



**POLITECNICO  
DI MILANO**

# **INFORMATICA**

**Discussione sul ruolo  
delle funzioni e del  
passaggio dei  
parametri**

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

void visualizzaDeviazioni(char nomeZona[], int somma, int num,
                          int prezzi[])
{ float media, deviazione;
  int i;
  if (num != 0) // se c'è almeno una rilevazione
  { cout << endl << setw(20)
    << "rilevazioni    zona " // crea intestazione
    << setw(10) << nomeZona << endl << endl // tabella per zona
    << setw(20) << "prezzi rilevati" // di nome nomeZona
    << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)somma / num; // ne calcola il prezzo medio
    for (i = 0; i < num; i++) // per ogni rilevazione, stampa:
    { deviazione = prezzi[i] - media;
      cout << setw(20) << prezzi[i] // una riga contenente il prezzo
        << setw(40) << setprecision(2) // rilevato e la deviazione
        << setiosflags (ios::fixed | ios::showpoint) // rispetto
        << deviazione // alla media, in formato fisso, con il
        << endl; // punto decimale e due cifre decimali
    }
  }
}

int main()
{ const int MAXDIM = 100, // massimo numero di rilevazioni per zona
    MAXZONE = 15; // massimo numero delle zone
    MAXNOME = 10; // massima lunghezza dei nomi delle zone
  int numZone; // numero delle zone
```

```
if (numC != 0) // se c'è almeno una rilevazione della zona centro
{
    cout << endl << setw(20)
        << "rilevazione    zona " // crea intestazione
        << setw(10) << "centro" << endl << endl // tabella
        << setw(20) << "prezzi rilevati" // per zona centro
        << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)sommaC / numC; // ne calcola il prezzo medio
    for (i = 0; i < numC; i++) // per ogni rilevazione, stampa una
    {
        deviazione = prezziC[i] - media; // riga contenente il
        cout << setw(20) << prezziC[i] // prezzo rilevato e la
            << setw(40) << setprecision(2) // deviazione rispetto alla
            << setiosflags (ios::fixed ! ios::showpoint) // media, in
            << deviazione // formato fisso, sempre con il punto
            << endl; // decimale e due cifre decimali
    }
}
```

```
if (numP != 0) // idem per la periferia
{
    cout << endl << setw(20) << "rilevazione    zona "
        << setw(10) << "periferia" << endl << endl
        << setw(20) << "prezzi rilevati"
        << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)sommaP / numP;
    for (i = 0; i < numP; i++)
    {
        deviazione = prezziP[i] - media;
        cout << setw(20) << prezziP[i]
            << setw(40) << setprecision(2)
            << setiosflags (ios::fixed ! ios::showpoint)
            << deviazione << endl;
    }
}
```

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

void visualizzaDeviazioni(char nomeZona[], int somma, int num,
                          int prezzi[])
{ float media, deviazione;
  int i;
  if (num != 0) // se c'è almeno una rilevazione
  { cout << endl << setw(20)
    << "rilevazioni   zona " // crea intestazione
    << setw(10) << nomeZona << endl << endl // tabella per zona
    << setw(20) << "prezzi rilevati" // di nome nomeZona
    << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)somma / num; // ne calcola il prezzo medio
    for (i = 0; i < num; i++) // per ogni rilevazione, stampa:
    { deviazione = prezzi[i] - media;
      cout << setw(20) << prezzi[i] // una riga contenente il prezzo
        << setw(40) << setprecision(2) // rilevato e la deviazione
        << setiosflags (ios::fixed | ios::showpoint) // rispetto
        << deviazione // alla media, in formato fisso, con il
        << endl; // punto decimale e due cifre decimali
    }
  }
}

int main()
{ const int MAXDIM = 100, // massimo numero di rilevazioni per zona
    MAXZONE = 15; // massimo numero delle zone
    MAXNOME = 10; // massima lunghezza dei nomi delle zone
  int numZone; // numero delle zone
```

```
        num[zona]++;)                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni(nomeZone[zona], somma[zona],
                        num[zona], prezzi[zona]);
}
```

```
if (numC != 0) // Se c'è almeno una rilevazione della zona centro
{
    cout << endl << setw(20)
        << "rilevazione    zona " // crea intestazione
        << setw(10) << "centro" << endl << endl // tabella
        << setw(20) << "prezzi rilevati" // per zona centro
        << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)sommaC / numC; // ne calcola il prezzo medio
    for (i = 0; i < numC; i++) // per ogni rilevazione, stampa una
    {
        deviazione = prezziC[i] - media; // riga contenente il
        cout << setw(20) << prezziC[i] // prezzo rilevato e la
            << setw(40) << setprecision(2) // deviazione rispetto alla
            << setiosflags (ios::fixed ! ios::showpoint) // media, in
            << deviazione // formato fisso, sempre con il punto
            << endl; // decimale e due cifre decimali
    }
}
```

