



POLITECNICO  
DI MILANO

# INFORMATICA

Una metodologia per la  
scrittura di programmi  
che utilizzano molte  
funzioni

```
cout << "numero di zone presenti (massimo " << MAXZONE << "):";
cin >> numZone;
for (zona = 0; zona < numZone; zona++)
{ cout << "nome della zona denotata nel seguito"
    << " con il numero progressivo " << zona << ": ";
  cin >> nomeZone[zona];
}

// ciclo di acquisizione di zona e prezzo per ogni rilevazione
for(zona = 0; zona < numZone; zona++)
{ num[zona] = 0; somma[zona] = 0;
}
cout << setw(53) << "acquisizione    rilevazioni" << endl;
while(1) // ciclo di acquisizione zona e prezzo per ogni rilevazione
{ cout << "zona della nuova rilevazione, eof se fine: ";
  cin >> zona;
  if (cin.eof())                // se segnale di fine
    break;                      // chiudi il ciclo
  if (zona >= 0 && zona < numZone) // se la zona non è corretta
  { if (num[zona] >= MAXDIM)      // se non c'è posto
    break;                      // chiudi il ciclo
    cout << "prezzo della nuova rilevazione: "; cin >> prezzo;
    somma[zona] += prezzo; prezzi[zona][num[zona]] = prezzo;
    num[zona]++;                // gestisci la rilevazione
  }
  else                          // altrimenti segnala zona
    cout << "zona non corretta: inserire nuovo valore"; //errata
}

// stampa di una tabella separata per ogni zona, con prezzo e
```

```
cout << "numero di zone presenti (massimo " << MAXZONE << "):";  
cin >> numZone;  
for (zona = 0; zona < numZone; zona++)
```

```
    cout << "nome della zona denotata nel seguito"  
          << " con il numero progressivo " << zona << ": ";  
    cin >> nomeZone[zona];  
    zona++;
```

## Problema

```
// ciclo di acquisizione di zona e prezzo per ogni rilevazione
```

```
for(zona = 0; zona < numZone; zona++)  
    num[zona] = 0; somma[zona] = 0;
```

```
cout << setw(53) << "acquisizione   rilevazioni" << endl;
```

```
while(1) // ciclo di acquisizione zona e presso per ogni rilevazione
```

```
    cout << "zona della nuova rilevazione, eof se fine: ";
```

```
    cin >> zona;
```

```
    if (cin.eof())
```

```
        break;
```

```
    if (zona >= 0 && zona < numZone)
```

```
        { if (num[zona] >= MAXDIM)
```

```
            break;
```

```
            cout << "prezzo della nuova rilevazione: "; cin >> prezzo;
```

```
            somma[zona] += prezzo; prezzi[zona][num[zona]] = prezzo;
```

```
            num[zona]++; }
```

```
            // gestisci la rilevazione
```

```
    else
```

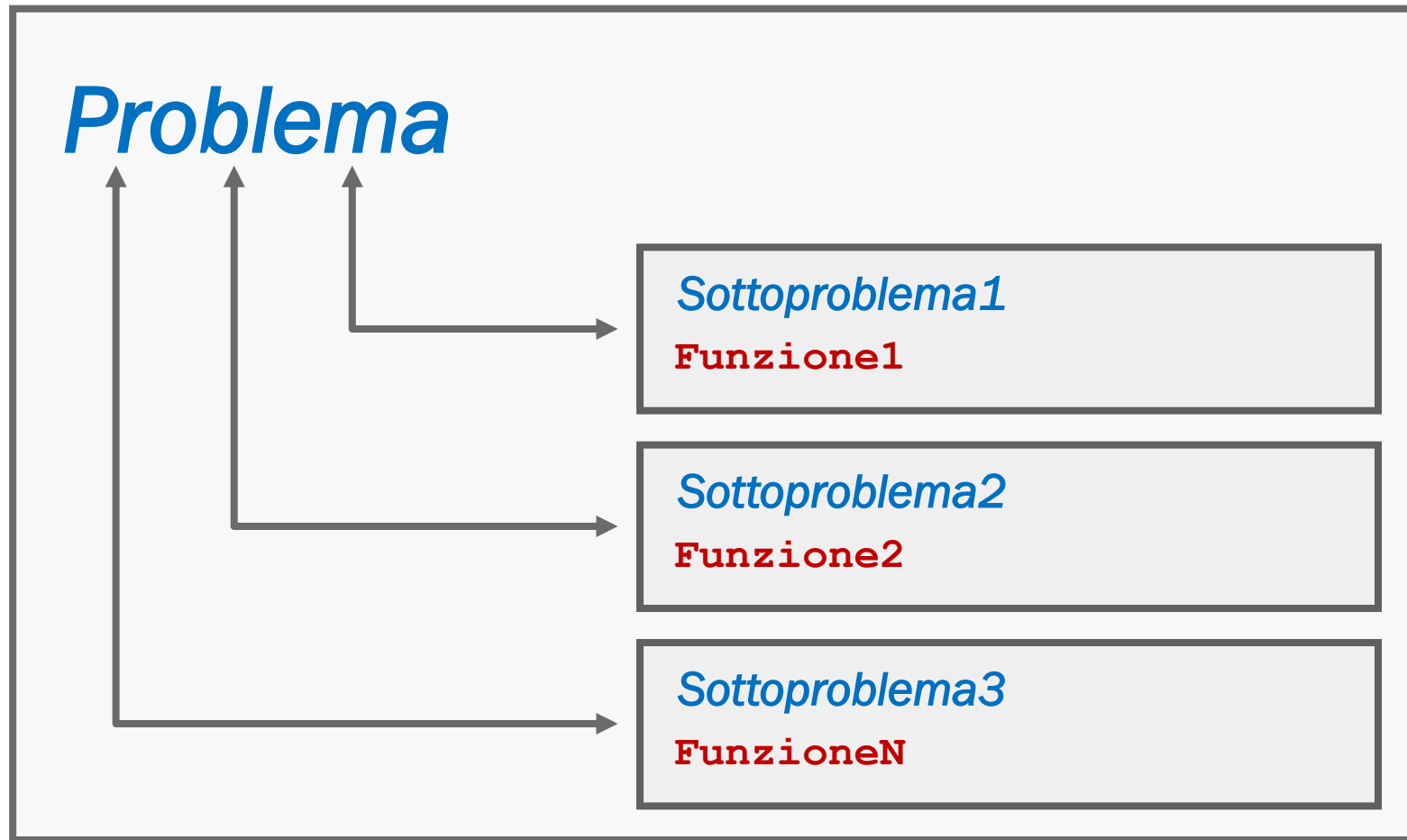
```
        // altrimenti segnala zona
```

```
        cout << "zona non corretta: inserire nuovo valore"; //errata
```

## Sottoproblema

**visualizzaDeviazioni(...)**

```
// stampa di una tabella separata per ogni zona, con prezzo e
```



## Funzioni

# Problema

*Analisi del prezzo di un prodotto in una città divisa in più zone.*

acquisizione di numero e nome delle zone

acquisizione di zona e prezzo per ogni rilevazione

stampa di una tabella separata per ogni zona,  
conversione e visualizzazione di ogni  
rilevazione rispetto alla media della zona

**visualizzaDeviazioni(...)**

# Problema

*Analisi del prezzo di un prodotto in una città divisa in più zone.*

acquisizione di numero e nome delle zone

acquisizione di zona e prezzo per ogni rilevazione

per ogni zona  
se la zona è caratterizzata da almeno una rilevazione

stampa l'intestazione della tabella

calcola lo scarto rispetto alla media di  
ogni rilevazione

stampa prezzo e scarto rispetto alla media  
di ogni rilevazione

Scelta di numero e nome di ogni zona

nome scelto

scegliZone

## Scelta di numero e nome di ogni zona

nome scelto

scegliZone

parametri in ingresso



parametri in uscita



Scelta di numero e nome di ogni zona

nome scelto

scegliZone

parametri in ingresso

*nessuno*

parametri in uscita

## Scelta di numero e nome di ogni zona

nome scelto

scegliZone

parametri in ingresso

*nessuno*

parametri in uscita

numZone **di tipo int**

## Scelta di numero e nome di ogni zona

nome scelto

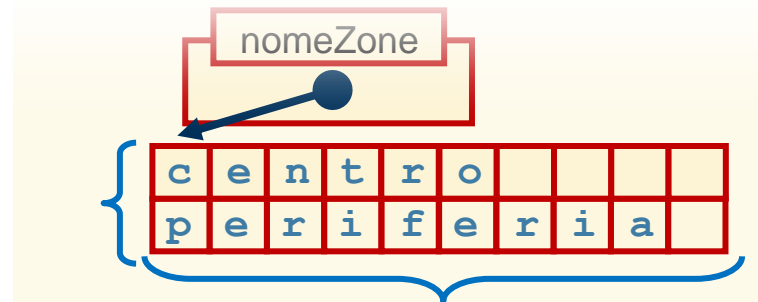
scegliZone

parametri in ingresso

*nessuno*

parametri in uscita

numZone di tipo int

**nomeZone**

## Scelta di numero e nome di ogni zona

nome scelto

scegliZone

parametri in ingresso

*nessuno*

parametri in uscita

numZone di tipo int

nomeZone di tipo array di array di char

## Scelta di numero e nome di ogni zona

nome scelto

scegliZone

tipo associato

**void**

parametri in ingresso

*nessuno*

parametri in uscita

numZone di tipo int

nomeZone di tipo array di array di char

## Scelta di numero e nome di ogni zona

nome scelto

scegliZone

tipo associato

void

parametri in ingresso

*nessuno*

parametri in uscita

numZone di tipo int

nomeZone di tipo array di array di char

## Intestazione

**void scegliZone**

**Scelta di numero e nome di ogni zona****nome scelto****scegliZone****tipo associato****void****parametri in ingresso*****nessuno*****parametri in uscita****numZone di tipo int****nomeZone di tipo array di array di char****Intestazione****`void scegliZone(int & numZone, char nomeZone[][MAXNOME])`**

