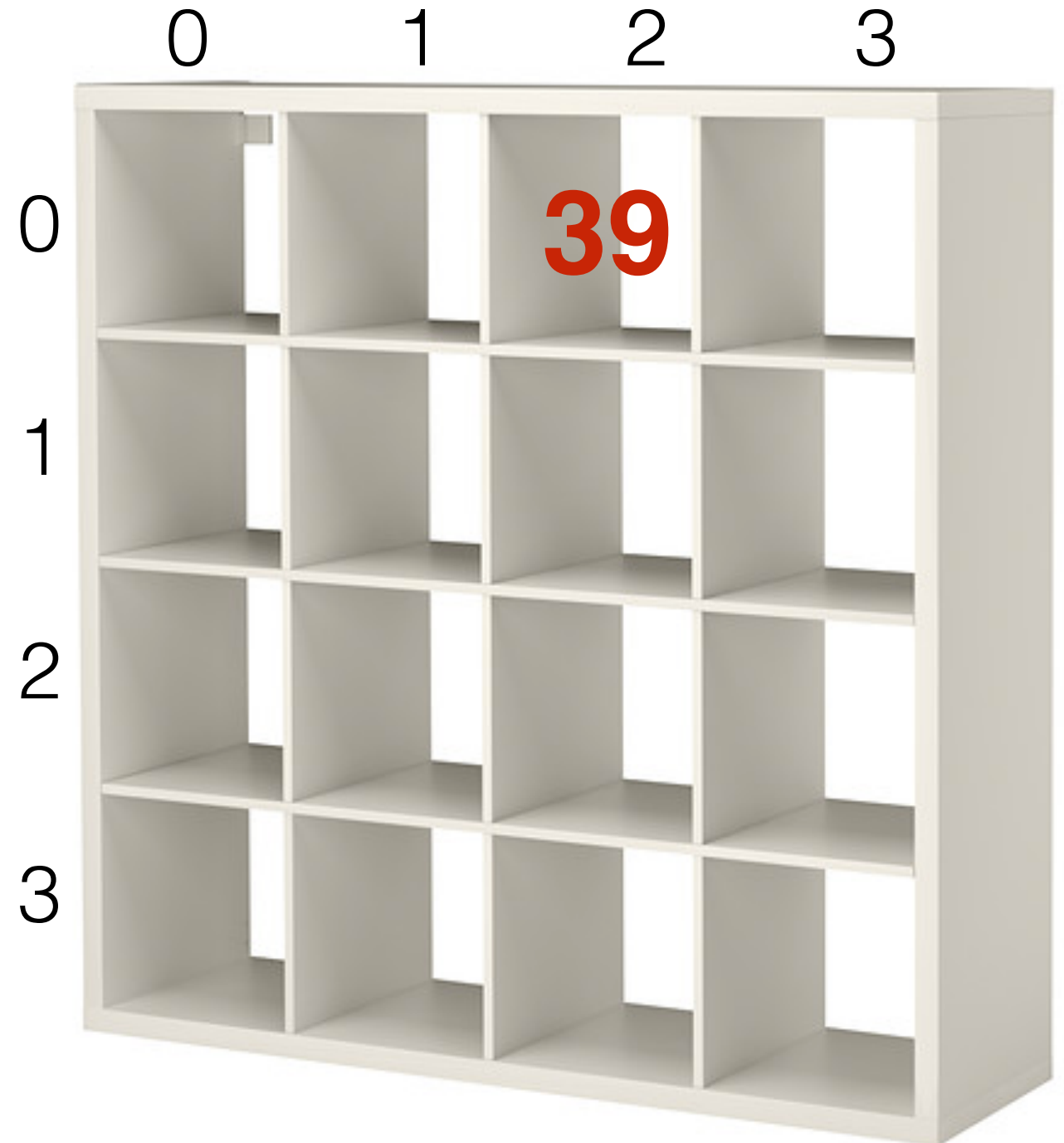


Matrici nel mondo reale

```
int matrice[4][4];
```

```
matrice[0][2] = 39;
```

```
int i = 2;  
int j = 3;
```



Matrici nel mondo reale

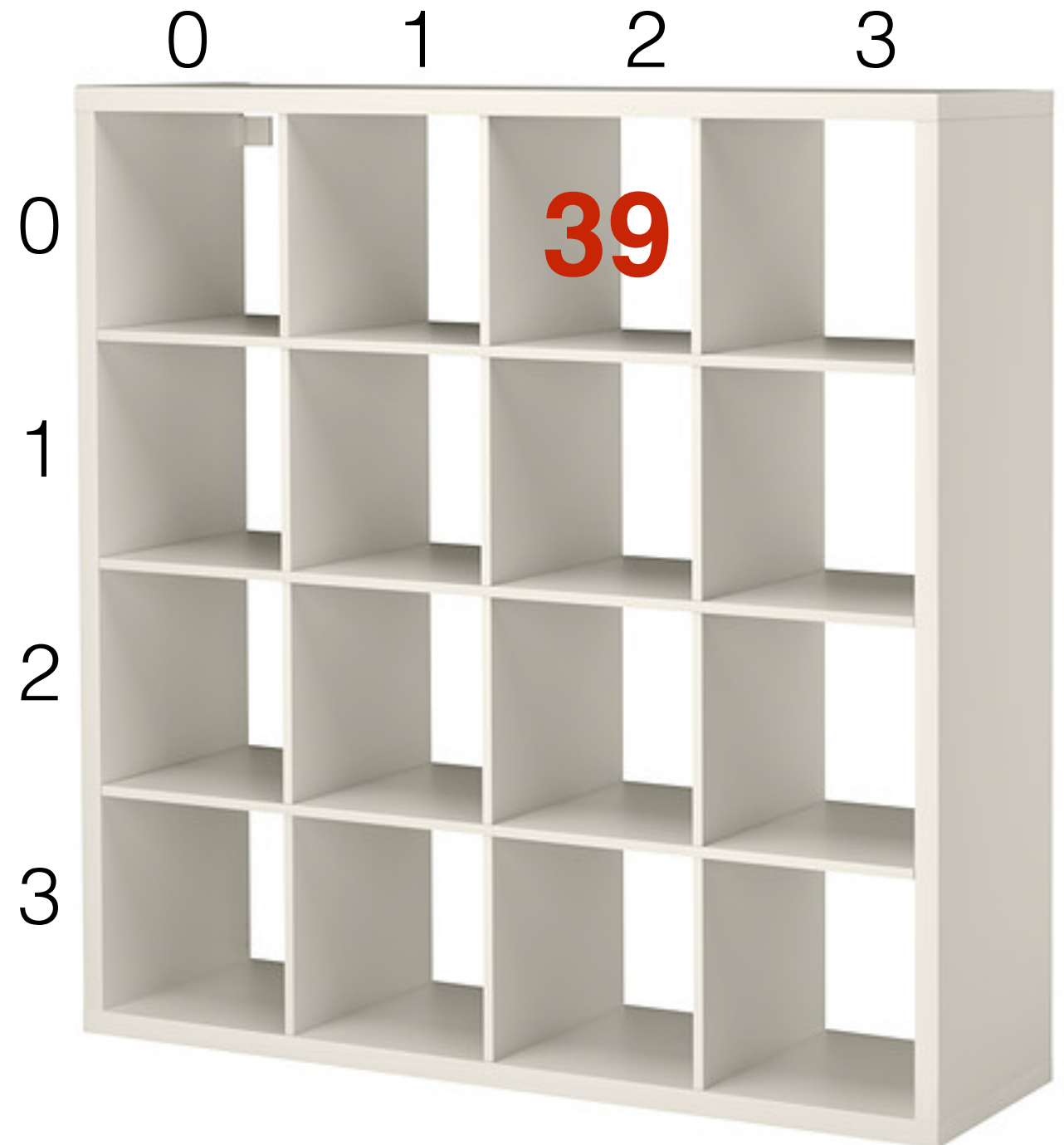
```
int matrice[4][4];
```

```
matrice[0][2] = 39;
```

```
int i = 2;
```

```
int j = 3;
```

```
matrice[i][j] = 42;
```



Matrici nel mondo reale

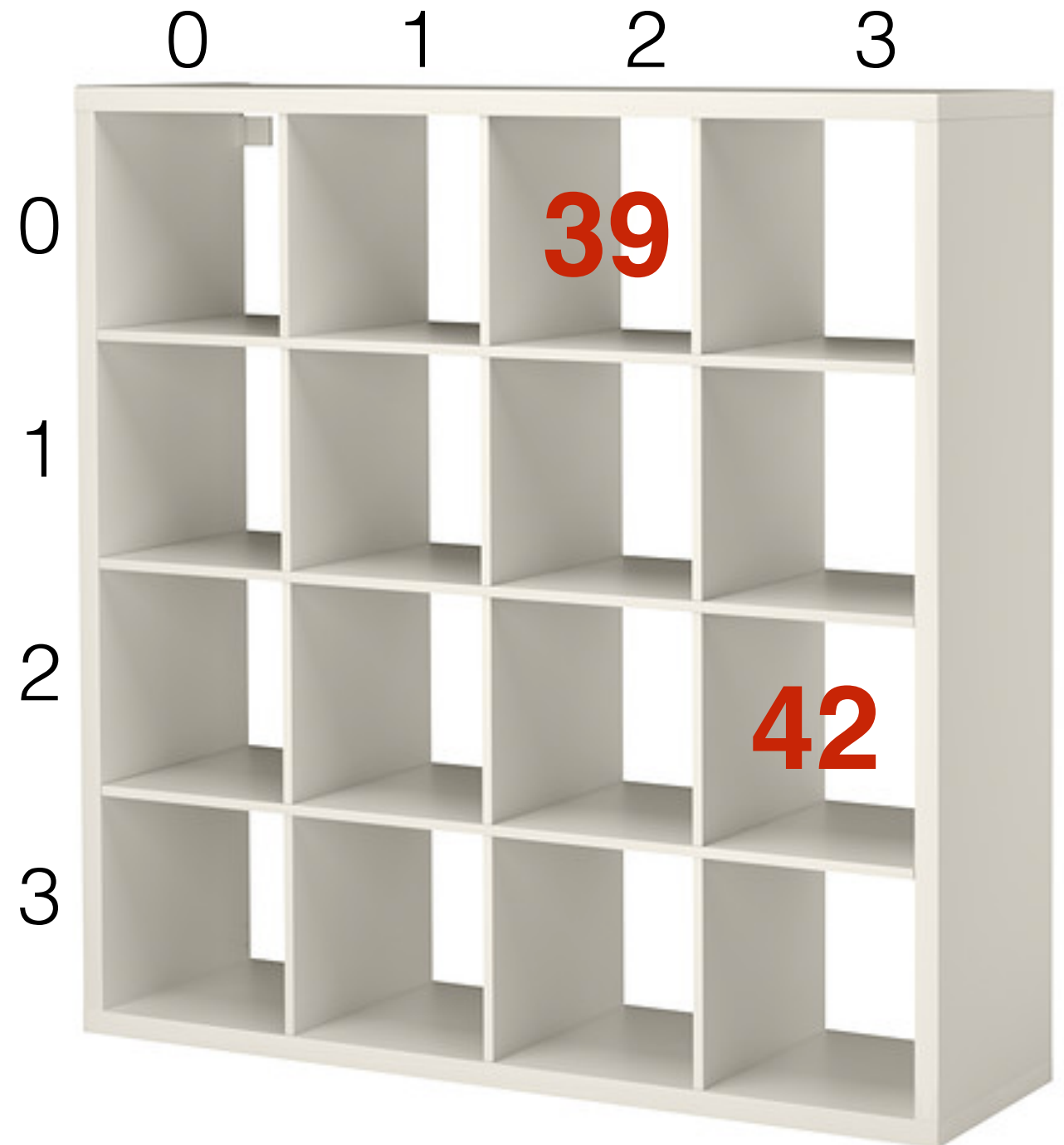
```
int matrice[4][4];
```

```
matrice[0][2] = 39;
```

```
int i = 2;
```

```
int j = 3;
```

```
matrice[i][j] = 42;
```



Esercizio: Matrici

- Scrivere un programma che legge una matrice quadrata di dimensioni specificate dall'utente (al massimo 10 righe e 10 colonne):
 - Calcolare la somma dei valori sulle **righe**
 - Calcolare la somma dei valori sulle **colonne**
 - calcolare la somma dei valori sulla **diagonale** principale
 - calcolare la somma dei valori **sopra la diagonale** principale
 - calcolare la somma dei valori **sotto la diagonale** principale

Esercizio: Matrici

Esercizio: Matrici

```
#include <stdio.h>
#include <string.h>

#define MAXDIM 10
```

Esercizio: Matrici

```
#include <stdio.h>
#include <string.h>

#define MAXDIM 10

int main(){
    int matrice[MAXDIM][MAXDIM];
```


Esercizio: Matrici

```
#include <stdio.h>
#include <string.h>

#define MAXDIM 10

int main(){
    int matrice[MAXDIM][MAXDIM];
    int righe[MAXDIM];
    int colonne[MAXDIM];
```


Esercizio: Matrici

```
#include <stdio.h>
#include <string.h>

#define MAXDIM 10

int main(){
    int matrice[MAXDIM][MAXDIM];
    int righe[MAXDIM];
    int colonne[MAXDIM];
    int sup = 0, inf = 0, diag = 0, dim = 0, i = 0, j = 0;
```

Esercizio: Matrici

```
#include <stdio.h>
#include <string.h>
```

```
#define MAXDIM 10
```

```
int main(){
    int matrice[MAXDIM][MAXDIM];
    int righe[MAXDIM];
    int colonne[MAXDIM];
    int sup = 0, inf = 0, diag = 0, dim = 0, i = 0, j = 0;
```

```
printf("inserire il numero di righe della matrice (massimo 10)\n");
scanf("%d", &dim);
```