## Un codice unico

```
if (numP != 0) // idem per la periferia
{
    cout << endl << setw(20) << "rilevazione    zona "
        << setw(10) << "periferia" << endl << endl
        << setw(20) << "prezzi rilevati"
        << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)sommaP / numP;
    for (i = 0; i < numP; i++)
    {
        deviazione = prezziP[i] - media;
        cout << setw(20) << prezziP[i]
            << setw(40) << setprecision(2)
            << setiosflags (ios::fixed ! ios::showpoint)
            << deviazione << endl;
    }
}
```



```
        num[zona]++;)                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni();
}
```

```

        num[zona]++;)                                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni();
}

```

## Dichiarate a livello globale

```

void visualizzaDeviazioni()
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2)
        << setiosflags (ios::fixed | ios::showpoint)
        << deviazione << endl;
    }
  }
}

```



```
        num[zona]++;)                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni();
}

void visualizzaDeviazioni()
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2)
        << setiosflags (ios::fixed | ios::showpoint)
        << deviazione << endl;
    }
  }
}
```

## Funzione immutata

```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    visualizzaDeviazioni("centro",    sommaC, numC, prezziC);  
    visualizzaDeviazioni("periferia",  sommaP, numP, prezziP);  
}
```



## Funzione immutata


```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    visualizzaDeviazioni("centro",    sommaC, numC, prezziC);  
    visualizzaDeviazioni("periferia",  sommaP, numP, prezziP);  
}
```

## Funzione immutata

```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    for (zona = 0; zona < numZone; zona++)  
        visualizzaDeviazioni(nomeZone[zona], somma[zona],  
                               num[zona], prezzi[zona]);  
}
```



## Funzione immutata

```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    for (zona = 0; zona < numZone; zona++)  
        visualizzaDeviazioni(nomeZone[zona], somma[zona],  
                               num[zona], prezzi[zona]);  
}
```

```
int prezzi[MAXZONE][MAXDIM];
```

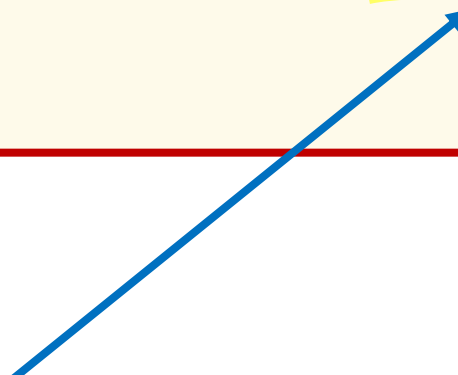
## Funzione immutata

```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    for (zona = 0; zona < numZone; zona++)  
        visualizzaDeviazioni(nomeZone[zona], somma[zona],  
                               num[zona], prezzi[zona]);  
}
```

```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

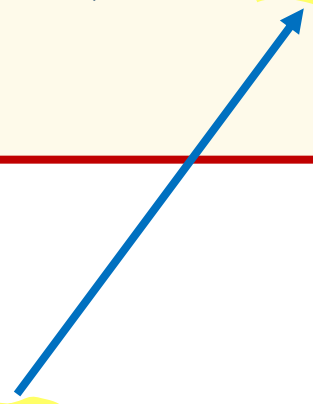
```
int main()  
{  
    ...  
    visualizzaDeviazioni( a, b, c, d );  
}
```






```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    visualizzaDeviazioni( a, b, c, d );  
}
```



```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    visualizzaDeviazioni( a, b, c, d );  
}
```



```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    visualizzaDeviazioni( a, b, c, d );  
}
```



```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    visualizzaDeviazioni( a, b, c, d );  
}
```