## Scelta di numero e nome di ogni zona

nome scelto

scegliZone

tipo associato

void

parametri in ingresso

*nessuno*

parametri in uscita

numZone di tipo int

nomeZone di tipo array di array di char

## Intestazione

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME])
```



## Passato per valore

Scelta di numero e nome di ogni zona

nome scelto

scegliZone

tipo associato

void

parametri in ingresso

*nessuno*

parametri in uscita

numZone di tipo int

nomeZone di tipo array di array di char

Intestazione

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

## Scelta di numero e nome di ogni zona

nome scelto	scegliZone
tipo associato	void
parametri in ingresso	<i>nessuno</i>
parametri in uscita	numZone di tipo int nomeZone di tipo array di array di char

## Intestazione

```
void scegliZone(int & numZone,  const char nomeZone[][MAXNOME])
```

### Scelta di numero e nome di ogni zona

nome scelto

scegliZone

tipo associato

void

parametri in ingresso

*nessuno*

parametri in uscita

numZone di tipo int

nomeZone di tipo array di array di char

### Intestazione

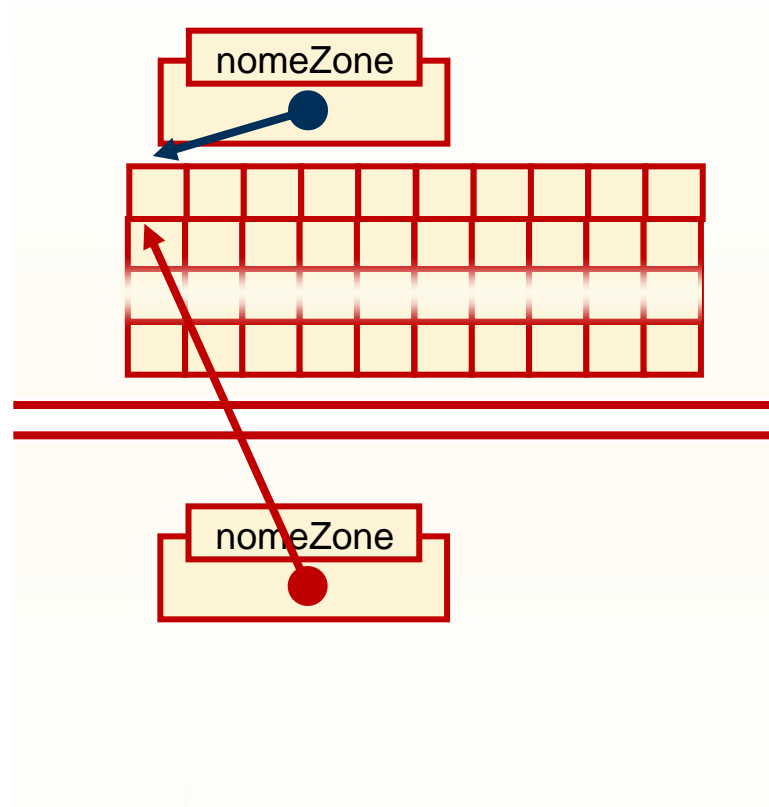
```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME])
```

### Scelta di numero e nome di ogni zona

nome scelto	scegliZone
tipo associato	void
parametri in ingresso	<i>nessuno</i>
parametri in uscita	numZone di tipo int nomeZone di tipo <b>array di array</b> di char

### Intestazione

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME] )
```



Intestazione

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

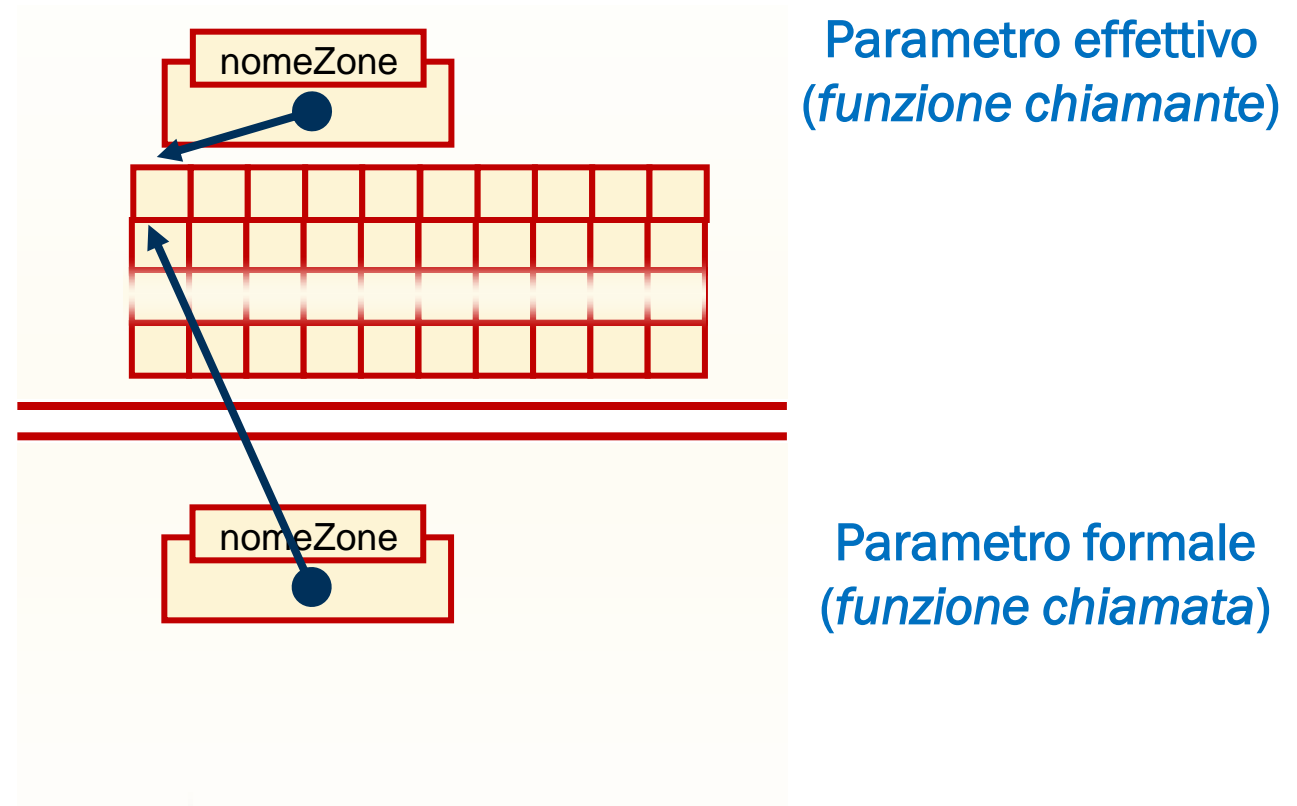
## Scelta di numero e nome di ogni zona

nome scelto	scegliZone
tipo associato	void
parametri in ingresso	<i>nessuno</i>
parametri in uscita	numZone di tipo int nomeZone di tipo array di array di char

## Intestazione

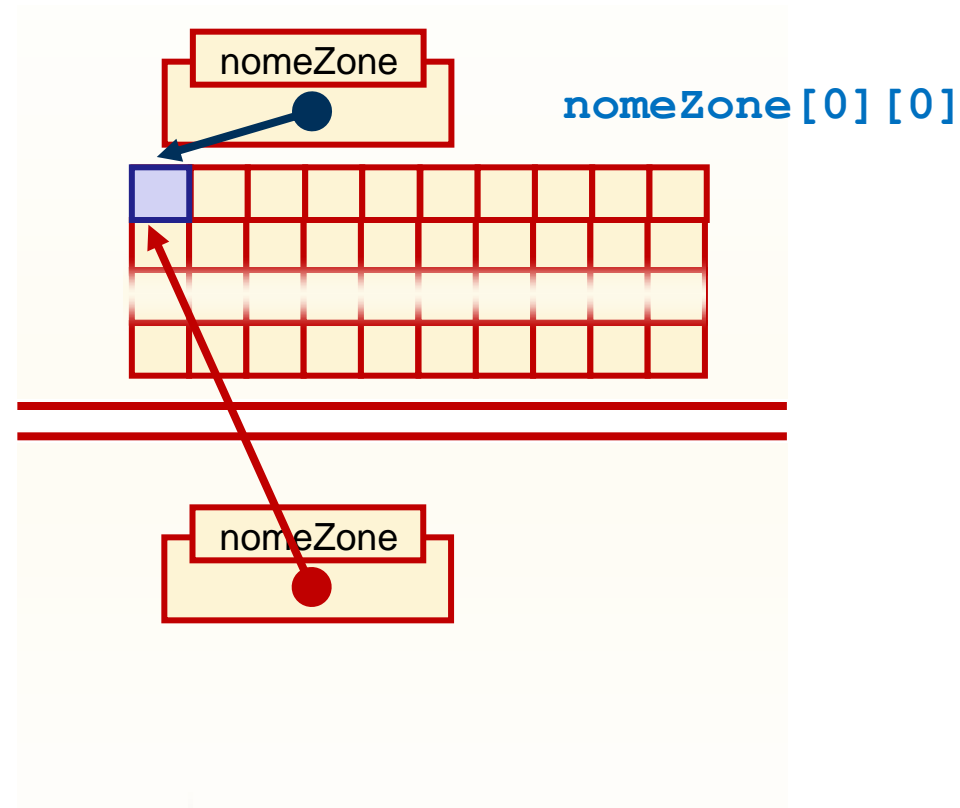
```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME])
```