Esercizio: Matrici

```
#include <stdio.h>
#include <string.h>

#define MAXDIM 10

int main(){
    int matrice[MAXDIM][MAXDIM];
    int righe[MAXDIM];
    int colonne[MAXDIM];
    int sup = 0, inf = 0, diag = 0, dim = 0, i = 0, j = 0;
```

```
do{
    printf("inserire il numero di righe della matrice (massimo 10)\n");
    scanf("%d", &dim);
}while(dim < 1 || dim > 10);
```

Esercizio: Matrici

```
#include <stdio.h>
#include <string.h>

#define MAXDIM 10

int main(){
    int matrice[MAXDIM][MAXDIM];
    int righe[MAXDIM];
    int colonne[MAXDIM];
    int sup = 0, inf = 0, diag = 0, dim = 0, i = 0, j = 0;
```

```
do{
    printf("inserire il numero di righe della matrice (massimo 10)\n");
    scanf("%d", &dim);
}while(dim < 1 || dim > 10);
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){
```

```
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/
```

```
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){
```

```
}
```

```
}
```


Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);
```

```
}
```

```
}
```


Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);
```

```
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori */  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
  
    }  
    printf("\n");  
}
```


Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```


Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori */  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori */  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```


Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```


Esercizio: Matrici

```
/** stampare la matrice a video e calcolare le somme di righe, colonne,  
    diagonale e triangolari superiori ed inferiori **/  
for(i=0; i<dim; i++){  
    for(j=0; j<dim; j++){  
        printf("%d ", matrice[i][j]);  
  
        righe[i] = righe[i] + matrice[i][j];  
  
        colonne[j] = colonne[j] + matrice[i][j];  
  
        if(i==j)  
            diag = diag + matrice[i][j];  
        else if (i < j)  
            sup = sup + matrice[i][j];  
        else  
            inf = inf + matrice[i][j];  
    }  
    printf("\n");  
}
```

Esercizio: Matrici

```
/** stampare le somme delle righe**/
```

Esercizio: Matrici

```
/** stampare le somme delle righe**/  
for(i=0; i<dim; i++){
```

Esercizio: Matrici

```
/** stampare le somme delle righe**/  
for(i=0; i<dim; i++){  
    printf("somma riga %d = %d\n", i+1, righe[i]);  
}
```

Esercizio: Matrici

```
/** stampare le somme delle righe**/  
for(i=0; i<dim; i++){  
    printf("somma riga %d = %d\n", i+1, righe[i]);  
}
```

```
/** stampare le somme delle colonne**/
```

Esercizio: Matrici

```
/** stampare le somme delle righe**/  
for(i=0; i<dim; i++){  
    printf("somma riga %d = %d\n", i+1, righe[i]);  
}
```

```
/** stampare le somme delle colonne**/  
for(j=0; j<dim; j++){
```

Esercizio: Matrici

```
/** stampare le somme delle righe**/
for(i=0; i<dim; i++){
    printf("somma riga %d = %d\n", i+1, righe[i]);
}

/** stampare le somme delle colonne**/
for(j=0; j<dim; j++){
    printf("somma colonna %d = %d\n", j+1, colonne[j]);
}
```


Esercizio: Matrici

```
/** stampare le somme delle righe**/
for(i=0; i<dim; i++){
    printf("somma riga %d = %d\n", i+1, righe[i]);
}

/** stampare le somme delle colonne**/
for(j=0; j<dim; j++){
    printf("somma colonna %d = %d\n", j+1, colonne[j]);
}

printf("la somma delle diagonale e': %d\n", diag);
printf("la somma della parte superiore alla diagonale e': %d\n", sup);
printf("la somma della parte inferiore alla diagonale e': %d\n", inf);
```

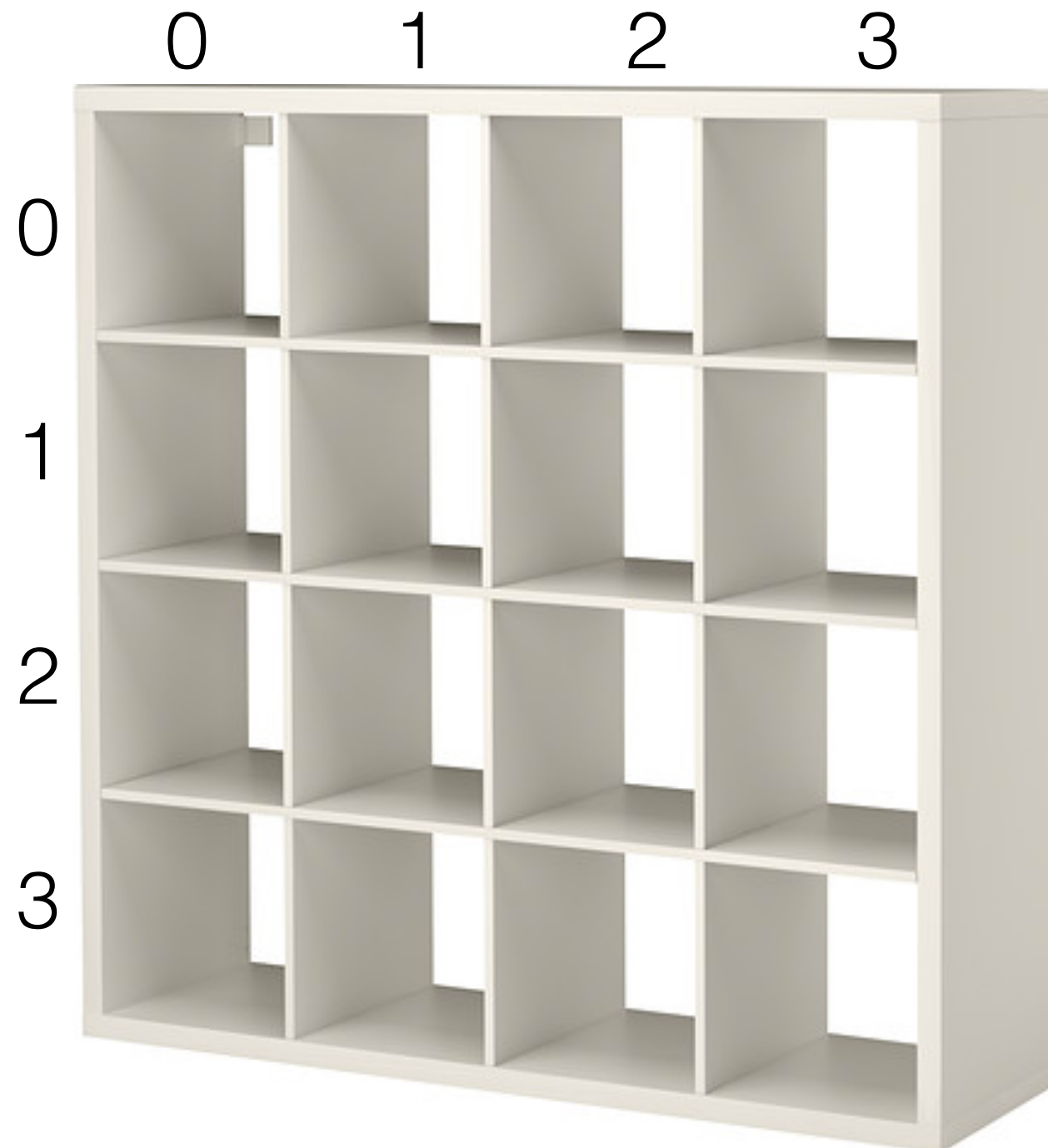

Matrici nel mondo reale

Matrici nel mondo reale

```
int matrice[4][4];
```

Matrici nel mondo reale

```
int matrice[4][4];
```



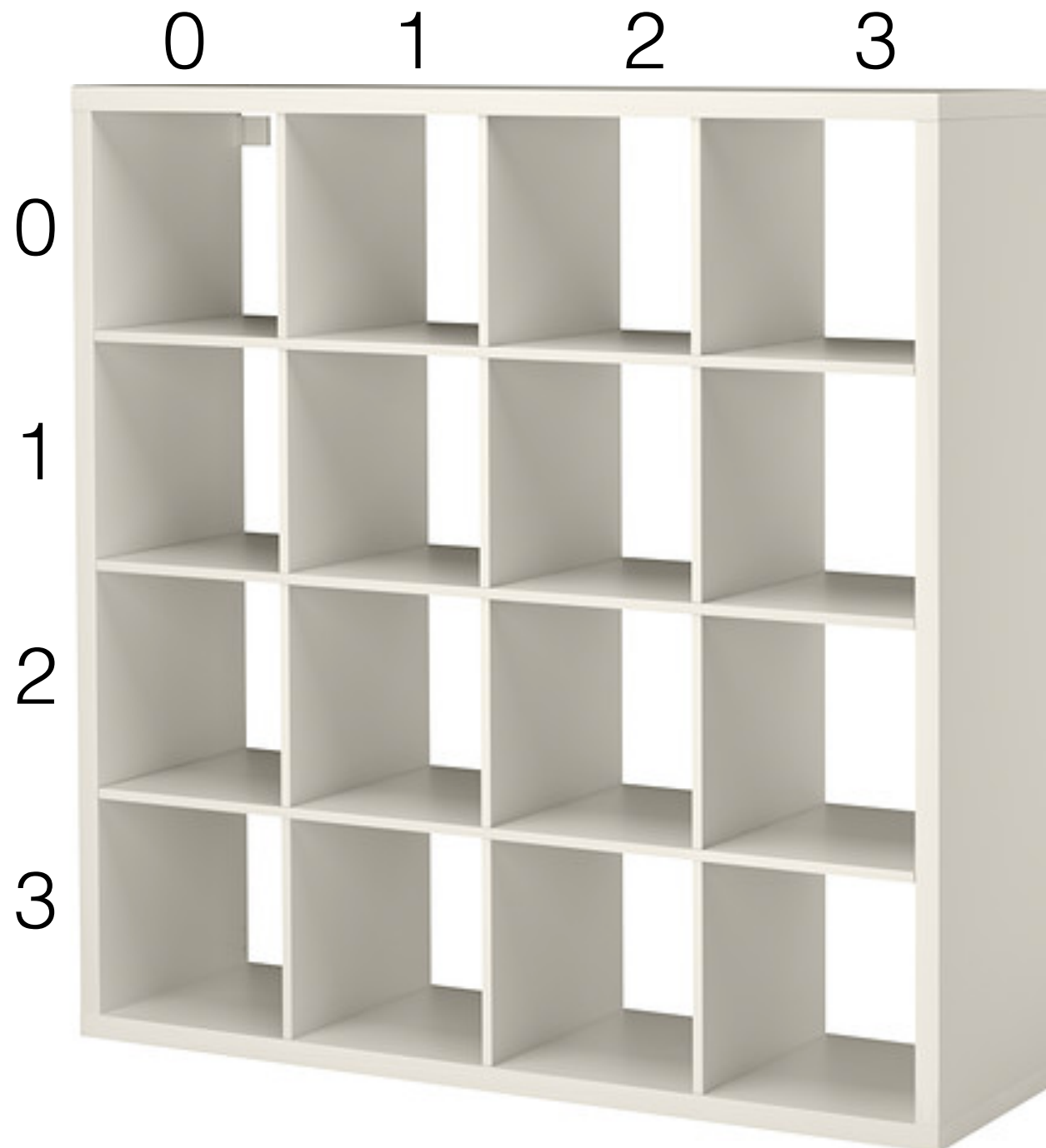
Matrici nel mondo reale

Matrici nel mondo reale

```
int matrice[4][4][4];
```

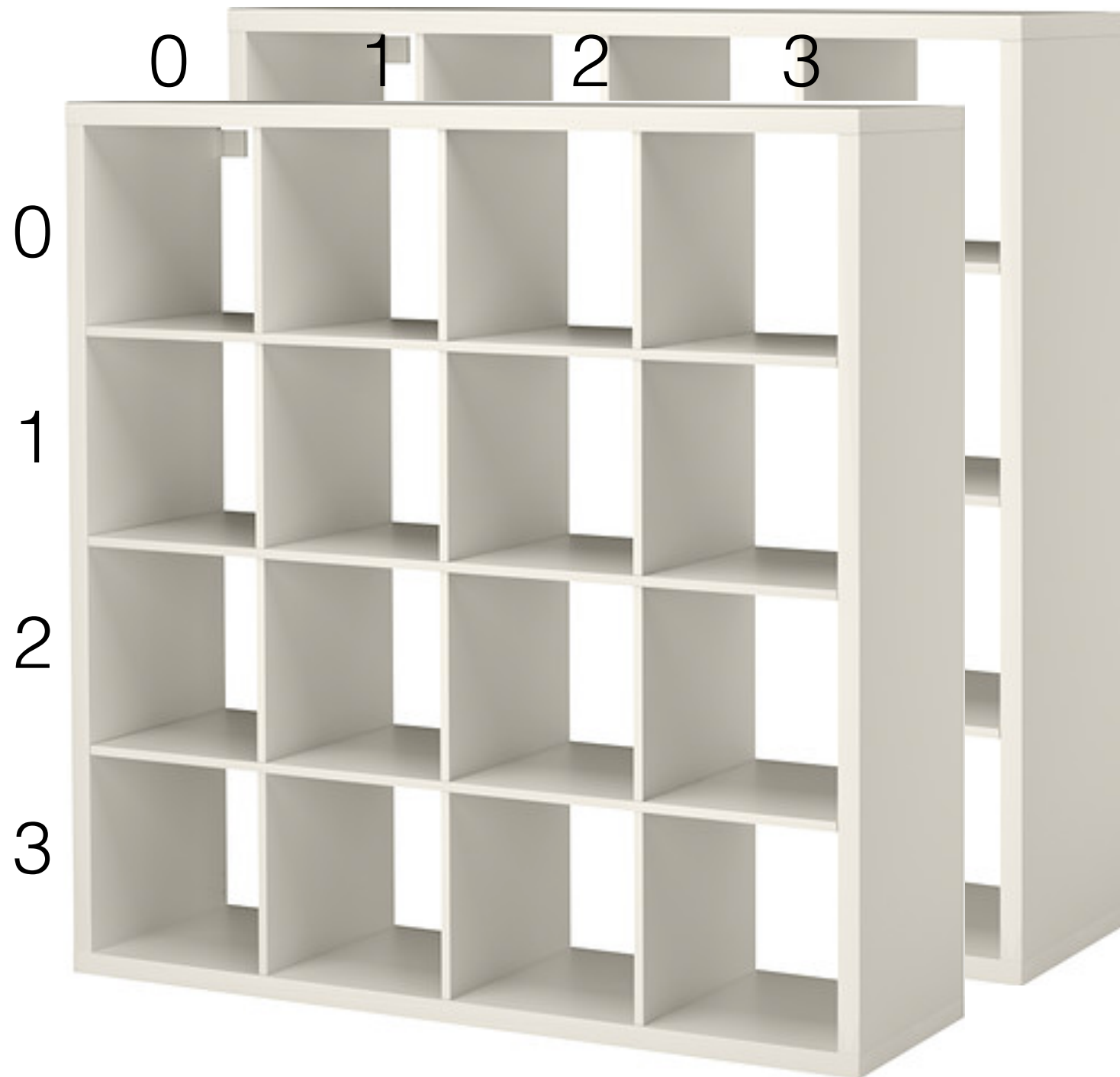
Matrici nel mondo reale

```
int matrice[4][4][4];
```