```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

Funzione  
altamente riutilizzabile

```
void visualizzaDeviazioni()
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio"
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2)
        << setiosflags (ios::fixed | ios::showpoint)
        << deviazione << endl;
    }
  }
}

int main()
{ ...
  for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni();
}
```

```
void visualizzaDeviazioni()  
{ float media, deviazione; int i;  
  if (num[zona] != 0)  
  { cout << endl << setw(20) << "rilevazioni      zona "  
    << setw(10) << nomeZone[zona] << endl << endl  
    << setw(20) << "prezzi rilevati "  
    << setw(40) << "deviazione rispetto al prezzo medio "  
    << endl;  
    media = (float)somma[zona] / num[zona];  
    for (i = 0; i < num[zona]; i++)  
    { deviazione = prezzi[zona][i] - media;  
      cout << setw(20) << prezzi[zona][i]  
        << setw(40) << setprecision(2)  
        << setiosflags (ios::fixed | ios::showpoint)  
        << deviazione << endl;  
    }  
  }  
}  
  
int main()  
{ ...  
  for (zona = 0; zona < numZone; zona++)  
    visualizzaDeviazioni();  
}
```



```
const int MAXDIM = 100, MAXZONE = 15, MAXNOME = 10;
char nomeZone[MAXZONE][MAXNOME];
int prezzi[MAXZONE][MAXDIM],
    num[MAXZONE], somma[MAXZONE];

void visualizzaDeviazioni(int zona)
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio"
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2)
        << setiosflags (ios::fixed | ios::showpoint)
        << deviazione << endl;
    }
  }
}

int main()
{ ...
  for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni();
}
```

Variabili globali

```
const int MAXDIM = 100, MAXZONE = 15, MAXNOME = 10;
char nomeZone[MAXZONE][MAXNOME];
int prezzi[MAXZONE][MAXDIM],
    num[MAXZONE], somma[MAXZONE];

void visualizzaDeviazioni()
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati "
    << setw(40) << "deviazione rispetto al prezzo medio"
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2)
        << setiosflags (ios::fixed | ios::showpoint)
        << deviazione << endl;
    }
  }
}

int main()
{ ...
  for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni();
}
```

Variabili globali

```
const int MAXDIM = 100, MAXZONE = 15, MAXNOME = 10;
char nomeZone[MAXZONE][MAXNOME];
int prezzi[MAXZONE][MAXDIM],
    num[MAXZONE], somma[MAXZONE];
```

```
void visualizzaDeviazioni()
{
    ...
}
```

```
void miaFunzione()
{
    ...
}
```

```
int main()
{
    ...
    for (zona = 0; zona < numZone; zona++)
        visualizzaDeviazioni();
}
```



```
const int MAXDIM = 100, MAXZONE = 15, MAXNOME = 10;
char nomeZone[MAXZONE][MAXNOME];
int prezzi[MAXZONE][MAXDIM],
    num[MAXZONE], somma[MAXZONE];
```

```
void visualizzaDeviazioni()
{
    ...
}
```

```
void miaFunzione()
{
    ...
    for (i = 0; i < MAXZONE; i++)
        { for (n = 0; n < MAXDIM; n++)
            prezzi[i][n] = 0;
          }
}
```

```
int main()
{
    ...
    for (zona = 0; zona < numZone; zona++)
        visualizzaDeviazioni();
}
```

```
const int MAXDIM = 100, MAXZONE = 15, MAXNOME = 10;
char nomeZone[MAXZONE][MAXNOME];
int prezzi[MAXZONE][MAXDIM],
    num[MAXZONE], somma[MAXZONE];

void visualizzaDeviazioni()
{
    ...
}

void miaFunzione()
{
    ...
    for (i = 0; i < MAXZONE; i++)
    { for (n = 0; n < MAXDIM; n++)
        prezzi[i][n] = 0;
    }
}

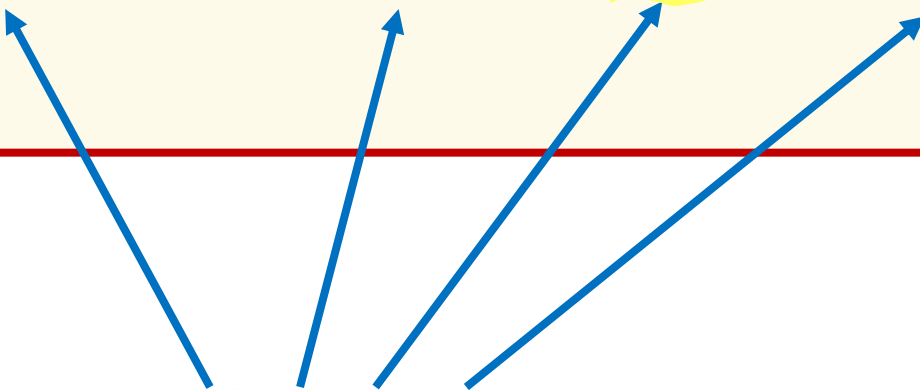
int main()
{
    ...
    for (zona = 0; zona < numZone; zona++)
        visualizzaDeviazioni();
}
```