Intestazione

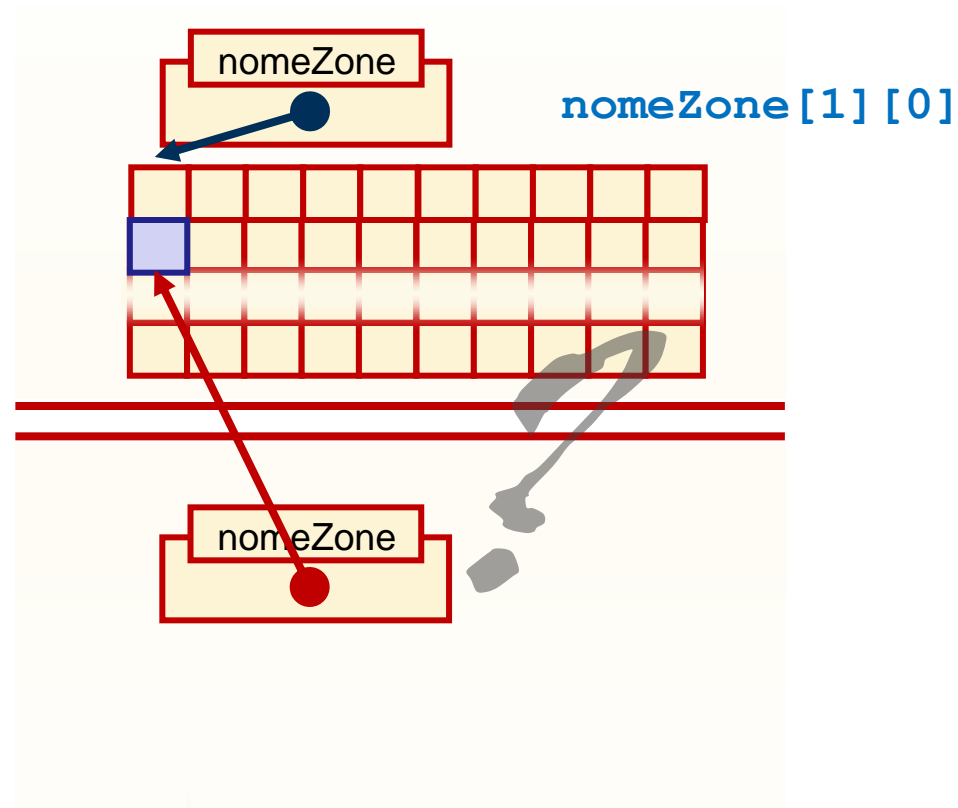
```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME])
```



Intestazione

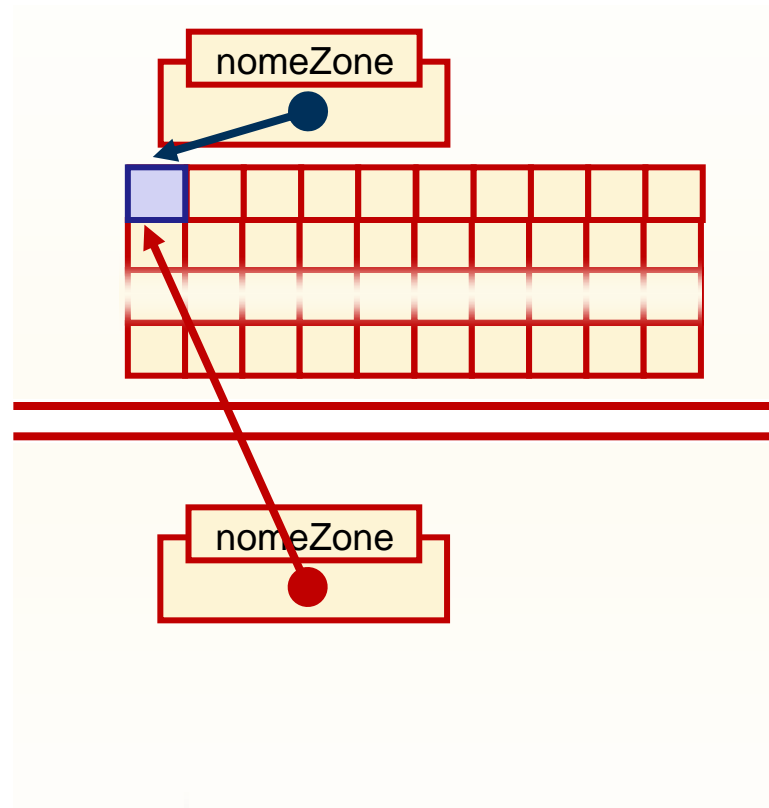
```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```





Intestazione

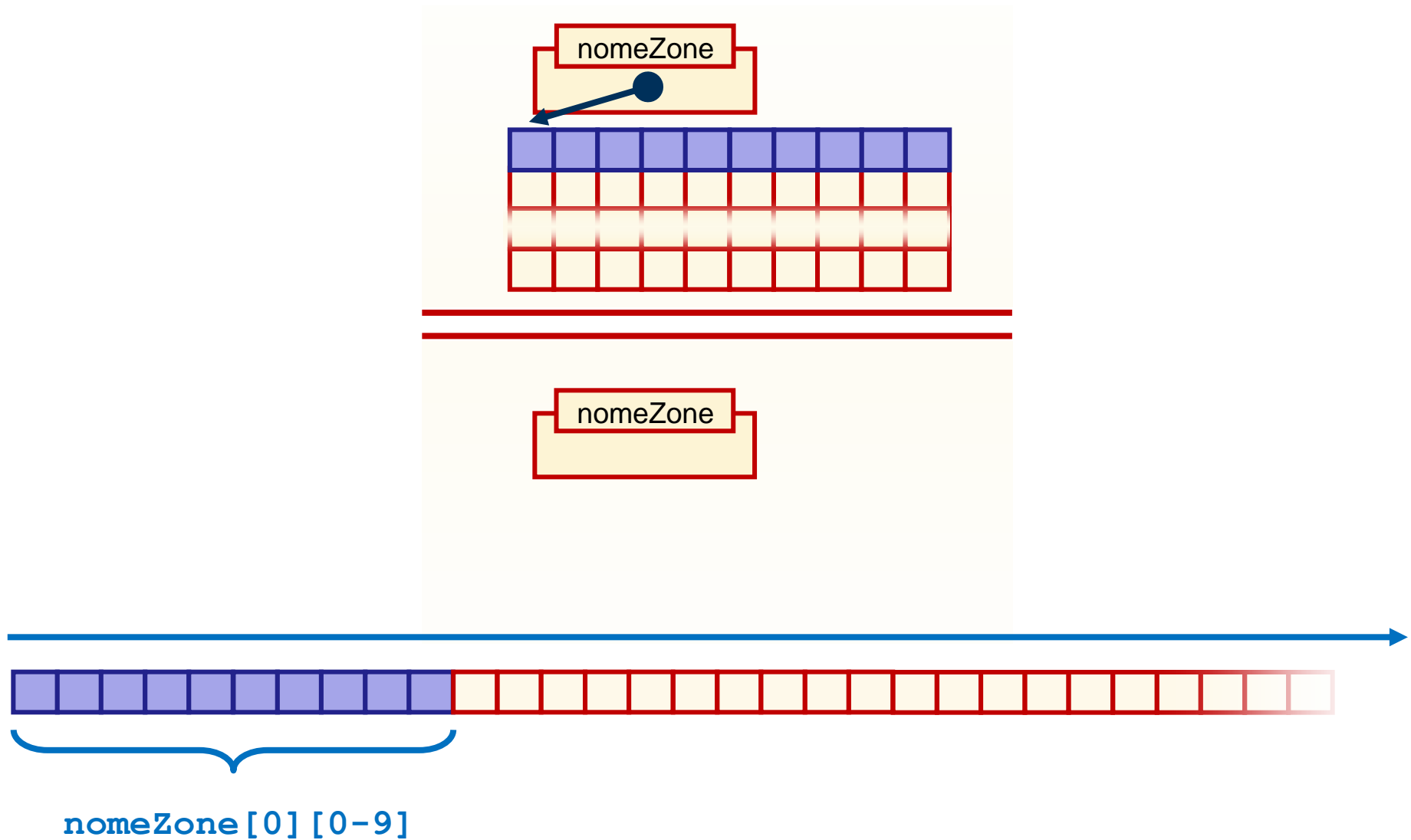
```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