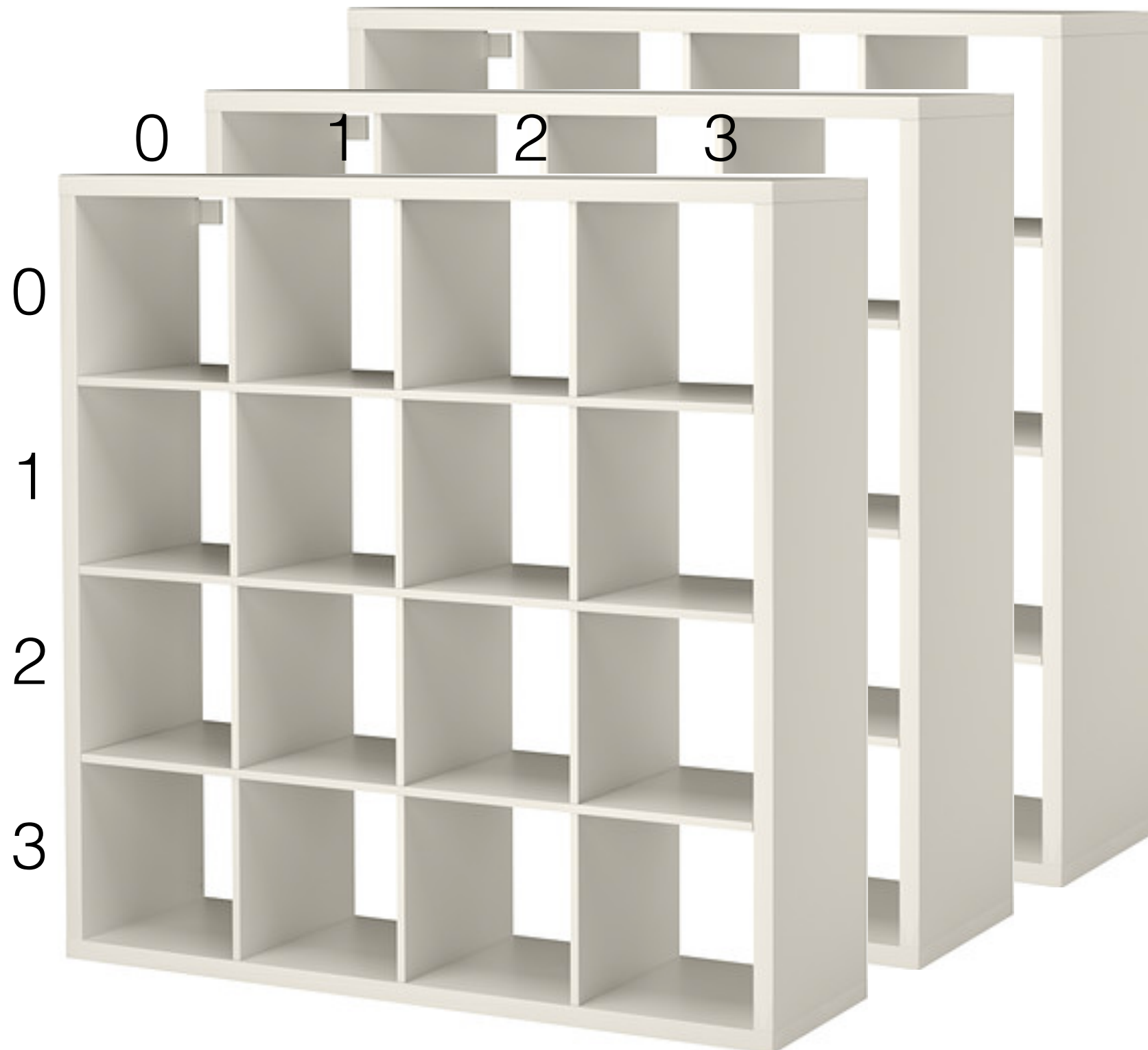
Matrici nel mondo reale

```
int matrice[4][4][4];
```



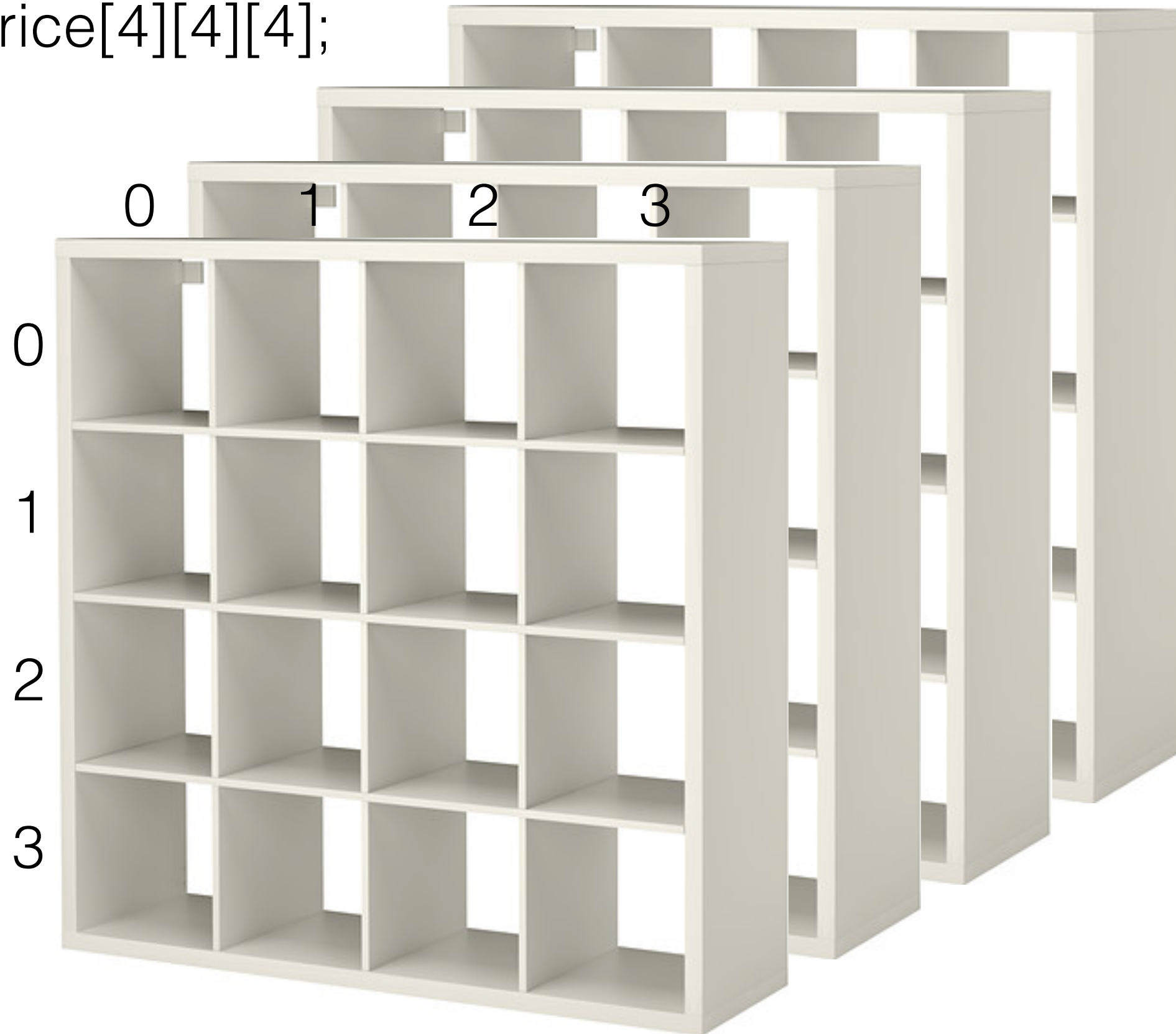
Matrici nel mondo reale

```
int matrice[4][4][4];
```



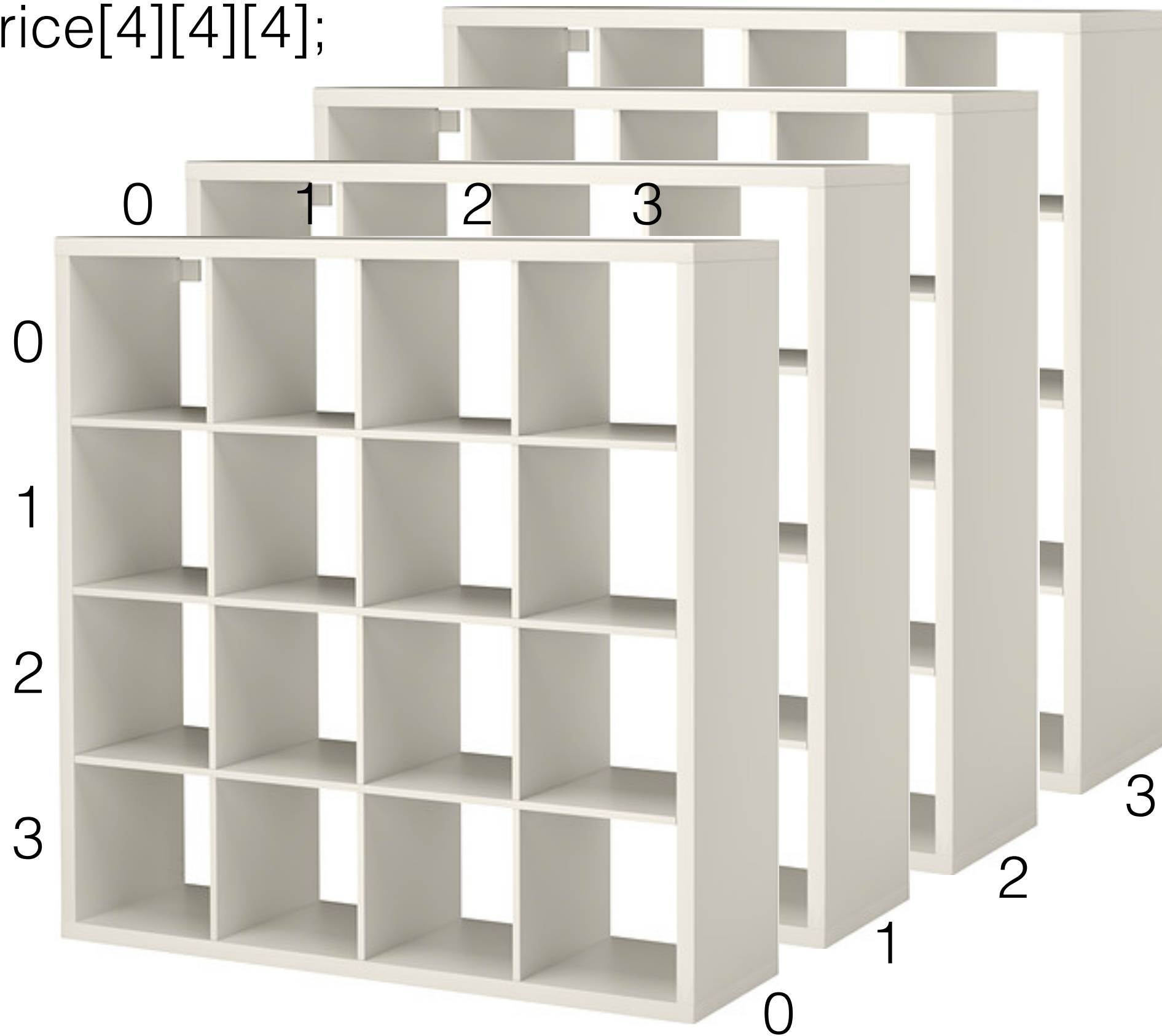
Matrici nel mondo reale

```
int matrice[4][4][4];
```



Matrici nel mondo reale

```
int matrice[4][4][4];
```



Esercizio: Matrici multidimensionali

Sia dato un array multidimensionale `char a[1000][34][131]`. Contare tutte le occorrenze delle sole vocali e stamparne a schermo la quantità, per ogni vocale.