# 1) Riutilizzabilità

## 2) Limitazione degli errori

```
void visualizzaDeviazioni  
    (char nomeZona[], int somma, int num, int prezzi[])  
{  
    ...  
}
```

```
int main()  
{  
    ...  
    visualizzaDeviazioni(a, b, c, d);  
}
```

A diagram with four blue arrows pointing from the arguments 'a', 'b', 'c', and 'd' in the main function call to the corresponding parameters 'nomeZona[]', 'somma', 'num', and 'prezzi[]' in the function signature of visualizzaDeviazioni.


```
        num[zona]++;)                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni(nomeZone[zona], somma[zona],
                        num[zona, prezzi[zona]]);
}
```



```
        num[zona]++;)                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
    visualizzaDeviazioni();
}

void visualizzaDeviazioni()
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2)
        << setiosflags (ios::fixed | ios::showpoint)
        << deviazione << endl;
    }
  }
}
```

```
        num[zona]++;)                                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio"
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2)
        << setiosflags (ios::fixed | ios::showpoint)
        << deviazione << endl;
    }
  }
}
```



```
        num[zona]++;)                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "prezzi rilevati" << endl
    << setw(10) << "deviazione rispetto al prezzo medio" << endl
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(10) << "prezzo" << setw(10) << "deviazione" << endl
        << setw(40) << setprecision(2)
        << setiosflags (ios::fixed | ios::showpoint)
        << deviazione << endl;
    }
  }
}
```

**Identificare  
un sottoproblema  
e la sua soluzione**

```
        num[zona]++;)                                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
{ float media; deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni    zona "
    << setw(10) << nomeZona[zona] << endl << endl
    << setw(20) << "prezzi rispetto al prezzo medio"
    << setw(40) << "deviazione rispetto al prezzo medio"
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
        deviazione = prezzi[zona][i] - media;
    cout << setw(20) << prezzo[zona][i]
    << setw(40) << setprecision(2)
    << setiosflags (ios::fixed | ios::showpoint)
    << deviazione << endl;
  }
}
```

**Problema**

**Sottoproblema1**  
**Funzione1**

**Sottoproblema2**  
**Funzione2**

**Sottoproblema3**  
**Funzione3**

# Problema

Sottoproblema1

Funzione1

Sottoproblema2

Funzione2

Sottoproblema3

Funzione3

```
num[zona]++;) // gestisce la rilevazione
else // altrimenti segnala zona
cout << "zona non corretta: inserire nuovo valore" << endl;
}
// stampa di una tabella separata per ogni zona,
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
{ float media; deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi zona " << nomeZone[zona] << endl
    << setw(40) << "deviazione rispetto al prezzo medio" << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
      deviazione = prezzi[zona][i] - media;
    cout << setw(20) << prezzi[zona][i] << endl
    << setw(40) << setprecision(2) << deviazione << endl;
  }
}
```

Parametri formali

**Problema**

**Sottoproblema1**

**Funzione1**

**Sottoproblema2**

**Funzione2**

**Sottoproblema3**

**Funzione3**

```
        num[zona]++;)                                // gestisce 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
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio"
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2) << deviazione << endl;
    }
  }
}
```

**main()****visualizzaDeviazioni****Uscita**



```
        num[zona]++;)                                // gestisce la rilevazione
    else                                              // altrimenti segnala zona
        cout << "zona non corretta: inserire numero corretto" << endl;
}
// stampa di una tabella separata per ogni zona
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio"
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2) << deviazione << endl;
    }
  }
}
```

main()