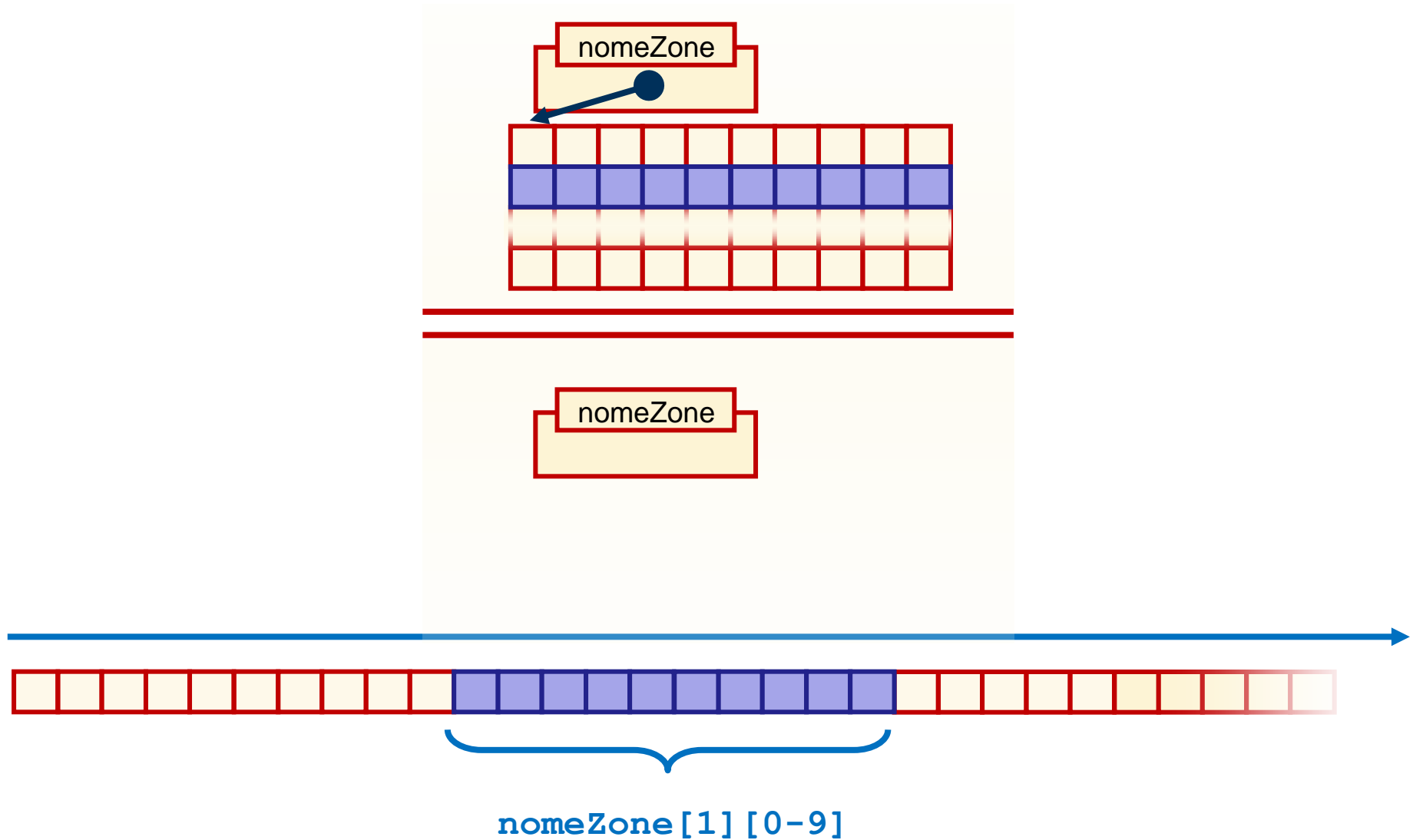


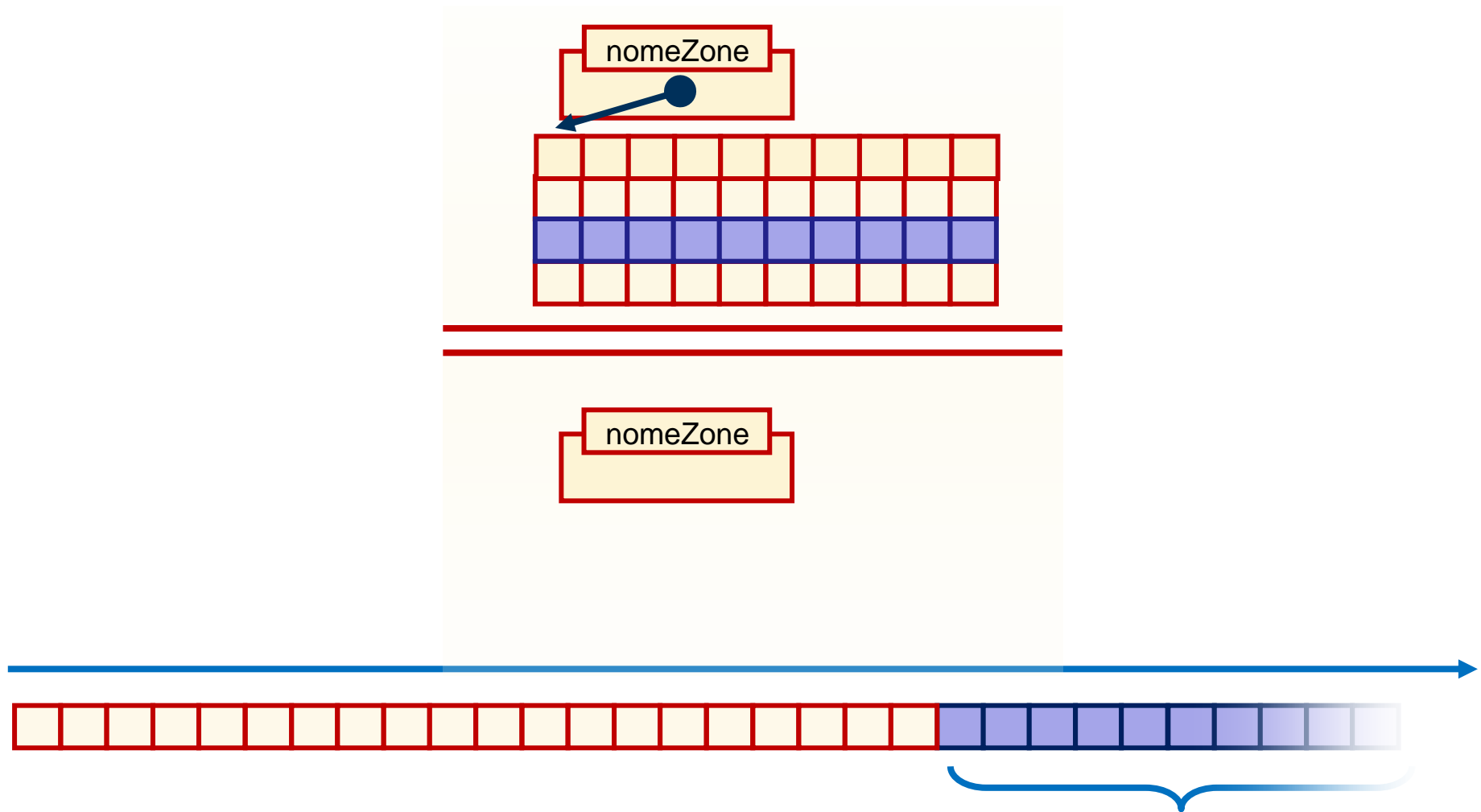
Intestazione

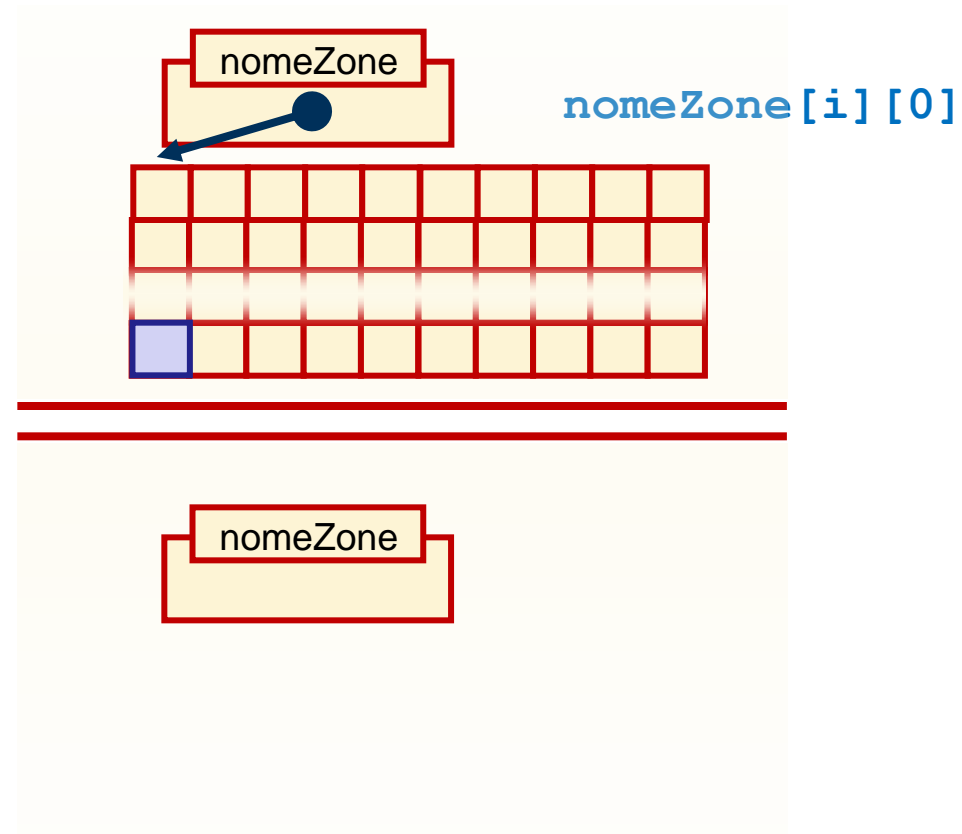
```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

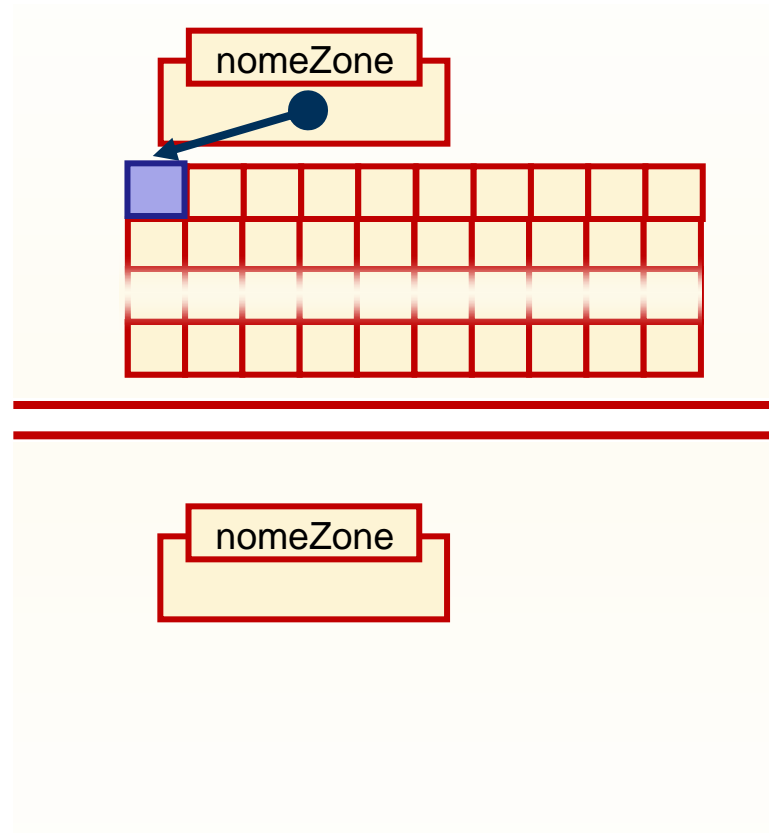






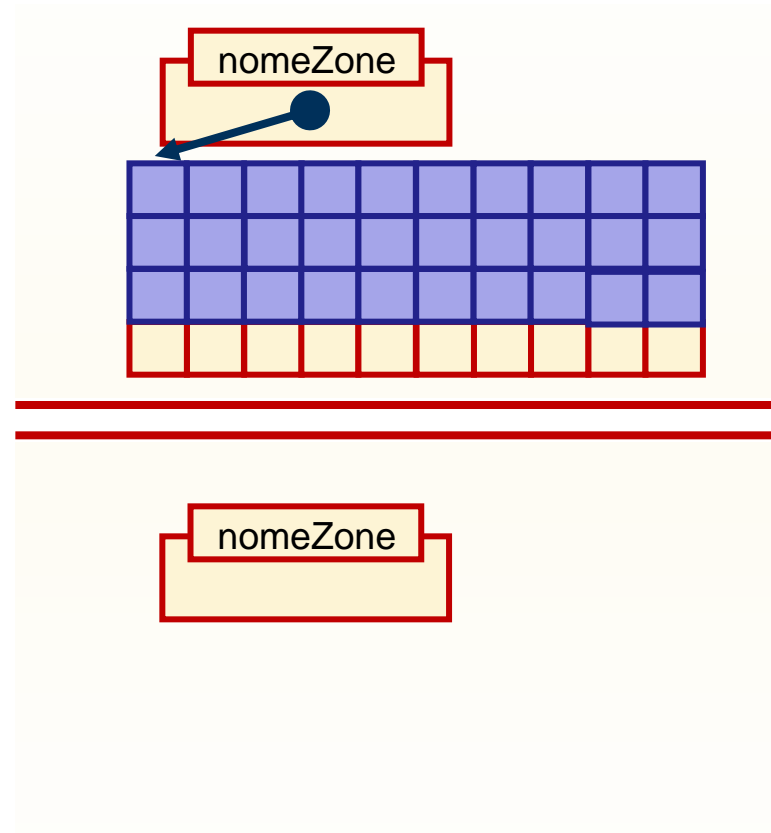




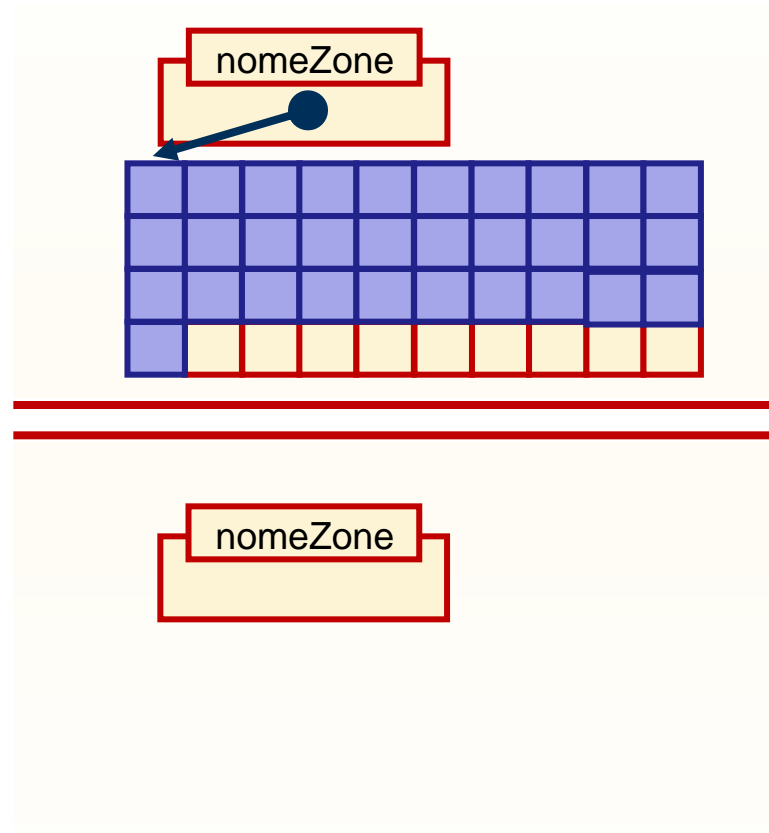


*indirizzo(nomeZone[0][0]) +*

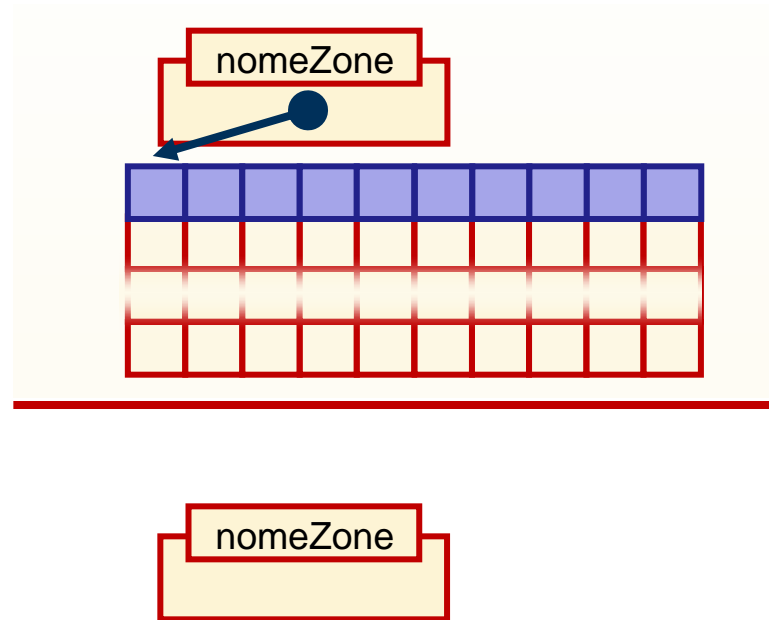




```
indirizzo(nomeZone[0][0]) +  
i * dimensione(riga) +
```



```
indirizzo(nomeZone[0][0]) +  
    i * dimensione(riga) +  
    0 * dimensione(cella)
```



```
indirizzo(nomeZone[0][0]) +  
    i * dimensione(riga) +  
    0 * dimensione(cella) +
```

### Scelta di numero e nome di ogni zona

nome scelto

scegliZone

tipo associato

void

parametri in ingresso

*nessuno*

parametri in uscita

numZone di tipo int

nomeZone di tipo array di array di char

### Intestazione

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME] )
```

Acquisizione di zona e prezzo per ogni rilevazione

nome scelto

acquisisciRilevazioni

**Acquisizione di zona e prezzo per ogni rilevazione****nome scelto****acquisisciRilevazioni****tipo associato****void**

## Intervallo permesso

Acquisizione di zona e prezzo per ogni rilevazione

nome scelto

acquisisciRilevazioni

tipo associato

void

parametri in ingresso

numZone

parametri in uscita

**Acquisizione di zona e prezzo per ogni rilevazione**

nome scelto

acquisisciRilevazioni

tipo associato

void

parametri in ingresso

numZone **di tipo int**

parametri in uscita



## Acquisizione di zona e prezzo per ogni rilevazione

nome scelto

acquisisciRilevazioni

tipo associato

void

parametri in ingresso

numZone di tipo int



parametri in uscita

num  
somma  
prezzi

**Acquisizione di zona e prezzo per ogni rilevazione**

nome scelto

acquisisciRilevazioni

tipo associato