Esercizio: Matrici multidimensionali

Sia dato un array multidimensionale `char a[1000][34][131]`. Contare tutte le occorrenze delle sole vocali e stamparne a schermo la quantità, per ogni vocale.

```
#define X 1000
#define Y 34
#define Z 131

char a[X][Y][Z];
int vocali[5]={0};
int x,y,z;
```

Esercizio: Matrici multidimensionali

Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){
```

```
}
```

Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){  
    for(y=0; y<Y; y++){
```

```
    }  
}
```


Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){  
    for(y=0; y<Y; y++){  
        for(z=0 ; z<Z; z++){
```

```
        }  
    }  
}
```


Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){  
    for(y=0; y<Y; y++){  
        for(z=0 ; z<Z; z++){  
  
            if(a[x][y][z]=='a' || a[x][y][z]=='e' ||  
                a[x][y][z]=='i' || a[x][y][z]=='o' ||  
                a[x][y][z]=='u' ){
```

```
            }  
        }  
    }  
}
```

Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){  
    for(y=0; y<Y; y++){  
        for(z=0 ; z<Z; z++){  
  
            if(a[x][y][z]=='a' || a[x][y][z]=='e' ||  
                a[x][y][z]=='i' || a[x][y][z]=='o' ||  
                a[x][y][z]=='u' ){  
  
                switch(a[x][y][z]){
```

```
                }  
            }  
        }  
    }  
}
```

Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){  
    for(y=0; y<Y; y++){  
        for(z=0 ; z<Z; z++){  
  
            if(a[x][y][z]=='a' || a[x][y][z]=='e' ||  
               a[x][y][z]=='i' || a[x][y][z]=='o' ||  
               a[x][y][z]=='u' ){  
  
                switch(a[x][y][z]){  
                    case 'a': vocali[0]++; break;
```

```
                }  
            }  
        }  
    }  
}
```

Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){  
    for(y=0; y<Y; y++){  
        for(z=0 ; z<Z; z++){  
  
            if(a[x][y][z]=='a' || a[x][y][z]=='e' ||  
                a[x][y][z]=='i' || a[x][y][z]=='o' ||  
                a[x][y][z]=='u' ){  
  
                switch(a[x][y][z]){  
                    case 'a': vocali[0]++; break;  
                    case 'e': vocali[1]++; break;
```

```
                }  
            }  
        }  
    }  
}
```

Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){  
    for(y=0; y<Y; y++){  
        for(z=0 ; z<Z; z++){  
  
            if(a[x][y][z]=='a' || a[x][y][z]=='e' ||  
                a[x][y][z]=='i' || a[x][y][z]=='o' ||  
                a[x][y][z]=='u' ){  
  
                switch(a[x][y][z]){  
                    case 'a': vocali[0]++; break;  
                    case 'e': vocali[1]++; break;  
                    case 'i': vocali[2]++; break;  

```

```
                }  
            }  
        }  
    }  
}
```

Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){
    for(y=0; y<Y; y++){
        for(z=0 ; z<Z; z++){

            if(a[x][y][z]=='a' || a[x][y][z]=='e' ||
               a[x][y][z]=='i' || a[x][y][z]=='o' ||
               a[x][y][z]=='u' ){

                switch(a[x][y][z]){
                    case 'a': vocali[0]++; break;
                    case 'e': vocali[1]++; break;
                    case 'i': vocali[2]++; break;
                    case 'o': vocali[3]++; break;
```

```
                }
            }
        }
    }
}
```

Esercizio: Matrici multidimensionali

```
for(x=0; x<X; x++){
    for(y=0; y<Y; y++){
        for(z=0 ; z<Z; z++){

            if(a[x][y][z]=='a' || a[x][y][z]=='e' ||
               a[x][y][z]=='i' || a[x][y][z]=='o' ||
               a[x][y][z]=='u' ){

                switch(a[x][y][z]){
                    case 'a': vocali[0]++; break;
                    case 'e': vocali[1]++; break;
                    case 'i': vocali[2]++; break;
                    case 'o': vocali[3]++; break;
                    case 'u': vocali[4]++; break;
                }
            }
        }
    }
}
```


Esercizio: Matrici multidimensionali

```
printf("\n\nLe a sono: %d", vocali[0]);  
printf("\nLe e sono: %d", vocali[1]);  
printf("\nLe i sono: %d", vocali[2]);  
printf("\nLe o sono: %d", vocali[3]);  
printf("\nLe u sono: %d\n\n", vocali[4]);
```


Struct nel mondo reale

Struct nel mondo reale



Struct nel mondo reale

```
int numero;  
char stringa[10];
```



Struct nel mondo reale

```
int numero;  
char stringa[10];
```



Struct nel mondo reale

```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

```
int numero;  
char stringa[10];
```



Struct nel mondo reale

```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

```
scatola laMiaNuovaScatola;
```

```
int numero;  
char stringa[10];
```



Struct nel mondo reale

```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

```
scatola laMiaNuovaScatola;
```

```
int numero;  
char stringa[10];
```



42

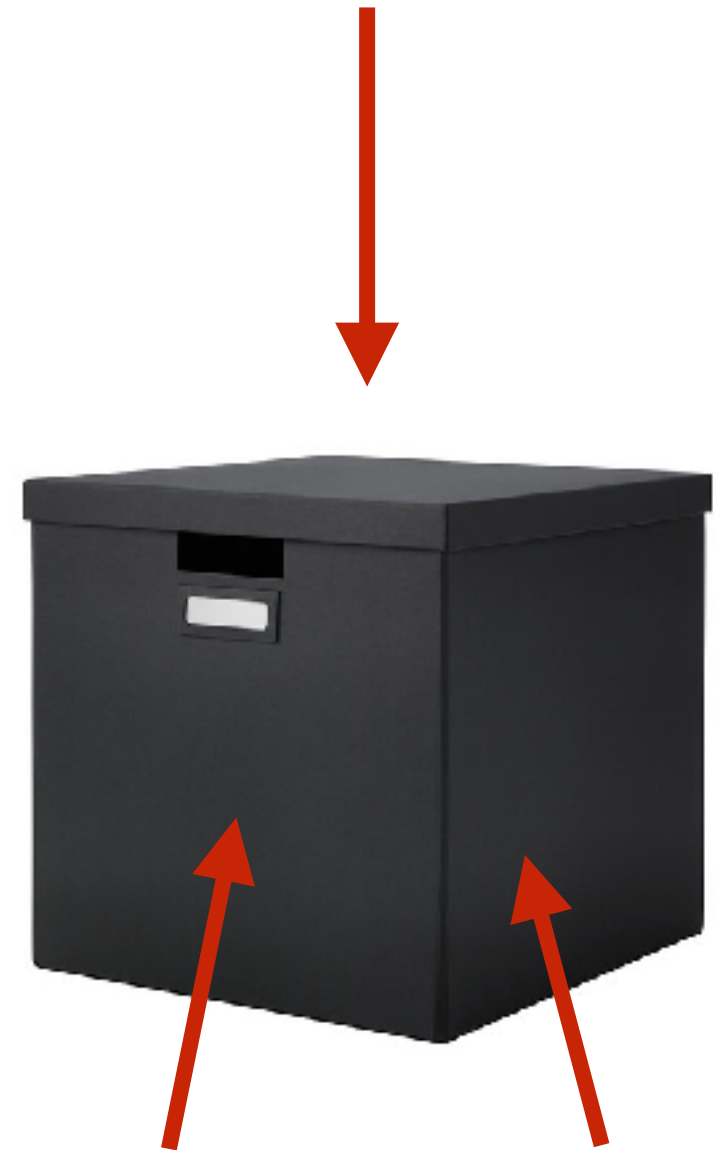


Struct nel mondo reale

```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

scatola laMiaNuovaScatola;

```
int numero;  
char stringa[10];
```



42

“ciao”

Struct nel mondo reale

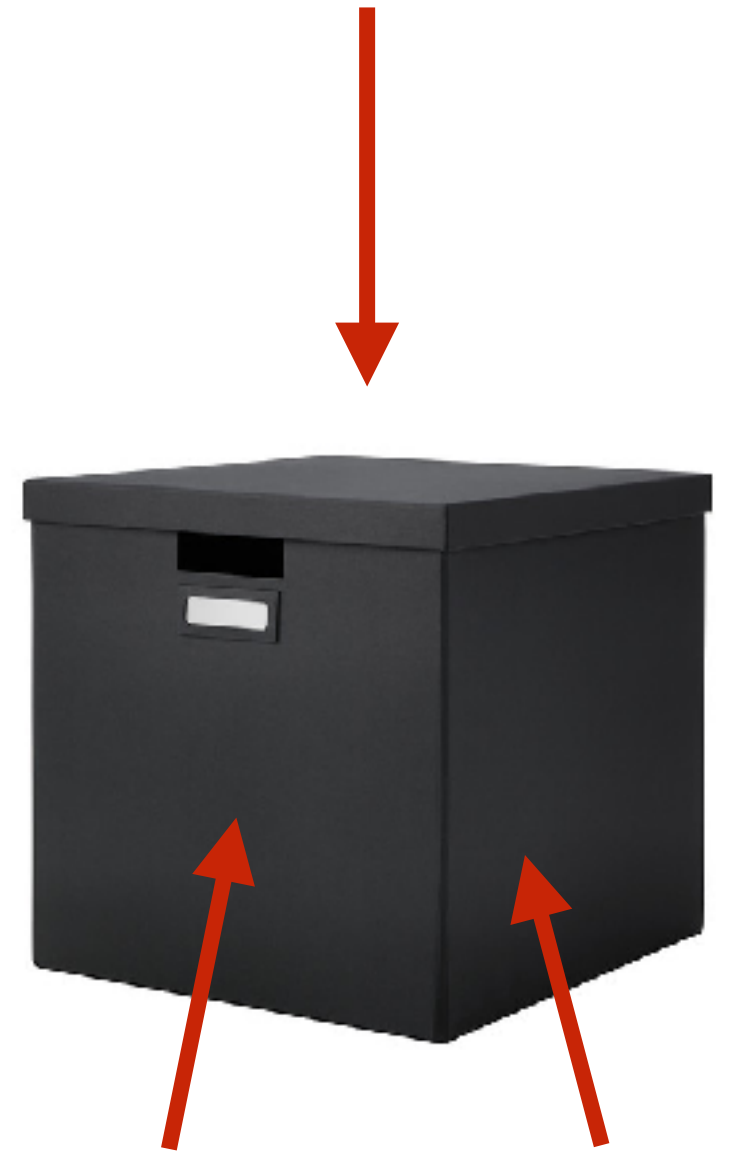
```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

scatola laMiaNuovaScatola;

laMiaNuovaScatola.numero = 42;

strcpy(laMiaNuovaScatola.stringa, "ciao");

```
int numero;  
char stringa[10];
```



42

"ciao"

Matrici nel mondo reale

Matrici nel mondo reale

```
int matrice[4][4];
```

Matrici nel mondo reale

```
int matrice[4][4];
```



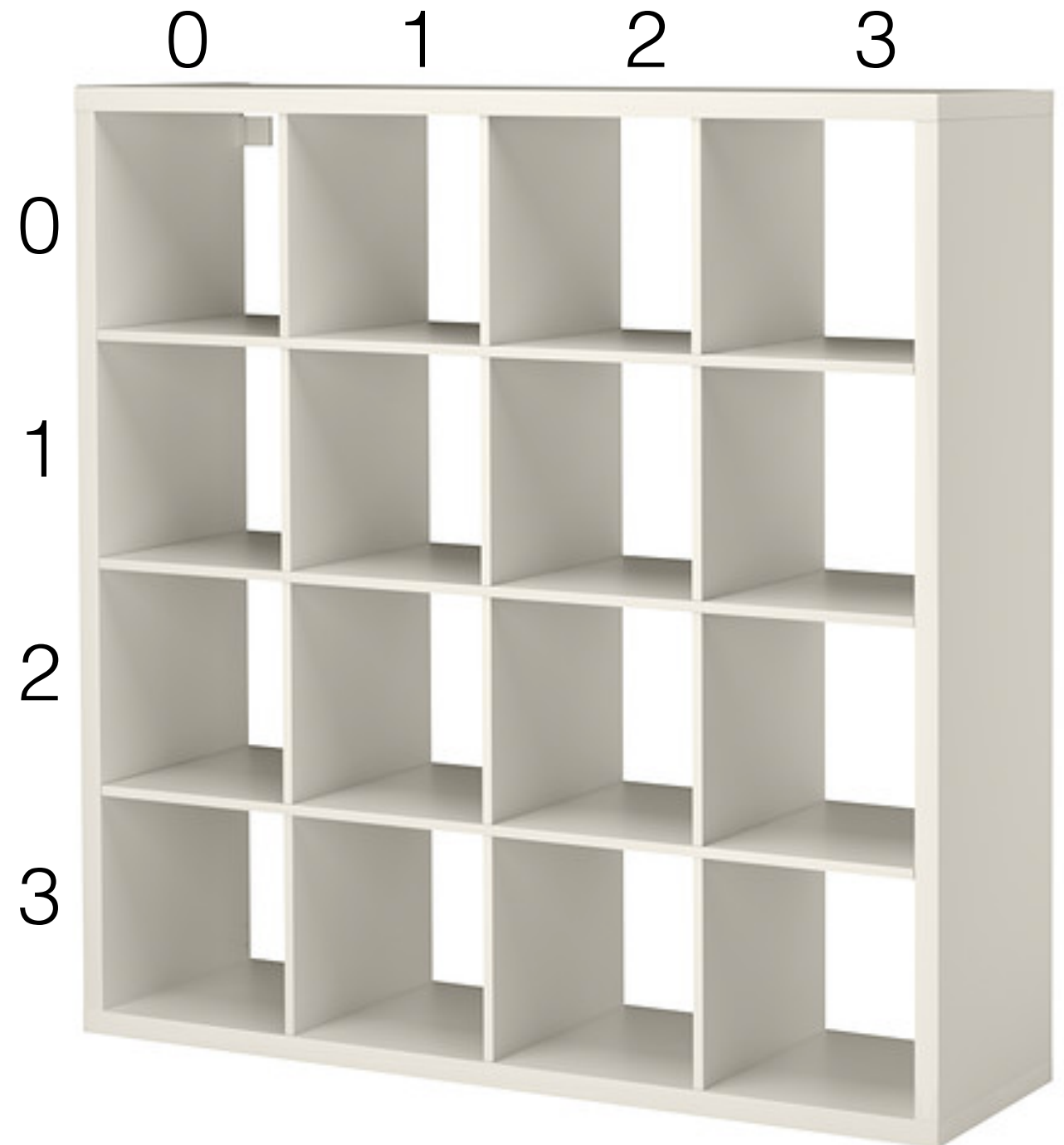
Matrici nel mondo reale

```
int matrice[4][4];
```