Teorema di Fermat

elevaAPotenza

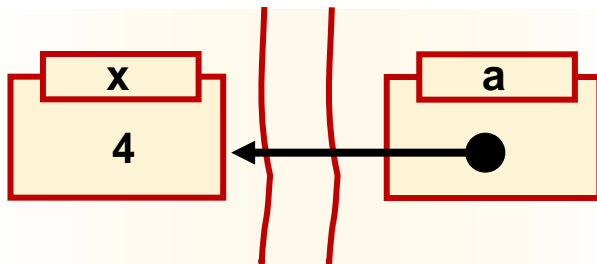
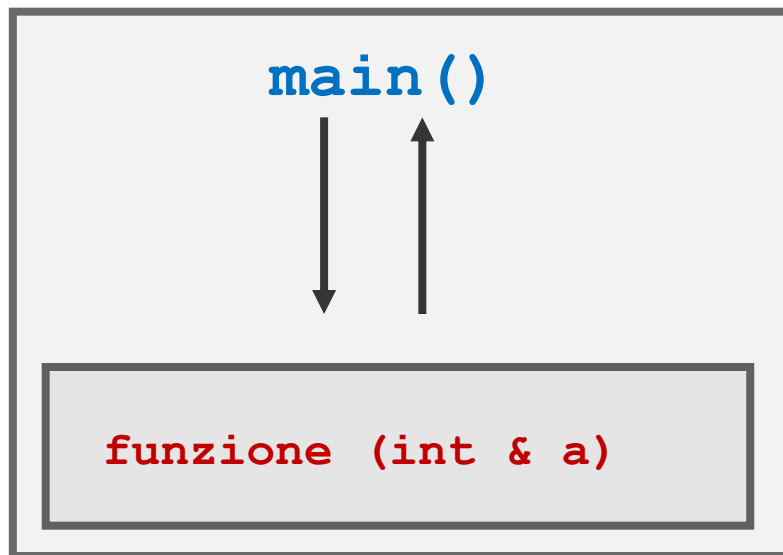
```
        num[zona]++;)                // gestisce la rilevazione
    else                             // altrimenti segnala zona
        cout << "zona non corretta: inserisci zona corretta" << endl;
}
// stampa di una tabella separata per ogni zona, con i prezzi rilevati e la deviazione di ogni rilevazione rispetto alla media della zona
// deviazione di ogni rilevazione rispetto alla media della zona
for (zona = 0; zona < numZone; zona++)
{ float media, deviazione; int i;
  if (num[zona] != 0)
  { cout << endl << setw(20) << "rilevazioni      zona "
    << setw(10) << nomeZone[zona] << endl << endl
    << setw(20) << "prezzi rilevati"
    << setw(40) << "deviazione rispetto al prezzo medio"
    << endl;
    media = (float)somma[zona] / num[zona];
    for (i = 0; i < num[zona]; i++)
    { deviazione = prezzi[zona][i] - media;
      cout << setw(20) << prezzi[zona][i]
        << setw(40) << setprecision(2) << deviazione << endl;
    }
  }
}
```

Passaggio per indirizzo

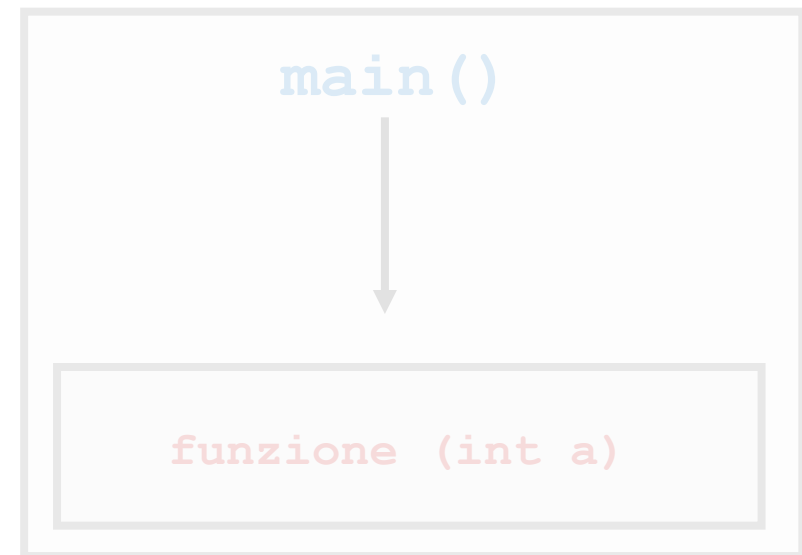
main()