void

parametri in ingresso

numZone di tipo int

parametri in uscita

num di tipo array

somma di tipo array

prezzi di tipo array

**Acquisizione di zona e prezzo per ogni rilevazione**

nome scelto

acquisisciRilevazioni

tipo associato

void

parametri in ingresso

numZone di tipo int

parametri in uscita

num di tipo array di int  
somma di tipo array di int  
prezzi di tipo array

**Acquisizione di zona e prezzo per ogni rilevazione**

nome scelto

acquisisciRilevazioni

tipo associato

void

parametri in ingresso

numZone di tipo int

parametri in uscita

num di tipo array di int

somma di tipo array di int

prezzi di tipo array di array di int

**Acquisizione di zona e prezzo per ogni rilevazione**

nome scelto

acquisisciRilevazioni

tipo associato

void

parametri in ingresso

numZone di tipo int

parametri in uscita

num di tipo array di int  
somma di tipo array di int  
prezzi di tipo array di array di int  
*(con dimensione di riga MAXDIM)*

Acquisizione di zona e prezzo per ogni rilevazione	
nome scelto	acquisisciRilevazioni
tipo associato	void
parametri in ingresso	numZone di tipo int
parametri in uscita	num di tipo array di int somma di tipo array di int prezzi di tipo array di array di int (con dimensione di riga MAXDIM)
Intestazione	
<pre>void acquisisciRilevazioni(int numZone, int num[],                            int somma[], int prezzi[][MAXDIM])</pre>	

Acquisizione di zona e prezzo per ogni rilevazione	
nome scelto	acquisisciRilevazioni
tipo associato	void
parametri in ingresso	numZone di tipo int
parametri in uscita	num di tipo array di int somma di tipo array di int prezzi di tipo array di array di int (con dimensione di riga MAXDIM)

### Intestazione

```

void acquisisciRilevazioni(int numZone, int num[],
                           int somma[], int prezzi[][MAXDIM])
    
```