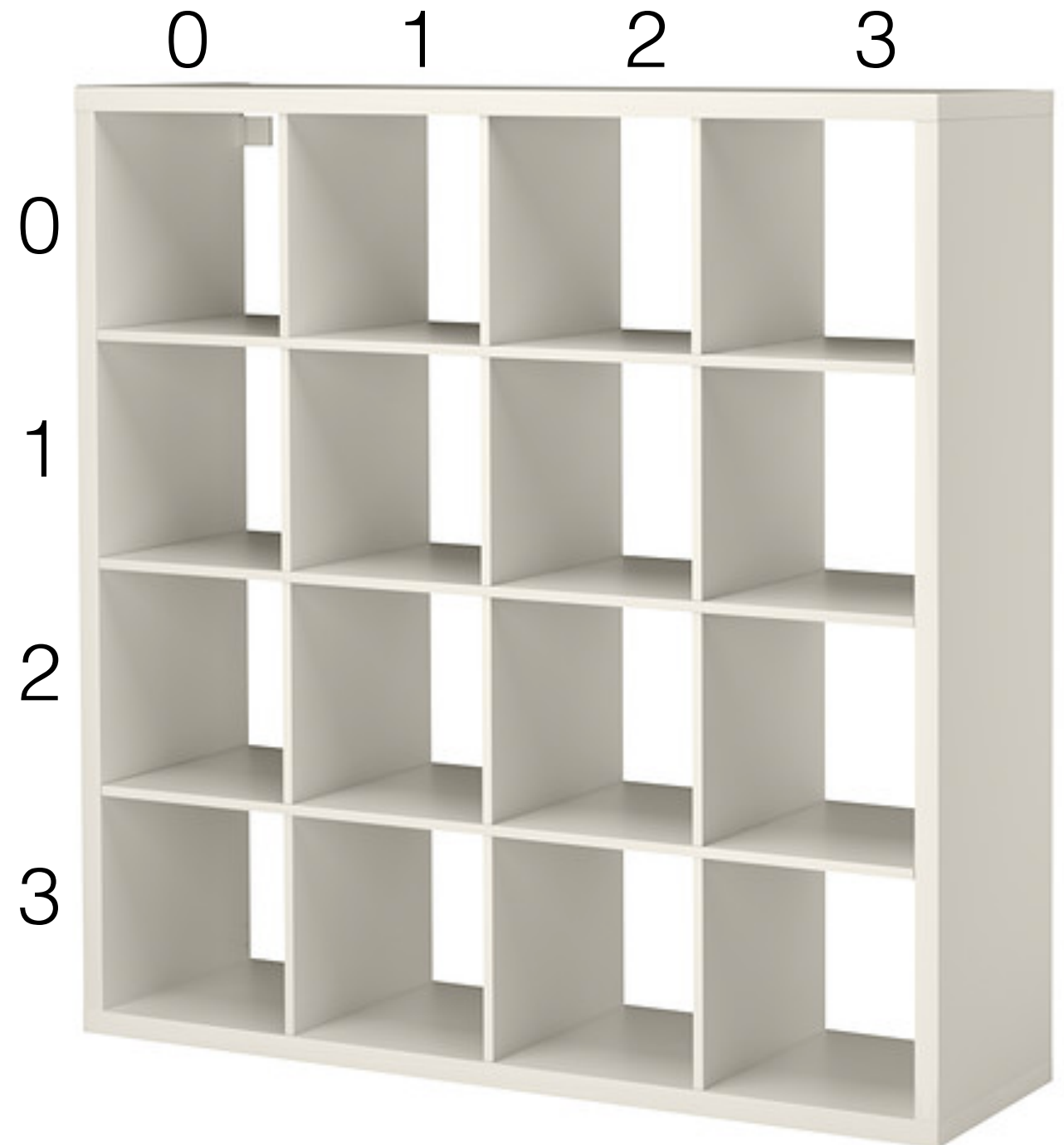
Matrici nel mondo reale

```
int matrice[4][4];
```



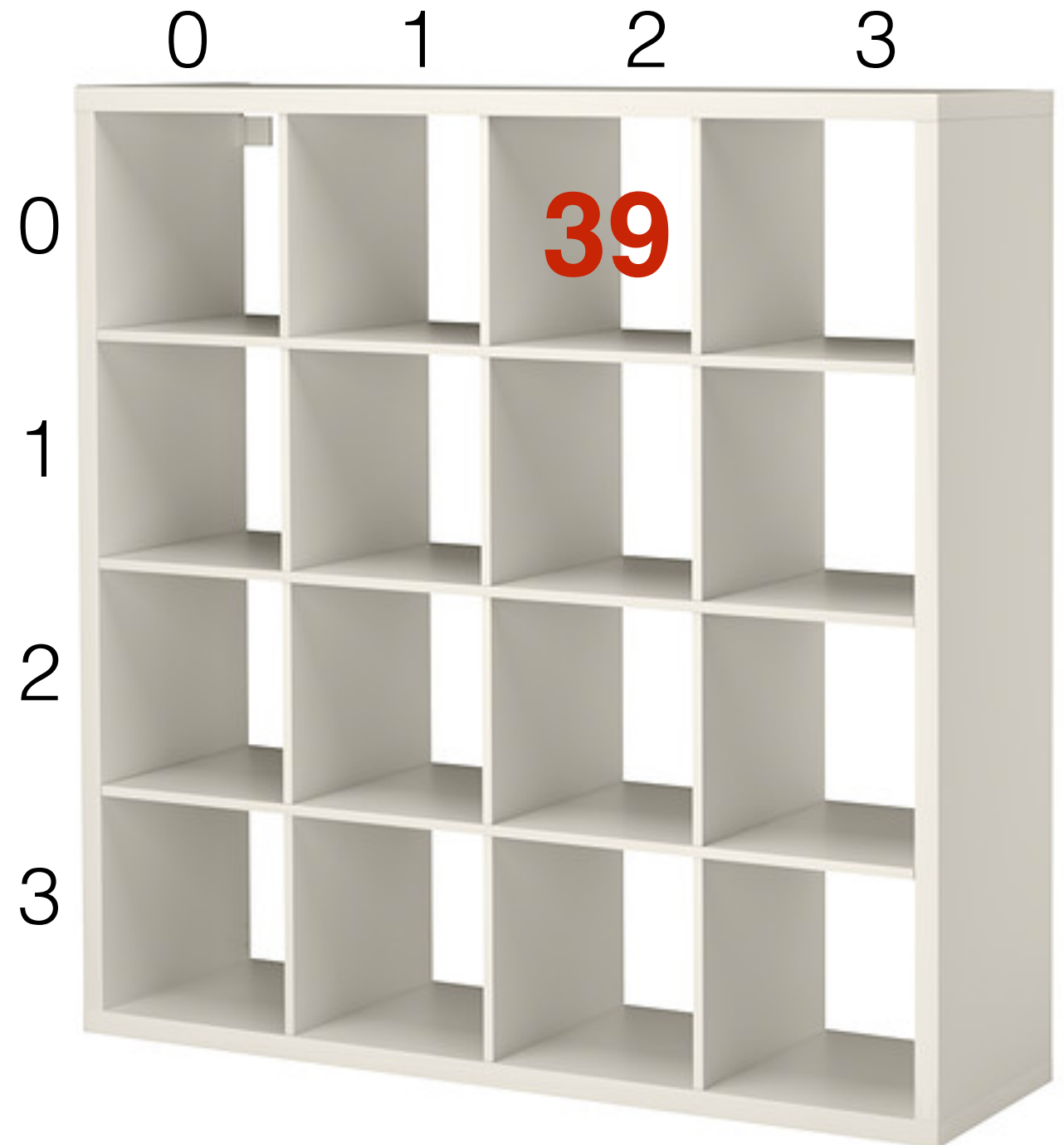
Matrici nel mondo reale

```
int matrice[4][4];  
matrice[0][2] = 39;
```



Matrici nel mondo reale

```
int matrice[4][4];  
matrice[0][2] = 39;
```



Matrici e struct nel mondo reale

Matrici e struct nel mondo reale

scatola matrice[4][4];

Matrici e struct nel mondo reale

scatola matrice[4][4];



Matrici e struct nel mondo reale

scatola matrice[4][4];



Matrici e struct nel mondo reale

scatola matrice[4][4];



Matrici e struct nel mondo reale

```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

```
scatola matrice[4][4];
```

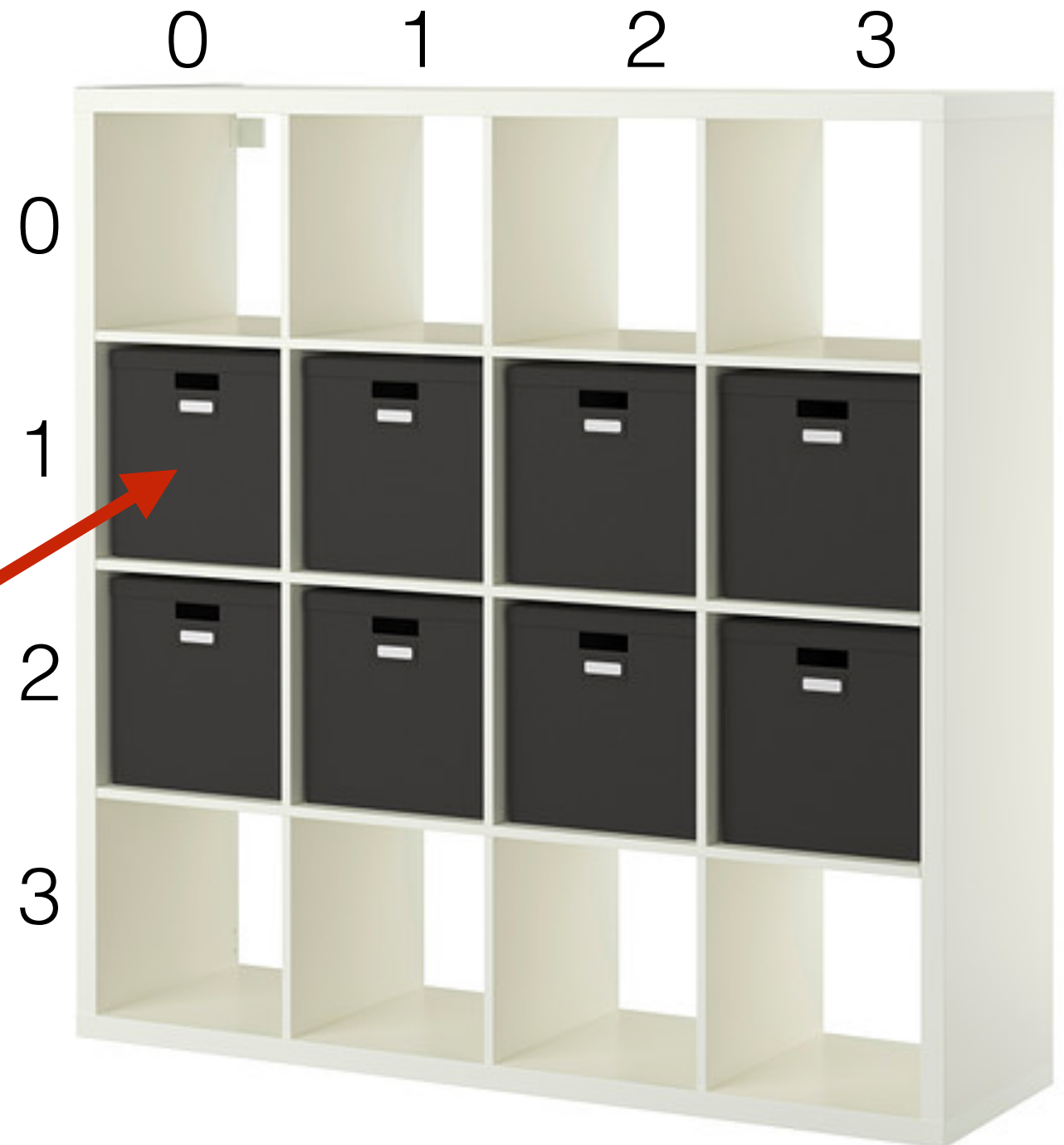


Matrici e struct nel mondo reale

```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

scatola matrice[4][4];