# Problema

*Analisi del prezzo di un prodotto in una città divisa in più zone.*

acquisizione di numero e nome delle zone

acquisizione di zona e prezzo per ogni rilevazione

per ogni zona

se la zona è caratterizzata da almeno una rilevazione

stampa l'intestazione della tabella

calcola lo scarto rispetto alla media di  
ogni rilevazione

stampa prezzo e scarto rispetto alla media  
di ogni rilevazione



# Problema

*Analisi del prezzo di un prodotto in una città divisa in più zone.*

acquisizione di numero e nome delle zone

acquisizione di zona e prezzo per ogni rilevazione

per ogni zona

se la zona è caratterizzata da almeno una rilevazione

stampa l'intestazione della tabella

calcola lo scarto rispetto alla media di  
ogni rilevazione

stampa prezzo e scarto rispetto alla media  
di ogni rilevazione

# Problema

*Analisi del prezzo di un prodotto in una città divisa in più zone.*

acquisizione di numero e nome delle zone

acquisizione di zona e prezzo per ogni rilevazione

per ogni zona  
se la zona è caratterizzata da almeno una rilevazione

stampa l'intestazione della tabella

calcola lo scarto rispetto alla media di  
ogni rilevazione

stampa prezzo e scarto rispetto alla media  
di ogni rilevazione

Stampa l'intestazione della tabella

nome scelto

**stampaIntestazione**

Stampa l'intestazione della tabella

nome scelto

stampaIntestazione

tipo associato

void

### Stampa l'intestazione della tabella

nome scelto

stampaIntestazione

tipo associato

void

parametri in ingresso

nomeZonadi tipo array di char

### Stampa l'intestazione della tabella

nome scelto

stampaIntestazione

tipo associato

void

parametri in ingresso

nomeZonadi tipo array di char

parametri in uscita

*nessuno*

## Uscita su COUT

Stampa l'intestazione della tabella

nome scelto	stampaIntestazione
tipo associato	void
parametri in ingresso	nomeZonadi tipo array di char
parametri in uscita	<i>nessuno</i>

### Stampa l'intestazione della tabella

nome scelto	stampaIntestazione
tipo associato	void
parametri in ingresso	nomeZona di tipo array di char
parametri in uscita	<i>nessuno</i>

### Intestazione

```
void stampaIntestazione(const char nomeZona[])
```



## Per valore

Stampa l'intestazione della tabella

nome scelto

stampaIntestazione

tipo associato

void

parametri in ingresso

nomeZona di tipo array di char

parametri in uscita

*nessuno*

Intestazione

```
void stampaIntestazione(const char nomeZona[])
```

## Per valore

Stampa l'intestazione della tabella

nome scelto

stampaIntestazione

tipo associato

void

parametri in ingresso

nomeZona di tipo array di char

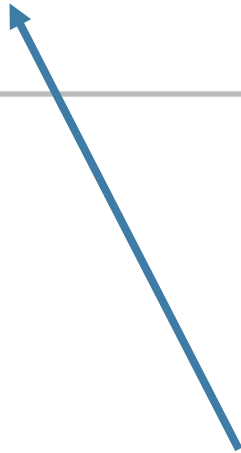
parametri in uscita

*nessuno*

Intestazione

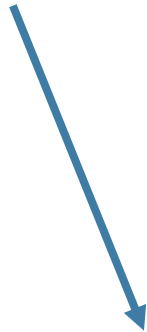
```
void stampaIntestazione(const char nomeZona[])
```

Calcola lo scarto rispetto alla media di ogni rilevazione della zona	
parametri in ingresso	



Informazioni per ogni singola zona

Calcola lo scarto rispetto alla media di ogni rilevazione della zona	
parametri in uscita	



**Risultati passati ad un'altra funzione**

Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto

calcolaDeviazioni

tipo associato

void

parametri in ingresso

somma di tipo int

num di tipo int

prezzi di tipo array di int

Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto

calcolaDeviazioni

tipo associato

void

parametri in ingresso

somma di tipo int

num di tipo int

prezzi di tipo array di int

parametri in uscita

di tipo array di float

Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto

calcolaDeviazioni

tipo associato

void

parametri in ingresso

somma di tipo int

num di tipo int

prezzi di tipo array di int

parametri in uscita

deviazioni di tipo array di float

Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto

calcolaDeviazioni

tipo associato

void

parametri in ingresso

somma di tipo int

num di tipo int

prezzi di tipo array di int

parametri in uscita

deviazioni di tipo array di float

Intestazione

```
void calcolaDeviazioni(int somma, int num,  
                        const int prezzi[], float deviazioni[])
```



Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto

calcolaDeviazioni

tipo associato

void

parametri in ingresso

somma di tipo int

num di tipo int

prezzi di tipo array di int

parametri in uscita

deviazioni di tipo array di float

Intestazione

```
void calcolaDeviazioni(int somma, int num,  
                        const int prezzi[], float deviazioni[])
```

Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto	calcolaDeviazioni
tipo associato	void
parametri in ingresso	somma di tipo int num di tipo int prezzi di tipo array di int
parametri in uscita	deviazioni di tipo array di float

Intestazione

```
void calcolaDeviazioni(int somma, int num,  
                        const int prezzi[], float deviazioni[])
```

Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto

calcolaDeviazioni

tipo associato

void

parametri in ingresso

somma di tipo int

num di tipo int

prezzi di tipo array di int

parametri in uscita

deviazioni di tipo array di float

Intestazione

```
void calcolaDeviazioni(int somma, int num,  
                        const int prezzi[], float deviazioni[])
```

Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto	calcolaDeviazioni
tipo associato	void
parametri in ingresso	somma di tipo int num di tipo int prezzi di tipo array di int
parametri in uscita	deviazioni di tipo array di float

Intestazione

```
void calcolaDeviazioni(int somma, int num,  
                        const int prezzi[], float deviazioni[])
```

Calcola lo scarto rispetto alla media  
di ogni rilevazione della zona

nome scelto

calcolaDeviazioni

tipo associato

void

parametri in ingresso

somma di tipo int

num di tipo int

prezzi di tipo array di int

parametri in uscita

deviazioni di tipo array di float

Intestazione

```
void calcolaDeviazioni(int somma, int num,  
                        const int prezzi[], float deviazioni[])
```

Stampa prezzo e scarto rispetto alla media di ogni rilevazione della zona

nome scelto

stampaintestazione

tipo associato

void

parametri in ingresso

prezzi di tipo array di int

Stampa prezzo e scarto rispetto alla media di ogni rilevazione della zona

nome scelto

stampaIntestazione

tipo associato

void

parametri in ingresso

prezzi di tipo array di int  
deviazioni di tipo array di float

Stampa prezzo e scarto rispetto alla media di ogni rilevazione della zona

nome scelto

stampaIntestazione

tipo associato

void

parametri in ingresso

prezzi di tipo array di int  
deviazioni di tipo array di float  
num di tipo int



Stampa prezzo e scarto rispetto alla media di ogni rilevazione della zona

nome scelto

stampaintestazione

tipo associato

void

parametri in ingresso

prezzi di tipo array di int  
deviazioni di tipo array di float  
num di tipo int

parametri in uscita

**nessuno**

Stampa prezzo e scarto rispetto alla media di ogni rilevazione della zona

nome scelto

stampaIntestazione

tipo associato

void

parametri in ingresso

prezzi di tipo array di int  
deviazioni di tipo array di float  
num di tipo int

parametri in uscita

*nessuno*

Intestazione

```
void stampaDeviazioni(int num, const int prezzi[],  
                      const float deviazioni[])
```

Stampa prezzo e scarto rispetto alla media di ogni rilevazione della zona

nome scelto

stampaintestazione

tipo associato

void

parametri in ingresso

prezzi di tipo array di int  
deviazioni di tipo array di float  
num di tipo int

parametri in uscita

*nessuno*

Intestazione

```
void stampaDeviazioni(int num, const int prezzi[],  
                        const float deviazioni[])
```

```
void scegliZone  
    (int & numZone, char nomeZone[][MAXNOME])
```

```
void acquisisciRilevazioni(int numZone,  
    int num[], int somma[], int prezzi[][MAXDIM])
```

```
void stampaIntestazione(const char nomeZona[])
```

```
void calcolaDeviazioni(int somma, int num,  
    const int prezzi[], float deviazioni[])
```

```
void stampaDeviazioni(int num,  
    const int prezzi[], const float deviazioni[])
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])  
{ ... }
```

```
void acquisisciRilevazioni(int numZone, int num[],  
                           int somma[], int prezzi[][MAXDIM])
```

```
void stampaIntestazione(const char nomeZona[])
```

```
void calcolaDeviazioni(int somma, int num,  
                      const int prezzi[], float deviazioni[])
```

```
void stampaDeviazioni(int num, const int prezzi[],  
                     const float deviazioni[])
```

```
main()  
{ ... }
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])  
{ ... }
```

```
void acquisisciRilevazioni(int numZone, int num[],  
                           int somma[], int prezzi[][MAXDIM])
```

```
void stampaIntestazione(const char nomeZona[])
```

```
void calcolaDeviazioni(int somma, int num,  
                       const int prezzi[], float deviazioni[])
```

```
void stampaDeviazioni(int num, const int prezzi[],  
                      const float deviazioni[])
```

```
main()  
{ ... }
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])  
{ ... }
```

```
void acquisisciRilevazioni(int numZone, int num[],  
                           int somma[], int prezzi[][MAXDIM])  
{ ... }
```

```
void stampaIntestazione(const char nomeZona[])  
{ ... }
```

```
void calcolaDeviazioni(int somma, int num,  
                      const int prezzi[], float deviazioni[])  
{ ... }
```

```
void stampaDeviazioni(int num, const int prezzi[],  
                     const float deviazioni[])  
{ ... }
```

```
main()  
{ ... }
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ ... }

void acquisisciRilevazioni(int numZone, int num[],
                           int somma[], int prezzi[][MAXDIM])
{ ... }

void stampaIntestazione(const char nomeZona[])
{ ... }

void calcolaDeviazioni(int somma, int num,
                       const int prezzi[], float deviazioni[])
{ ... }

void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[])
{ ... }
```

```
main()
{ ... }
```



```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ ... }

void acquisisciRilevazioni(int numZone, int num[],
                           int somma[], int prezzi[][MAXDIM])
{ ... }

void stampaIntestazione(const char nomeZona[])
{ ... }

void calcolaDeviazioni(int somma, int num,
                      const int prezzi[], float deviazioni[])
{ ... }

void stampaDeviazioni(int num, const int prezzi[],
                     const float deviazioni[])
{ ... }
```