42

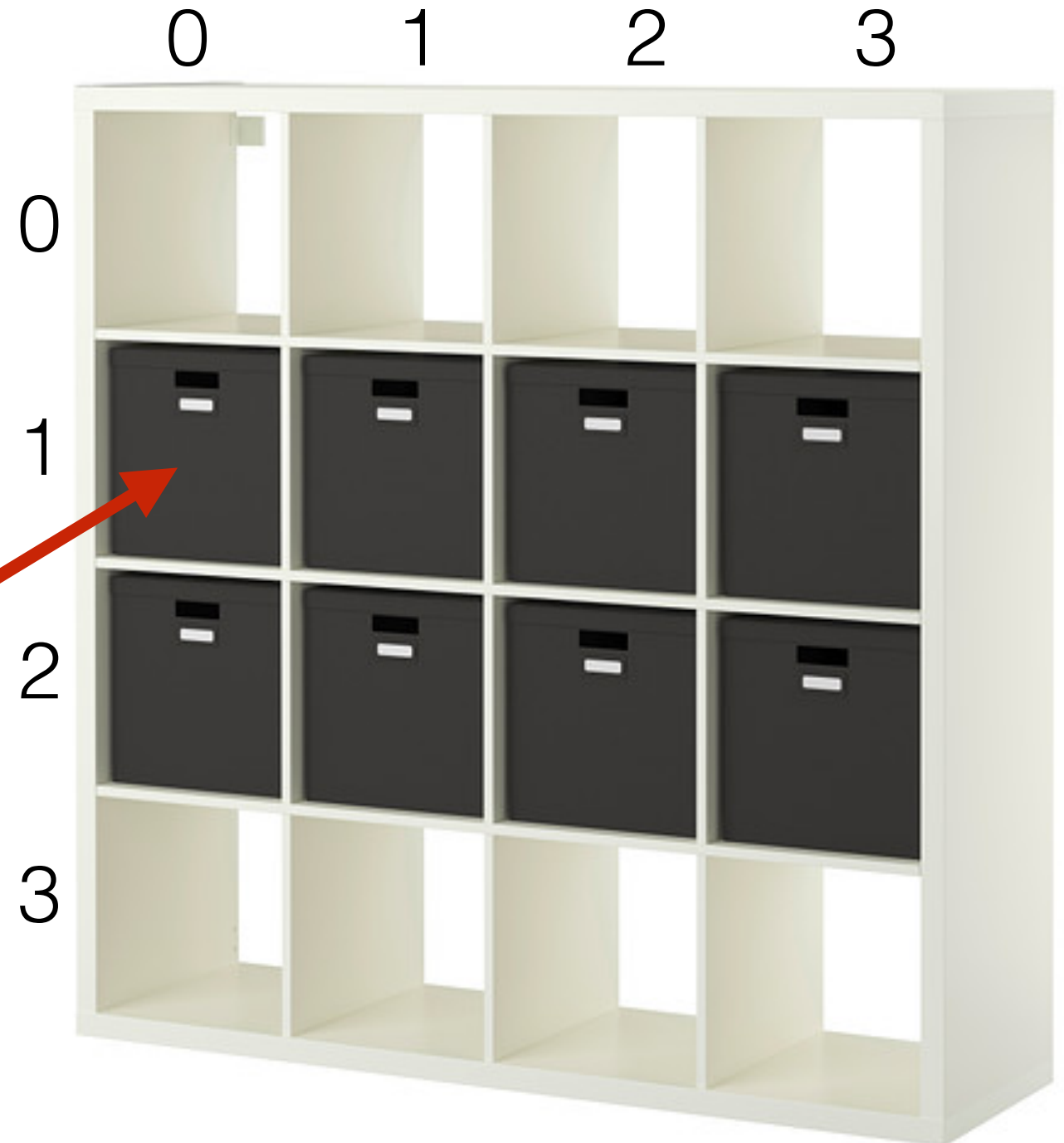


Matrici e struct nel mondo reale

```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

```
scatola matrice[4][4];
```

42



```
matrice[1][0].numero = 42;
```


Matrici e struct nel mondo reale

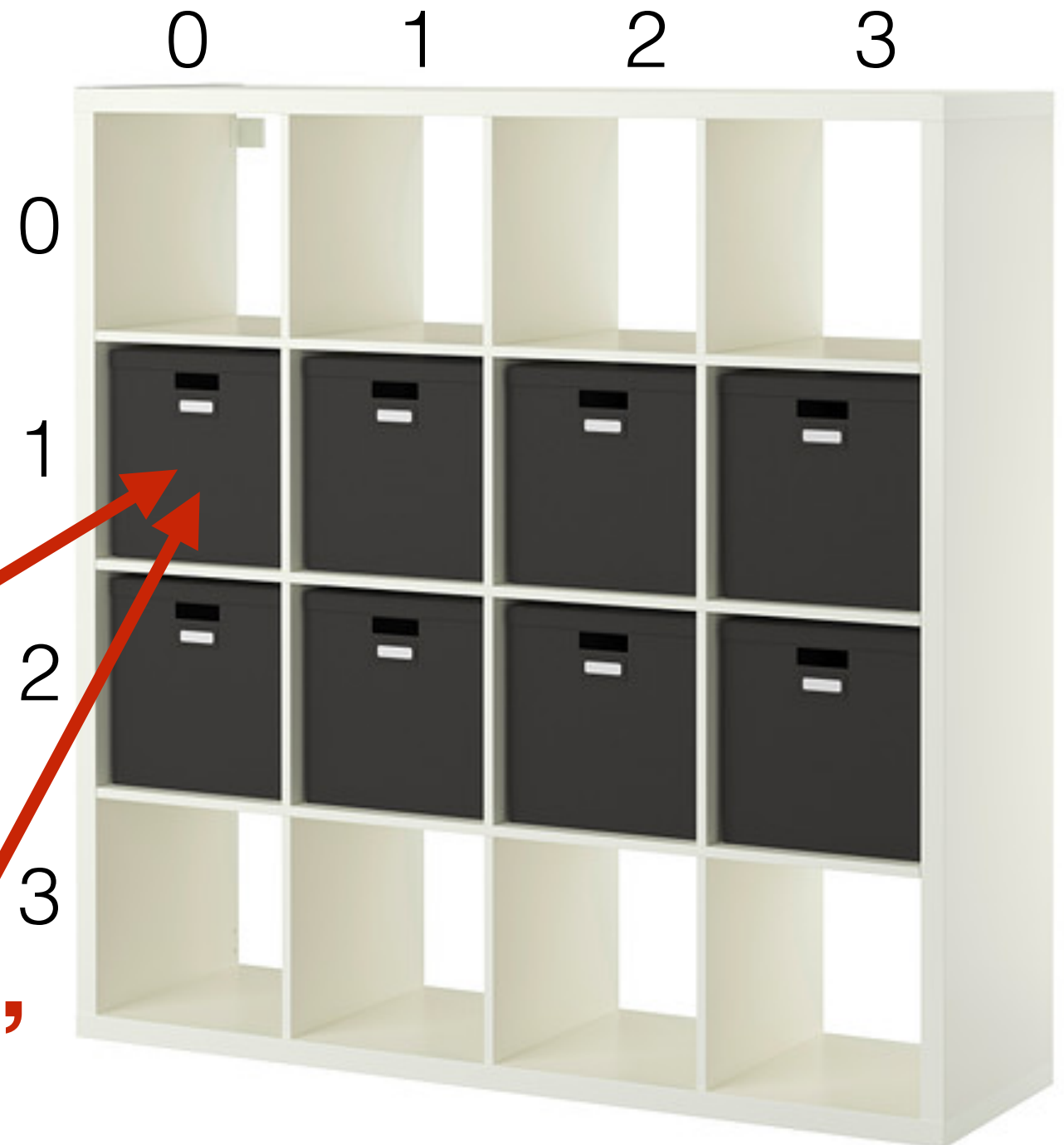
```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

scatola matrice[4][4];

42

“ciao”

matrice[1][0].numero = 42;



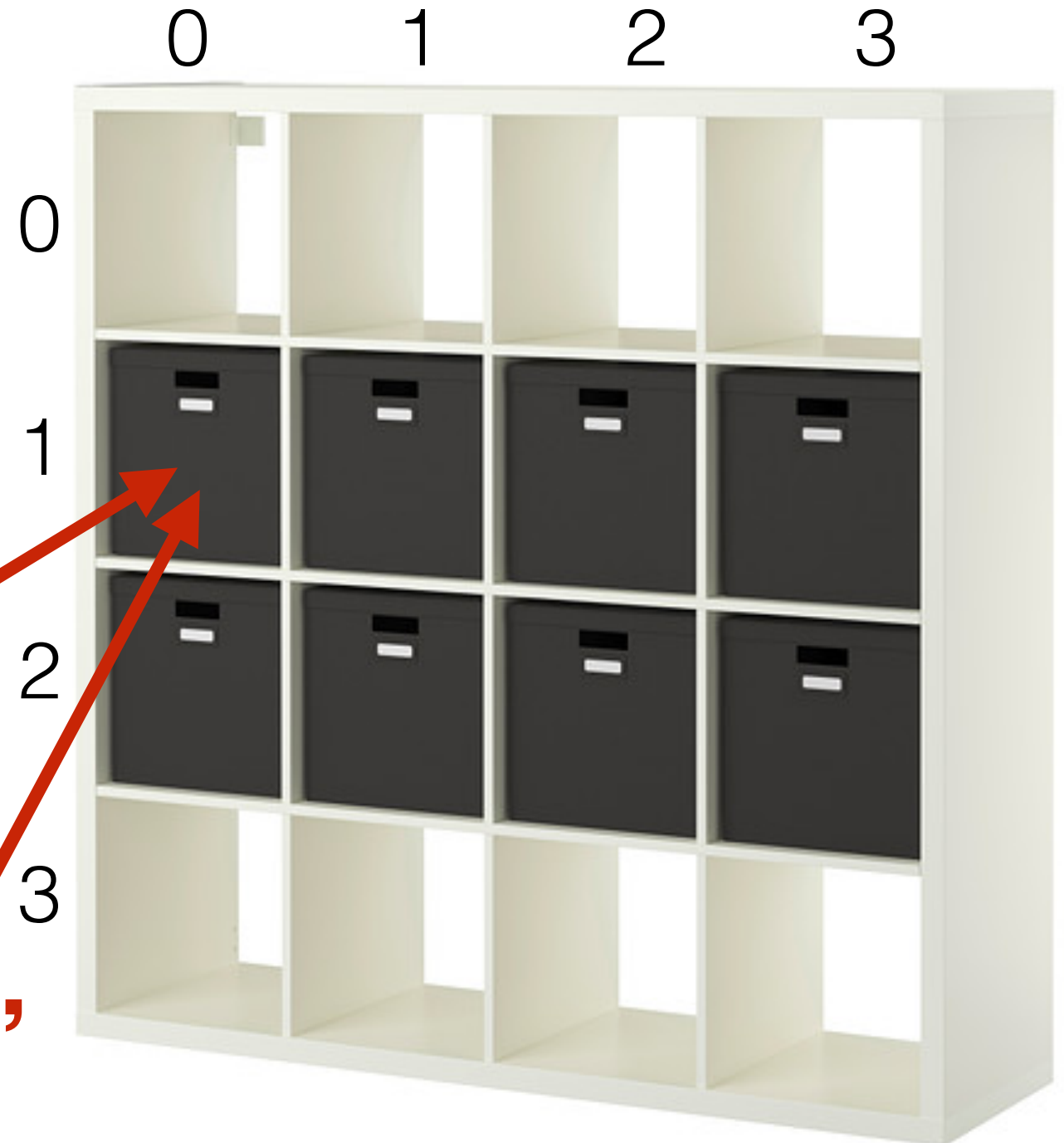
Matrici e struct nel mondo reale

```
typedef struct {  
    int numero;  
    char stringa[10];  
} scatola;
```

scatola matrice[4][4];

42

“ciao”



```
matrice[1][0].numero = 42;  
strcpy(matrice[1][0].stringa, “ciao”);
```

Esercizio: Agenda

- Uno studio medico richiede di realizzare una piccola agenda degli appuntamenti settimanali
- Per semplicità si considerino solo i giorni e le ore lavorativi (5 giorni a settimana, dalle 8 alle 17)
- Sempre per semplicità, gli appuntamenti vengono allocati su base oraria
- Per ogni appuntamento occorre memorizzare nome e cognome del paziente, prestazione richiesta, eventuali note e la cifra pagata

Si devono poter effettuare le seguenti operazioni:

- Inserire un nuovo appuntamento
- Vedere tutti gli appuntamenti di un giorno
- Inserire un pagamento
- Stampare una “ricevuta” di pagamento

Esercizio: Agenda

- Uno studio medico richiede di realizzare una piccola agenda degli appuntamenti settimanali
- Per semplicità si considerino solo i giorni e le ore lavorativi (5 giorni a settimana, dalle 8 alle 17)
- Sempre per semplicità, gli appuntamenti vengono allocati su base oraria
- Per ogni appuntamento occorre memorizzare nome e cognome del paziente, prestazione richiesta, eventuali note e la cifra pagata

Si devono poter effettuare le seguenti operazioni:

- Inserire un nuovo appuntamento
- Vedere tutti gli appuntamenti di un giorno
- Inserire un pagamento
- Stampare una “ricevuta” di pagamento

Esercizio: Agenda

- Uno studio medico richiede di realizzare una piccola agenda degli appuntamenti settimanali
- Per semplicità si considerino solo i giorni e le ore lavorativi (5 giorni a settimana, dalle 8 alle 17)
- Sempre per semplicità, gli appuntamenti vengono allocati su base oraria
- Per ogni appuntamento occorre memorizzare nome e cognome del paziente, prestazione richiesta, eventuali note e la cifra pagata

Esercizio: Agenda

- Uno studio medico richiede di realizzare una piccola **agenda** degli appuntamenti settimanali
- Per semplicità si considerino solo i giorni e le ore lavorativi (**5 giorni** a settimana, **dalle 8 alle 17**)
- Sempre per semplicità, gli appuntamenti vengono **allocati su base oraria**

Agenda: Struttura dati

```
#include <stdio.h>
#include <string.h>

#define GIORNI 5
#define ORE 10
```

```
typedef struct {
```

```
} Prenotazione;
```

```
Prenotazione agendaSett[GIORNI][ORE];
```


Esercizio: Agenda

- Uno studio medico richiede di realizzare una piccola agenda degli appuntamenti settimanali
- Per semplicità si considerino solo i giorni e le ore lavorativi (5 giorni a settimana, dalle 8 alle 17)
- Sempre per semplicità, gli appuntamenti vengono allocati su base oraria
- **Per ogni appuntamento** occorre memorizzare **nome** e **cognome** del paziente, **prestazione** richiesta, eventuali **note** e la **cifra** pagata

Agenda: Struttura dati

```
#include <stdio.h>
#include <string.h>

#define GIORNI 5
#define ORE 10
```

```
typedef struct {
```

```
    char paziente[40];
    char tipoPrestazione[100];
    char note[75];
    float pagato;
```

```
} Prenotazione;
```

```
Prenotazione agendaSett[GIORNI][ORE];
```

Esercizio: Agenda

- Uno studio medico richiede di realizzare una piccola agenda degli appuntamenti settimanali
- Per semplicità si considerino solo i giorni e le ore lavorativi (5 giorni a settimana, dalle 8 alle 17)
- Sempre per semplicità, gli appuntamenti vengono allocati su base oraria
- Per ogni appuntamento occorre memorizzare nome e cognome del paziente, prestazione richiesta, eventuali note e la cifra pagata