1

```
main()
{ ... }
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ ... }

void acquisisciRilevazioni(int numZone, int num[],
                           int somma[], int prezzi[][MAXDIM])
{ ... }

void stampaIntestazione(const char nomeZona[])
{ ... }

void calcolaDeviazioni(int somma, int num,
                       const int prezzi[], float deviazioni[])
{ ... }

void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[])
{ ... }
```

```
main()
{ ... }
```



Divisione del lavoro

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ ... }

void acquisisciRilevazioni(int numZone, int num[],
                           int somma[], int prezzi[][MAXDIM])
{ ... }

void stampaIntestazione(const char nomeZona[])
{ ... }

void calcolaDeviazioni(int somma, int num,
                       const int prezzi[], float deviazioni[])
{ ... }

void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[])
{ ... }
```

1

```
main()
{ ... }
```

2

Struttura finale

```
#include <iostream>
using namespace std;
#include <iomanip.h>

const int MAXDIM = 100,
        MAXNOME = 10;
        MAXZONE = 15;

void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "):";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
    { cout << "nome della zona denotata nel seguito "
```

```
#include <iostream>
using namespace std;
#include <iomanip.h>
```

```
const int MAXDIM = 100,
        MAXNOME = 10;
        MAXZONE = 15;
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "):";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
    { cout << "nome della zona denotata nel seguito "
```

```
#include <iostream>
using namespace std;
#include <iomanip.h>
```

```
const int MAXDIM = 100,
        MAXNOME = 10;
        MAXZONE = 15;
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "):";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
    { cout << "nome della zona denotata nel seguito "
```

```
#include <iostream>
using namespace std;
#include <iomanip.h>
```

La dichiarazione precede l'uso

```
const int MAXDIM = 100,
        MAXNOME = 10;
        MAXZONE = 15;
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "):";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
    { cout << "nome della zona denotata nel seguito "
```

```
void acquisisciRilevazioni
    (int numZone, int num[], int somma[],
     int prezzi[][MAXDIM])
{ int zona;
  for(zona = 0; zona < numZone; zona++)
    { num[zona] = 0; somma[zona] = 0;
```

```
#include <iostream>
using namespace std;
#include <iomanip.h>

const int MAXDIM = 100,
        MAXNOME = 10;
        MAXZONE = 15;

void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "):";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
    { cout << "nome della zona denotata nel seguito "
```



```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "): ";
  cin >> numZone;
  for(zona = 0; zona < numZone; zona++)
  { cout << "nome della zona denotata nel seguito "
    << "con il numero progressivo " << zona << ": ";
    cin >> nomeZone[zona];
  }
}
```

```
void acquisisciRilevazioni(int numZone, int num[],
                           int somma[], int prezzi[][MAXDIM])
{ int zona;
  for(zona = 0; zona < numZone; zona++)
  { num[zona] = 0; somma[zona] = 0;
  }
  cout << setw(53) << "acquisizione   rilevazioni" << endl;
  while(1) // acquisizione zona e prezzo per ogni rilevazione
  { cout << "zona rilevazione?(eof=fine): ";
    cin >> zona;
    if (cin.eof() || num[zona] >= MAXDIM)
      break;
    if (zona >= 0 && zona < numZone)
    { cout << "prezzo della nuova rilevazione: ";
      cin >> prezzi[zona][num[zona]];
      somma[zona] += prezzi[zona][num[zona]]; num[zona]++;
    }
    else
      cout << "zona non corretta: inserire nuovo valore";
  }
}
```

```
void stampaIntestazione(const char nomeZona[])
{ cout << endl << setw(20) << "rilevazioni   zona "
  << setw(10) << nomeZona << endl << endl
  << setw(20) << "prezzi rilevati"
  << setw(40) << "deviazione rispetto al prezzo medio"
  << endl;
}
```

```
void calcolaDeviazioni(int somma, int num, const int prezzi[],  
                        float deviazioni[])  
{ float media;  
  int i;  
  media = (float)somma / num;  
  for (i = 0; i < num; i++)  
    deviazione[i] = prezzi[i] - media;  
}
```

```
void calcolaDeviazioni(int somma, int num, const int prezzi[],  
                      float deviazioni[])  
{ float media;  
  int i;  
  media = (float)somma / num;  
  for (i = 0; i < num; i++)  
    deviazione[i] = prezzi[i] - media;  
}
```

```
void calcolaDeviazioni(int somma, int num, const int prezzi[],  
                       float deviazioni[])  
{ float media;  
  int i;  
  media = (float)somma / num;  
  for (i = 0; i < num; i++)  
    deviazione[i] = prezzi[i] - media;  
}
```

```
void calcolaDeviazioni(int somma, int num, const int prezzi[],
                      float deviazioni[])
{ float media;
  int i;
  media = (float)somma / num;
  for (i = 0; i < num; i++)
    deviazione[i] = prezzi[i] - media;
}
```

```
void calcolaDeviazioni(int somma, int num, const int prezzi[],  
                      float deviazioni[])  
{ float media;  
  int i;  
  media = (float)somma / num;  
  for (i = 0; i < num; i++)  
    deviazione[i] = prezzi[i] - media;  
}
```



```
void calcolaDeviazioni(int somma, int num, const int prezzi[],  
                      float deviazioni[])  
{ float media;  
  int i;  
  media = (float)somma / num;  
  for (i = 0; i < num; i++)  
    deviazione[i] = prezzi[i] - media;  
}
```

```
void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[])
{ int i;
  for (i = 0; i < num; i++)
  { cout << setw(20) << prezzi[i]
      << setw(40) << setprecision(2)
      << setiosflags (ios::fixed | ios::showpoint)
      << deviazioni[i];
      << endl;
  }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```



```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

## Esercizio

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```



```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

## Variante

```
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;

  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
    { stampaIntestazione(nomeZone[zona]);
      calcolaDeviazioni(somma[zona], num[zona],
                       prezzi[zona], deviazioni[zona]);
      stampaDeviazioni(num[zona], prezzi[zona],
                       deviazioni[zona]);
    }
}
```

```
        else                                     // altrimenti segnala un errore
    }
}

void stampaIntestazione(const char nomeZona[])
{ cout << endl << setw(20) << "rilevazioni   zona " //crea
  << setw(10) << nomeZona << endl << endl // intestazione tabella
  << setw(20) << "prezzi rilevati" // per zona di nome nomeZona
  << setw(40) << "deviazione rispetto al prezzo medio" << endl;
}

void calcolaDeviazioni
    (int somma, int num, const int prezzi[],float deviazioni[])
{ float media;
  int i;
  media = (float)somma / num; // calcola il prezzo medio
  for (i = 0; i < num; i++) // per ogni rilevazione
      deviazione[i] = prezzi[i] - media; // calcola lo scarto
}

void stampaDeviazioni
    (int num, const int prezzi[], const float deviazioni[])
{ int i;
  for (i = 0; i < num; i++) // per ogni rilevazione
  { cout << setw(20) << prezzi[i] // stampa una riga contenente il
    << setw(40) << setprecision(2) // prezzo rilevato e la
    << setiosflags (ios::fixed | ios::showpoint) // deviazione
    << deviazioni[i] // rispetto alla media, in formato fisso,
    << endl; // sempre con il punto decimale e due cifre decimali
  }
}

int main()
{ int numZone; // numero delle zone
  char nomeZone[MAXZONE][MAXNOME]; // nomi delle zone
  int prezzi[MAXZONE][MAXDIM], // prezzi rilevati per zona
  float deviazioni[MAXZONE][MAXDIM];
```

```
        else // altrimenti segnala un errore
            cout << "zona non corretta: inserire nuovo valore";
    }
}

void stampaIntestazione(const char nomeZona[])
{ cout << endl << setw(20) << "rilevazioni   zona " //crea
  << setw(10) << nomeZona << endl << endl // intestazione tabella
  << setw(20) << "prezzi rilevati" // per zona di nome nomeZona
  << setw(40) << "deviazione rispetto al prezzo medio" << endl;
}

void calcolaDeviazioni
    (int somma, int num, const int prezzi[], float deviazioni[])
{ float media;
  int i;
  media = (float)somma / num; // calcola il prezzo medio
  for (i = 0; i < num; i++) // per ogni rilevazione
      deviazione[i] = prezzi[i] - media; // calcola lo scarto
}

void stampaDeviazioni
    (int num, const int prezzi[], const float deviazioni[])
{ int i;
  for (i = 0; i < num; i++) // per ogni rilevazione
  { cout << setw(20) << prezzi[i] // stampa una riga contenente il
    << setw(40) << setprecision(2) // prezzo rilevato e la
    << setiosflags (ios::fixed | ios::showpoint) // deviazione
    << deviazioni[i] // rispetto alla media, in formato fisso,
    << endl; // sempre con il punto decimale e due cifre decimali
  }
}

1 int main()
{ int numZone; // numero delle zone
  char nomeZone[MAXZONE][MAXNOME]; // nomi delle zone
  int prezzi[MAXZONE][MAXDIM], // prezzi rilevati per zona

  float deviazioni[MAXZONE][MAXDIM];
```

```
        else // altrimenti segnala un errore
            cout << "zona non corretta: inserire nuovo valore";
    }
}

void stampaIntestazione(const char nomeZona[])
{ cout << endl << setw(20) << "rilevazioni   zona " //crea
  << setw(10) << nomeZona << endl << endl // intestazione tabella
  << setw(20) << "prezzi rilevati" // per zona di nome nomeZona
  << setw(40) << "deviazione rispetto al prezzo medio" << endl;
}

void calcolaDeviazioni
    (int somma, int num, const int prezzi[], float deviazioni[])
{ float media;
  int i;
  media = (float)somma / num; // calcola il prezzo medio
  for (i = 0; i < num; i++) // per ogni rilevazione
      deviazione[i] = prezzi[i] - media; // calcola lo scarto
}

void stampaDeviazioni
    (int num, const int prezzi[], const float deviazioni[])
{ int i;
  for (i = 0; i < num; i++) // per ogni rilevazione
  { cout << setw(20) << prezzi[i] // stampa una riga contenente il
    << setw(40) << setprecision(2) // prezzo rilevato e la
    << setiosflags (ios::fixed | ios::showpoint) // deviazione
    << deviazioni[i] // rispetto alla media, in formato fisso,
    << endl; // sempre con il punto decimale e due cifre decimali
  }
}

int main()
{ int numZone; // numero delle zone
  char nomeZone[MAXZONE][MAXNOME]; // nomi delle zone
  int prezzi[MAXZONE][MAXDIM], // prezzi rilevati per zona

  float deviazioni[MAXZONE][MAXDIM];
```



```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "): ";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
    { cout << "nome della zona denotata nel seguito "
      << "con il numero progressivo " << zona << ": ";
      cin >> nomeZone[zona];
    }
}

...

int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;
  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if (num[zona] != 0)
```

## Intestazione

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "): ";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
  { cout << "nome della zona denotata nel seguito "
    << "con il numero progressivo " << zona << ": ";
    cin >> nomeZone[zona];
  }
}

...

int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;
  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if(num[zona] != 0)
```

## Corpo della funzione

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

```
{ int zona;  
  cout << "numero di zone presenti (massimo " << MAXZONE << "):";  
  cin >> numZone;  
  for (zona = 0; zona < numZone; zona++)  
  { cout << "nome della zona denotata nel seguito "  
    << "con il numero progressivo " << zona << ": ";  
    cin >> nomeZone[zona];  
  }  
}
```

...

```
int main()  
{ int numZone;  
  char nomeZone[MAXZONE][MAXNOME];  
  int prezzi[MAXZONE][MAXDIM],  
      num[MAXZONE], somma[MAXZONE];  
  float deviazioni[MAXZONE][MAXDIM];  
  int zona;  
  scegliZone(numZone, nomeZone);  
  acquisisciRilevazioni(numZone, num, somma, prezzi);  
  for (zona = 0; zona < numZone; zona++)  
    if(num[zona] != 0)
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "): ";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
  { cout << "nome della zona denotata nel seguito "
    << "con il numero progressivo " << zona << ": ";
    cin >> nomeZone[zona];
  }
}


...

int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;
  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if(num[zona] != 0)
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "): ";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
    { cout << "nome della zona denotata nel seguito "
      << "con il numero progressivo " << zona << ": ";
      cin >> nomeZone[zona];
    }
}

...


int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;
  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if(num[zona] != 0)
```



```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "): ";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
  { cout << "nome della zona denotata nel seguito "
    << "con il numero progressivo " << zona << ": ";
    cin >> nomeZone[zona];
  }
}

...

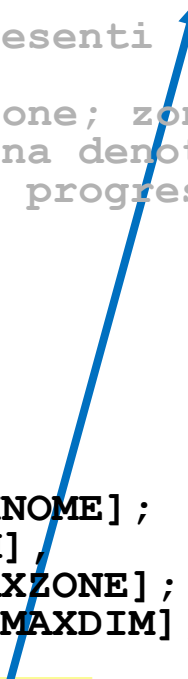
int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
      num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;
  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if(num[zona] != 0)
```



```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo " << MAXZONE << "): ";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
  { cout << "nome della zona denotata nel seguito "
    << "con il numero progressivo " << zona << ": ";
    cin >> nomeZone[zona];
  }
}

...

int main()
{ int numZone;
  char nomeZone[MAXZONE][MAXNOME];
  int prezzi[MAXZONE][MAXDIM],
    num[MAXZONE], somma[MAXZONE];
  float deviazioni[MAXZONE][MAXDIM];
  int zona;
  scegliZone(numZone, nomeZone);
  acquisisciRilevazioni(numZone, num, somma, prezzi);
  for (zona = 0; zona < numZone; zona++)
    if(num[zona] != 0)
```



## Prototipo di funzione

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

```
int main()  
{ ...  
}
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])  
{ int zona;  
  cout << "numero di zone presenti (massimo " << MAXZONE << "):";  
  cin >> numZone;  
  for (zona = 0; zona < numZone; zona++)  
  { cout << "nome della zona denotata nel seguito "  
    << "con il numero progressivo " << zona << ": ";  
    cin >> nomeZone[zona];  
  }  
}
```



## Intestazione o prototipo?

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

```
int main()  
{ ...  
}
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])  
{ int zona;  
  cout << "numero di zone presenti (massimo " << MAXZONE << "):";  
  cin >> numZone;  
  for (zona = 0; zona < numZone; zona++)  
  { cout << "nome della zona denotata nel seguito "  
    << "con il numero progressivo " << zona << ": ";  
    cin >> nomeZone[zona];  
  }  
}
```

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME])
```

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME])
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ int zona;
  cout << "numero di zone presenti (massimo "
        << MAXZONE << ") : ";
  cin >> numZone;
  for (zona = 0; zona < numZone; zona++)
  { cout << "nome della zona denotata nel seguito con"
        << " il numero progressivo " << zona << ": ";
    cin >> nomeZone[zona];
  }
}
```

## Prototipo

```
void scegliZone(int &, char[] [MAXNOME])
```

## Commento

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

## Intestazione

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME])
```

```
void scegliZone(int & numZone, char nomeZone[][MAXNOME]);  
{ ... }  
  
void acquisisciRilevazioni(int numZone, int num[],  
                           int somma[], int prezzi[][MAXDIM]);  
  
{ ... }  
  
void stampaIntestazione(const char nomeZona[]);  
{ ... }  
  
void calcolaDeviazioni(int somma, int num,  
                      const int prezzi[], float deviazioni[]);  
  
{ ... }  
  
void stampaDeviazioni(int num, const int prezzi[],  
                     const float deviazioni[]);  
  
{ ... }
```

```
main()  
{ ... }
```



```
#include <iostream>
using namespace std;
#include <iomanip.h>

Const int MAXDIM = 100,
        MAXZONE = 15,
        MAXNOME = 10;
void scegliZone(int & numZone, char nomeZone[][MAXNOME]);
void acquisisciRilevazioni(int numZone, int num[], int somma[],
                           int prezzi[][MAXDIM]);

void stampaIntestazione(const char nomeZona[]);
void calcolaDeviazioni(int somma, int num, const int prezzi[],
                       float deviazioni[]);

void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[]);

int main()
{ ...
}
```

```
#include <iostream>
using namespace std;
#include <iomanip.h>
```

```
Const int MAXDIM = 100,
        MAXZONE = 15,
        MAXNOME = 10;
```

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME]);
```

```
void acquisisciRilevazioni(int numZone, int num[], int somma[],
                           int prezzi[] [MAXDIM]);
```

```
void stampaIntestazione(const char nomeZona[]);
```

```
void calcolaDeviazioni(int somma, int num, const int prezzi[],
                       float deviazioni[]);
```

```
void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[]);
```

```
int main()
{ ...
}
```

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME])
```



Cut



Copy



Paste

```
#include <iostream>
using namespace std;
#include <iomanip.h>

Const int MAXDIM = 100,
        MAXZONE = 15,
        MAXNOME = 10;
void scegliZone(int & numZone, char nomeZone[][MAXNOME]);
void acquisisciRilevazioni(int numZone, int num[], int somma[],
                           int prezzi[][MAXDIM]);

void stampaIntestazione(const char nomeZona[]);
void calcolaDeviazioni(int somma, int num, const int prezzi[],
                       float deviazioni[]);

void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[]);

int main()
{ ...
}

void scegliZone(int & numZone, char nomeZone[][MAXNOME])
{ ...
}
```

```
#include <iostream>
using namespace std;
#include <iomanip.h>
```

```
Const int MAXDIM = 100,
        MAXZONE = 15,
        MAXNOME = 10;
```

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME]);
```

```
void acquisisciRilevazioni(int numZone, int num[], int somma[],
                           int prezzi[] [MAXDIM]);
```

```
void stampaIntestazione(const char nomeZona[]);
```

```
void calcolaDeviazioni(int somma, int num, const int prezzi[],
                       float deviazioni[]);
```

```
void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[]);
```

```
int main()
{ ...
}
```

```
void scegliZone(int & numZone, char nomeZone[] [MAXNOME])
{ ...
}
```



Cut



Copy



Paste

```
#include <iostream>
using namespace std;
#include <iomanip.h>

Const int MAXDIM = 100,
        MAXZONE = 15,
        MAXNOME = 10;
void scegliZone(int & numZone, char nomeZone[][MAXNOME]);
void acquisisciRilevazioni(int numZone, int num[], int somma[],
                           int prezzi[][MAXDIM]);

void stampaIntestazione(const char nomeZona[]);
void calcolaDeviazioni(int somma, int num, const int prezzi[],
                       float deviazioni[]);

void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[]);

int main()
{ ...
}
```

```
#include <iostream>
using namespace std;
#include <iomanip.h>

Const int MAXDIM = 100,
        MAXZONE = 15,
        MAXNOME = 10;
void scegliZone(int & numZone, char nomeZone[][MAXNOME]);
void acquisisciRilevazioni(int numZone, int num[], int somma[],
                           int prezzi[][MAXDIM]);

void stampaIntestazione(const char nomeZona[]);
void calcolaDeviazioni(int somma, int num, const int prezzi[],
                      float deviazioni[]);

void stampaDeviazioni(int num, const int prezzi[],
                     const float deviazioni[]);

int main()
{ ...
}
```

## Localizzato

```
#include <iostream>
using namespace std;
#include <iomanip.h>

Const int MAXDIM = 100,
        MAXZONE = 15,
        MAXNOME = 10;
void scegliZone(int & numZone, char nomeZone[][MAXNOME]);
void acquisisciRilevazioni(int numZone, int num[], int somma[],
                           int prezzi[][MAXDIM]);

void stampaIntestazione(const char nomeZona[]);
void calcolaDeviazioni(int somma, int num, const int prezzi[],
                       float deviazioni[]);

void stampaDeviazioni(int num, const int prezzi[],
                      const float deviazioni[]);

int main()
{ ...
}

...
```