Piccolo suggerimento:

*vi servirà anche uno “**stato**” (**typedef enum?**)*⁸⁰

Agenda: Struttura dati

```
#include <stdio.h>
#include <string.h>

#define GIORNI 5
#define ORE 10
```

```
typedef enum {LIBERO, PRENOTATO, ESEGUITO} Stato;
typedef struct {
    char paziente[40];
    char tipoPrestazione[100];
    char note[75];
    float pagato;
    Stato stato;
} Prenotazione;
```

```
Prenotazione agendaSett[GIORNI][ORE];
int giorno = 0, ora = 0;
char scelta;
int giornoSel = -1, oraSel = -1;
```

Agenda: Struttura dati

```
#include <stdio.h>
#include <string.h>

#define GIORNI 5
#define ORE 10
```

```
typedef enum {LIBERO, PRENOTATO, ESEGUITO} Stato;
typedef struct {
    char paziente[40];
    char tipoPrestazione[100];
    char note[75];
    float pagato;
    Stato stato;
} Prenotazione;
```

*Cosa c'è nell'agenda,
ora?*

```
Prenotazione agendaSett[GIORNI][ORE];
int giorno = 0, ora = 0;
char scelta;
int giornoSel = -1, oraSel = -1;
```

Agenda: Inizializzazione

```
for(giorno = 0; giorno < GIORNI; giorno++) {  
    for(ora = 0; ora < ORE; ora++) {  
        strcpy(agendaSett[giorno][ora].paziente, "");  
        strcpy(agendaSett[giorno][ora].tipoPrestazione, "");  
        strcpy(agendaSett[giorno][ora].note, "");  
        agendaSett[giorno][ora].pagato = 0.0;  
        agendaSett[giorno][ora].stato = LIBERO;  
    }  
}
```


Agenda: Inizializzazione

```
for(giorno = 0; giorno < GIORNI; giorno++) {  
    for(ora = 0; ora < ORE; ora++) {  
        strcpy(agendaSett[giorno][ora].paziente, "");  
        strcpy(agendaSett[giorno][ora].tipoPrestazione, "");  
        strcpy(agendaSett[giorno][ora].note, "");  
        agendaSett[giorno][ora].pagato = 0.0;  
        agendaSett[giorno][ora].stato = LIBERO;  
    }  
}
```

Agenda: Inizializzazione

```
for(giorno = 0; giorno < GIORNI; giorno++) {  
    for(ora = 0; ora < ORE; ora++) {  
        strcpy(agendaSett[giorno][ora].paziente, "");  
        strcpy(agendaSett[giorno][ora].tipoPrestazione, "");  
        strcpy(agendaSett[giorno][ora].note, "");  
        agendaSett[giorno][ora].pagato = 0.0;  
        agendaSett[giorno][ora].stato = LIBERO;  
    }  
}
```

Esercizio: Agenda

- Uno studio medico richiede di realizzare una piccola agenda degli appuntamenti settimanali
- Per semplicità si considerino solo i giorni e le ore lavorativi (5 giorni a settimana, dalle 8 alle 17)
- Sempre per semplicità, gli appuntamenti vengono allocati su base oraria
- Per ogni appuntamento occorre memorizzare nome e cognome del paziente, prestazione richiesta, eventuali note e la cifra pagata

Si devono poter effettuare le seguenti operazioni:

- Inserire un nuovo appuntamento
- Vedere tutti gli appuntamenti di un giorno
- Inserire un pagamento
- Stampare una “ricevuta” di pagamento

Esercizio: Agenda

Si devono poter effettuare le seguenti operazioni:

- Inserire un nuovo appuntamento
- Vedere tutti gli appuntamenti di un giorno
- Inserire un pagamento
- Stampare una “ricevuta” di pagamento

Agenda: Menù principale

```
do {  
    printf("-----\n");  
    printf("Opzioni disponibili:\n");  
    printf("Inserisci appuntamento (i)\n");  
    printf("Visualizza appuntamenti di un giorno (v)\n");  
    printf("Inserisci pagamento (p)\n");  
    printf("Stampa ricevuta (s)\n");  
    printf("Operazione scelta: ");  
    scanf("%c", &scelta);  
    printf("-----\n");  
  
    switch(scelta) {  
    }while(scelta != 'q');
```

Agenda: Menù principale

```
do {  
    printf("-----\n");  
    printf("Opzioni disponibili:\n");  
    printf("Inserisci appuntamento (i)\n");  
    printf("Visualizza appuntamenti di un giorno (v)\n");  
    printf("Inserisci pagamento (p)\n");  
    printf("Stampa ricevuta (s)\n");  
    printf("Operazione scelta: ");  
    scanf("%c", &scelta);  
    printf("-----\n");  
    switch(scelta) {  
    }while(scelta != 'q');
```

Agenda: Switch

```
switch(scelta) {  
  
    case 'i':  
        printf("Inserimento nuovo appuntamento:\n");  
        do {  
            giornoSel = giornoSel - 1;  
        } do {  
            oraSel = oraSel - 8;  
            scanf("%*c"); //fflush(stdin);  
            if(agendaSett[giornoSel][oraSel].stato == LIBERO) {  
  
    case 'v':  
  
        if((giornoSel < 0)) {  
            printf("Appuntamenti del giorno %d:\n", giornoSel + 1);  
            for(ora = 0; ora < ORE; ora++) {  
                giornoSel = -1;  
                break;  
            }  
  
    case 'p':  
        printf("Registra pagamento:\n");  
        do {  
            giornoSel = giornoSel - 1;  
        } do {  
            oraSel = oraSel - 8;  
            scanf("%*c"); //fflush(stdin);  
            if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {  
  
    case 's':  
        printf("Stampa ricevuta:\n");  
        if((giornoSel < 0) || (oraSel < 0)) {  
            scanf("%*c"); //fflush(stdin);  
            if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {  
                giornoSel = -1;  
                oraSel = -1;  
                break;  
            }  
  
    case 'q':  
        printf("\nEsco dall'agenda, buon lavoro!\n");  
        break;  
    default:  
        printf("\nOperazione non consentita, inserire un'operazione valida.\n");  
  
}
```

Agenda: Switch

```
switch(scelta) {  
    case 'i':  
        printf("Inserimento nuovo appuntamento:\n");  
        do {  
            giornoSel = giornoSel - 1;  
        } do {  
            oraSel = oraSel - 8;  
            scanf("%*c"); //fflush(stdin);  
            if(agendaSett[giornoSel][oraSel].stato == LIBERO) {  
  
        case 'v':  
  
            if((giornoSel < 0)) {  
                printf("Appuntamenti del giorno %d:\n", giornoSel + 1);  
                for(ora = 0; ora < ORE; ora++) {  
                    giornoSel = -1;  
                    break;  
                }  
            }  
  
        case 'p':  
            printf("Registra pagamento:\n");  
            do {  
                giornoSel = giornoSel - 1;  
            } do {  
                oraSel = oraSel - 8;  
                scanf("%*c"); //fflush(stdin);  
                if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {  
  
        case 's':  
            printf("Stampa ricevuta:\n");  
            if((giornoSel < 0) || (oraSel < 0)) {  
                scanf("%*c"); //fflush(stdin);  
                if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {  
                    giornoSel = -1;  
                    oraSel = -1;  
                    break;  
                }  
            }  
  
        case 'q':  
            printf("\nEsco dall'agenda, buon lavoro!\n");  
            break;  
        default:  
            printf("\nOperazione non consentita, inserire un'operazione valida.\n");  
    }  
}
```

Agenda: Switch

```
switch(scelta) {
```

```
    case 'i':
        printf("Inserimento nuovo appuntamento:\n");
        do {
            giornoSel = giornoSel - 1;
        } do {
            oraSel = oraSel - 8;
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato == LIBERO) {
```

} Inserire un nuovo appuntamento

```
    case 'v':
        if((giornoSel < 0)) {
            printf("Appuntamenti del giorno %d:\n", giornoSel + 1);
            for(ora = 0; ora < ORE; ora++) {
                giornoSel = -1;
                break;
            }
        }
```

```
    case 'p':
        printf("Registra pagamento:\n");
        do {
            giornoSel = giornoSel - 1;
        } do {
            oraSel = oraSel - 8;
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {
```

```
    case 's':
        printf("Stampa ricevuta:\n");
        if((giornoSel < 0) || (oraSel < 0)) {
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {
                giornoSel = -1;
                oraSel = -1;
                break;
            }
        }
```

```
    case 'q':
        printf("\nEsco dall'agenda, buon lavoro!\n");
        break;
```

```
    default:
        printf("\nOperazione non consentita, inserire un'operazione valida.\n");
```

```
}
```


Agenda: Switch

```
switch(scelta) {
```

```
    case 'i':
        printf("Inserimento nuovo appuntamento:\n");
        do {
            giornoSel = giornoSel - 1;
        } do {
            oraSel = oraSel - 8;
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato == LIBERO) {
```

```
    case 'v':
        if((giornoSel < 0)) {
            printf("Appuntamenti del giorno %d:\n", giornoSel + 1);
            for(ora = 0; ora < ORE; ora++) {
                giornoSel = -1;
                break;
            }
        }
```

```
    case 'p':
        printf("Registra pagamento:\n");
        do {
            giornoSel = giornoSel - 1;
        } do {
            oraSel = oraSel - 8;
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {
```

```
    case 's':
        printf("Stampa ricevuta:\n");
        if((giornoSel < 0) || (oraSel < 0)) {
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {
                giornoSel = -1;
                oraSel = -1;
                break;
            }
        }
```

```
    case 'q':
        printf("\nEsco dall'agenda, buon lavoro!\n");
        break;
```

```
    default:
        printf("\nOperazione non consentita, inserire un'operazione valida.\n");
```

```
}
```

} Inserire un nuovo appuntamento

} Vedere tutti gli appuntamenti di un giorno

Agenda: Switch

```
switch(scelta) {
```

```
case 'i':
    printf("Inserimento nuovo appuntamento:\n");
    do {
        giornoSel = giornoSel - 1;
        do {
            oraSel = oraSel - 8;
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato == LIBERO) {
```

```
case 'v':
    if((giornoSel < 0)) {
        printf("Appuntamenti del giorno %d:\n", giornoSel + 1);
        for(ora = 0; ora < ORE; ora++) {
            giornoSel = -1;
            break;
```

```
case 'p':
    printf("Registra pagamento:\n");
    do {
        giornoSel = giornoSel - 1;
        do {
            oraSel = oraSel - 8;
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {
```

```
case 's':
    printf("Stampa ricevuta:\n");
    if((giornoSel < 0) || (oraSel < 0)) {
        scanf("%*c"); //fflush(stdin);
        if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {
            giornoSel = -1;
            oraSel = -1;
            break;
```

```
case 'q':
    printf("\nEsco dall'agenda, buon lavoro!\n");
    break;
```

```
default:
    printf("\nOperazione non consentita, inserire un'operazione valida.\n");
```

```
}
```

Inserire un nuovo appuntamento

Vedere tutti gli appuntamenti di un giorno

Inserire un pagamento

Agenda: Switch

```
switch(scelta) {
```

```
    case 'i':
        printf("Inserimento nuovo appuntamento:\n");
        do {
            giornoSel = giornoSel - 1;
        } do {
            oraSel = oraSel - 8;
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato == LIBERO) {
```

```
    case 'v':
        if((giornoSel < 0)) {
            printf("Appuntamenti del giorno %d:\n", giornoSel + 1);
            for(ora = 0; ora < ORE; ora++) {
                giornoSel = -1;
                break;
            }
        }
```

```
    case 'p':
        printf("Registra pagamento:\n");
        do {
            giornoSel = giornoSel - 1;
        } do {
            oraSel = oraSel - 8;
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {
```

```
    case 's':
        printf("Stampa ricevuta:\n");
        if((giornoSel < 0) || (oraSel < 0)) {
            scanf("%*c"); //fflush(stdin);
            if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {
                giornoSel = -1;
                oraSel = -1;
                break;
            }
        }
```

```
    case 'q':
        printf("\nEsco dall'agenda, buon lavoro!\n");
        break;
```

```
    default:
        printf("\nOperazione non consentita, inserire un'operazione valida.\n");
```

```
}
```

Inserire un nuovo appuntamento

Vedere tutti gli appuntamenti di un giorno

Inserire un pagamento

Stampare una “ricevuta” di pagamento

Agenda: Switch

```
switch(scelta) {
```

```
case 'i':
    printf("Inserimento nuovo appuntamento:\n");
    do {
        giornoSel = giornoSel - 1;
    } do {
        oraSel = oraSel - 8;
        scanf("%*c"); //fflush(stdin);
        if(agendaSett[giornoSel][oraSel].stato == LIBERO) {
```

```
case 'v':
    if((giornoSel < 0)) {
        printf("Appuntamenti del giorno %d:\n", giornoSel + 1);
        for(ora = 0; ora < ORE; ora++) {
            giornoSel = -1;
            break;
```

```
case 'p':
    printf("Registra pagamento:\n");
    do {
        giornoSel = giornoSel - 1;
    } do {
        oraSel = oraSel - 8;
        scanf("%*c"); //fflush(stdin);
        if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {
```

```
case 's':
    printf("Stampa ricevuta:\n");
    if((giornoSel < 0) || (oraSel < 0)) {
        scanf("%*c"); //fflush(stdin);
        if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {
            giornoSel = -1;
            oraSel = -1;
            break;
```

```
case 'q':
    printf("\nEsco dall'agenda, buon lavoro!\n");
    break;
```

```
default:
    printf("\nOperazione non consentita, inserire un'operazione valida.\n");
```

```
}
```

Inserire un nuovo appuntamento

Vedere tutti gli appuntamenti di un giorno

Inserire un pagamento

Stampare una “ricevuta” di pagamento

Uscita

Agenda: Inserire un nuovo appuntamento

```
case 'i':-
printf("Inserimento nuovo appuntamento:\n");-
do {-
    printf("Inserisci giorno (1 - %d): ", GIORNI);-
    scanf("%d", &giornoSel);-
}while((giornoSel < 1) || (giornoSel > GIORNI));-
giornoSel = giornoSel - 1;-
do{-
    printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
    scanf("%d", &oraSel);-
}while((oraSel < 8) || (oraSel >= ORE + 8 ));-
oraSel = oraSel - 8;-
scanf("%*c"); //fflush(stdin);-
if(agendaSett[giornoSel][oraSel].stato == LIBERO) {-
    printf("Inserisci nome paziente: ");-
    gets(agendaSett[giornoSel][oraSel].paziente);-
    printf("Inserisci tipo di prestazione: ");-
    gets(agendaSett[giornoSel][oraSel].tipoPrestazione);-
    printf("Inserisci eventuali note: ");-
    gets(agendaSett[giornoSel][oraSel].note);-
    agendaSett[giornoSel][oraSel].stato = PRENOTATO;-
} else {-
    printf("Impossibile completare l'operazione; per l'orario scelto e' gia' presen");-
    break;-
}-
```


Agenda: Inserire un nuovo appuntamento

```
case 'i':-
    printf("Inserimento nuovo appuntamento:\n");-
    do {-
        printf("Inserisci giorno (1 - %d): ", GIORNI);-
        scanf("%d", &giornoSel);-
    }while((giornoSel < 1) || (giornoSel > GIORNI));-
    giornoSel = giornoSel - 1;-
    do{-
        printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
        scanf("%d", &oraSel);-
    }while((oraSel < 8) || (oraSel >= ORE + 8 ));-
    oraSel = oraSel - 8;-
    scanf("%*c"); //fflush(stdin);-
    if(agendaSett[giornoSel][oraSel].stato == LIBERO) {-
        printf("Inserisci nome paziente: ");-
        gets(agendaSett[giornoSel][oraSel].paziente);-
        printf("Inserisci tipo di prestazione: ");-
        gets(agendaSett[giornoSel][oraSel].tipoPrestazione);-
        printf("Inserisci eventuali note: ");-
        gets(agendaSett[giornoSel][oraSel].note);-
        agendaSett[giornoSel][oraSel].stato = PRENOTATO;-
    } else {-
        printf("Impossibile completare l'operazione; per l'orario scelto e' gia' presen");-
        break;-
    }-
}
```

Agenda: Inserire un nuovo appuntamento

```
case 'i':-
printf("Inserimento nuovo appuntamento:\n");-
do {-
    printf("Inserisci giorno (1 - %d): ", GIORNI);-
    scanf("%d", &giornoSel);-
}while((giornoSel < 1) || (giornoSel > GIORNI));-
giornoSel = giornoSel - 1;-
do{-
    printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
    scanf("%d", &oraSel);-
}while((oraSel < 8) || (oraSel >= ORE + 8 ));-
oraSel = oraSel - 8;-
scanf("%*c"); //fflush(stdin);-
if(agendaSett[giornoSel][oraSel].stato == LIBERO) {-
    printf("Inserisci nome paziente: ");-
    gets(agendaSett[giornoSel][oraSel].paziente);-
    printf("Inserisci tipo di prestazione: ");-
    gets(agendaSett[giornoSel][oraSel].tipoPrestazione);-
    printf("Inserisci eventuali note: ");-
    gets(agendaSett[giornoSel][oraSel].note);-
    agendaSett[giornoSel][oraSel].stato = PRENOTATO;-
} else {-
    printf("Impossibile completare l'operazione; per l'orario scelto e' gia' presen");-
    break;-
}-
```

Agenda: Vedere tutti gli appuntamenti di un giorno

```
case 'v':-
    if((giornoSel < 0)) {-
        do {-
            printf("Inserisci giorno (1 - %d): ", GIORNI);-
            scanf("%d", &giornoSel);-
        }while((giornoSel < 1) || (giornoSel > GIORNI));-
        giornoSel = giornoSel - 1;-
        scanf("%*c"); //fflush(stdin);-
    }-
    printf("Appuntamenti del giorno %d:\n", giornoSel + 1);-
    for(ora = 0; ora < ORE; ora++) {-
        if(agendaSett[giornoSel][ora].stato == LIBERO) {-
            printf("- Alle %d non sono previsti appuntamenti\n", ora + 8);-
        } else if(agendaSett[giornoSel][ora].stato == PRENOTATO) {-
            printf("- Alle %d il signor %s ha prenotato una %s", ora + 8, agendaSett[giornoSel][ora].paziente, -
                agendaSett[giornoSel][ora].tipoPrestazione);-
            if(strlen(agendaSett[giornoSel][ora].note) > 0) {-
                printf(" (NOTE: %s)\n", agendaSett[giornoSel][ora].note);-
            } else {-
                printf("\n");-
            }-
        } else if(agendaSett[giornoSel][ora].stato == ESEGUITO) {-
            printf("- Alle %d il signor %s ha fatto una %s, e ha pagato %f\n", ora + 8, agendaSett[giornoSel][ora].paziente, -
                agendaSett[giornoSel][ora].tipoPrestazione, agendaSett[giornoSel][ora].pagato);-
        }-
    }-
    giornoSel = -1;-
    break;-
```


Agenda: Vedere tutti gli appuntamenti di un giorno

```
case 'v':-

if((giornoSel < 0)) {-
    do {-
        printf("Inserisci giorno (1 - %d): ", GIORNI);-
        scanf("%d", &giornoSel);-
    }while((giornoSel < 1) || (giornoSel > GIORNI));-
    giornoSel = giornoSel - 1;-
    scanf("%*c"); //fflush(stdin);-
}-
printf("Appuntamenti del giorno %d:\n", giornoSel + 1);-
for(ora = 0; ora < ORE; ora++) {-
    if(agendaSett[giornoSel][ora].stato == LIBERO) {-
        printf("- Alle %d non sono previsti appuntamenti\n", ora + 8);-
    } else if(agendaSett[giornoSel][ora].stato == PRENOTATO) {-
        printf("- Alle %d il signor %s ha prenotato una %s", ora + 8, agendaSett[giornoSel][ora].paziente, -
            agendaSett[giornoSel][ora].tipoPrestazione);-
        if(strlen(agendaSett[giornoSel][ora].note) > 0) {-
            printf(" (NOTE: %s)\n", agendaSett[giornoSel][ora].note);-
        } else {-
            printf("\n");-
        }-
    } else if(agendaSett[giornoSel][ora].stato == ESEGUITO) {-
        printf("- Alle %d il signor %s ha fatto una %s, e ha pagato %f\n", ora + 8, agendaSett[giornoSel][ora].paziente, -
            agendaSett[giornoSel][ora].tipoPrestazione, agendaSett[giornoSel][ora].pagato);-
    }-
}-
giornoSel = -1;-
break;-
```

Agenda: Inserire un pagamento

```
case 'p':-
printf("Registra pagamento:\n");-
do {-
    printf("Inserisci giorno (1 - %d): ", GIORNI);-
    scanf("%d", &giornoSel);-
}while((giornoSel < 1) || (giornoSel > GIORNI));-
giornoSel = giornoSel - 1;-
do{-
    printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
    scanf("%d", &oraSel);-
}while((oraSel < 8) || (oraSel >= ORE + 8 ));-
oraSel = oraSel - 8;-
scanf("%*c"); //fflush(stdin);-
if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {-
    printf("Per il giorno e l'ora selezionati non risulta nessun appuntamento, quindi non e' possibile effettuare\n");-
    break;-
} else {-
    printf("Alle %d il signor %s ha fatto una %s, e paga: ", oraSel + 8, agendaSett[giornoSel][oraSel].paziente, -
    > > > > > agendaSett[giornoSel][oraSel].tipoPrestazione);-
    scanf("%f", &agendaSett[giornoSel][oraSel].pagato);-
    agendaSett[giornoSel][oraSel].stato = ESEGUITO;-
}-
```


Agenda: Inserire un pagamento

```
case 'p':-
printf("Registra pagamento:\n");-
do {-
    printf("Inserisci giorno (1 - %d): ", GIORNI);-
    scanf("%d", &giornoSel);-
}while((giornoSel < 1) || (giornoSel > GIORNI));-
giornoSel = giornoSel - 1;-
do{-
    printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
    scanf("%d", &oraSel);-
}while((oraSel < 8) || (oraSel >= ORE + 8 ));-
oraSel = oraSel - 8;-
scanf("%*c"); //flush(stdin);-
if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {-
    printf("Per il giorno e l'ora selezionati non risulta nessun appuntamento, quindi non e' possibile effettuare\n");-
    break;-
} else {-
    printf("Alle %d il signor %s ha fatto una %s, e paga: ", oraSel + 8, agendaSett[giornoSel][oraSel].paziente, -
    > > > > > agendaSett[giornoSel][oraSel].tipoPrestazione);-
    scanf("%f", &agendaSett[giornoSel][oraSel].pagato);-
    agendaSett[giornoSel][oraSel].stato = ESEGUITO;-
}-
```

Agenda: Inserire un pagamento

```
case 'p':-
printf("Registra pagamento:\n");-
do {-
    printf("Inserisci giorno (1 - %d): ", GIORNI);-
    scanf("%d", &giornoSel);-
}while((giornoSel < 1) || (giornoSel > GIORNI));-
giornoSel = giornoSel - 1;-
do{-
    printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
    scanf("%d", &oraSel);-
}while((oraSel < 8) || (oraSel >= ORE + 8 ));-
oraSel = oraSel - 8;-
scanf("%*c"); //flush(stdin);-
if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {-
    printf("Per il giorno e l'ora selezionati non risulta nessun appuntamento, quindi non e' possibile effettuare\n");-
    break;-
} else {-
    printf("Alle %d il signor %s ha fatto una %s, e paga: ", oraSel + 8, agendaSett[giornoSel][oraSel].paziente, -
        agendaSett[giornoSel][oraSel].tipoPrestazione);-
    scanf("%f", &agendaSett[giornoSel][oraSel].pagato);-
    agendaSett[giornoSel][oraSel].stato = ESEGUITO;-
}-
```

Agenda: Inserire un pagamento

```
case 'p':-
printf("Registra pagamento:\n");-
do {-
    printf("Inserisci giorno (1 - %d): ", GIORNI);-
    scanf("%d", &giornoSel);-
}while((giornoSel < 1) || (giornoSel > GIORNI));-
giornoSel = giornoSel - 1;-
do{-
    printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
    scanf("%d", &oraSel);-
}while((oraSel < 8) || (oraSel >= ORE + 8 ));-
oraSel = oraSel - 8;-
scanf("%*c"); //flush(stdin);-
if(agendaSett[giornoSel][oraSel].stato != PRENOTATO) {-
    printf("Per il giorno e l'ora selezionati non risulta nessun appuntamento, quindi non e' possibile effettuare\n");-
    break;-
} else {-
    printf("Alle %d il signor %s ha fatto una %s, e paga: ", oraSel + 8, agendaSett[giornoSel][oraSel].paziente, -
        agendaSett[giornoSel][oraSel].tipoPrestazione);-
    scanf("%f", &agendaSett[giornoSel][oraSel].pagato);-
    agendaSett[giornoSel][oraSel].stato = ESEGUITO;-
}-
```

Agenda: Stampare una “ricevuta” di pagamento

```
case 's':-
    printf("Stampa ricevuta:\n");-
    if((giornoSel < 0) || (oraSel < 0)) {-
        do {-
            printf("Inserisci giorno (1 - %d): ", GIORNI);-
            scanf("%d", &giornoSel);-
        }while((giornoSel < 1) || (giornoSel > GIORNI));-
        giornoSel = giornoSel - 1;-
        do{-
            printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
            scanf("%d", &oraSel);-
        }while((oraSel < 8) || (oraSel >= ORE + 8 ));-
        oraSel = oraSel - 8;-
    }-
    scanf("%*c"); //fflush(stdin);-
    if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {-
        printf("Per il giorno e l'ora selezionati non risulta nessun pagamento.\n");-
    } else {-
        printf("Il giorno %d alle %d il signor %s ha fatto una %s, e ha pagato %.2f\n", giornoSel + 1, oraSel + 8,
            agendaSett[giornoSel][oraSel].paziente, agendaSett[giornoSel][oraSel].tipoPrestazione, agendaSett[gi
        });-
    }-
    giornoSel = -1;-
    oraSel = -1;-
    break;-
```


Agenda: Stampare una “ricevuta” di pagamento

```
case 's':-
    printf("Stampa ricevuta:\n");-
    if((giornoSel < 0) || (oraSel < 0)) {-
        do {-
            printf("Inserisci giorno (1 - %d): ", GIORNI);-
            scanf("%d", &giornoSel);-
        }while((giornoSel < 1) || (giornoSel > GIORNI));-
        giornoSel = giornoSel - 1;-
        do{-
            printf("Inserisci ora (8 - %d): ", ORE + 8 - 1);-
            scanf("%d", &oraSel);-
        }while((oraSel < 8) || (oraSel >= ORE + 8 ));-
        oraSel = oraSel - 8;-
    }-
    scanf("%*c"); //fflush(stdin);
    if(agendaSett[giornoSel][oraSel].stato != ESEGUITO) {-
        printf("Per il giorno e l'ora selezionati non risulta nessun pagamento.\n");-
    } else {-
        printf("Il giorno %d alle %d il signor %s ha fatto una %s, e ha pagato %.2f\n", giornoSel + 1, oraSel + 8,
            agendaSett[giornoSel][oraSel].paziente, agendaSett[giornoSel][oraSel].tipoPrestazione, agendaSett[gi
        });-
        giornoSel = -1;-
        oraSel = -1;-
        break;-
    }-
}
```

Agenda: Uscita

```
case 'q':-  
    printf("\nEsco dall'agenda, buon lavoro!\n");-  
    break;|  
default:-  
    printf("\nOperazione non consentita, inserire un'operazione valida.\n");-
```



**Tutte il materiale sarà
disponibile sul mio sito
internet!**

alessandronacci.it

See You Next Time!