elevaAPotenza

## Passaggio per indirizzo

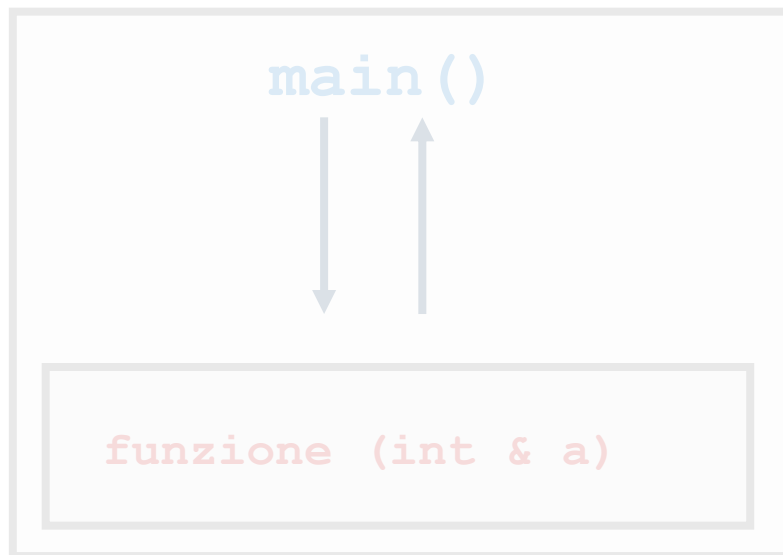


## Passaggio per valore

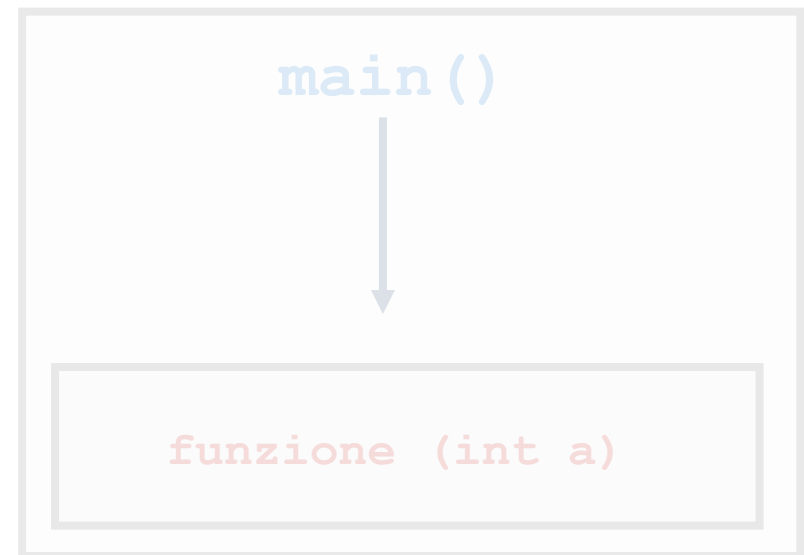


```
void funzione(int a)
{ ...
  a = a + 1;
}
```

## Passaggio per indirizzo

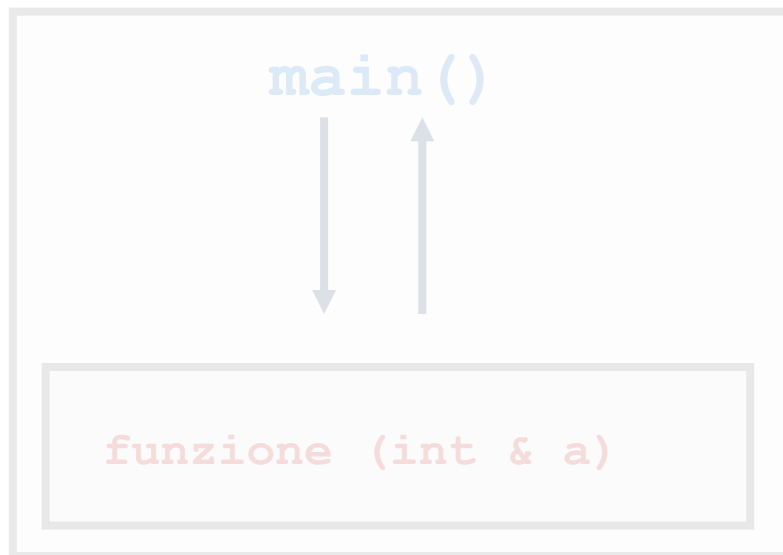


## Passaggio per valore

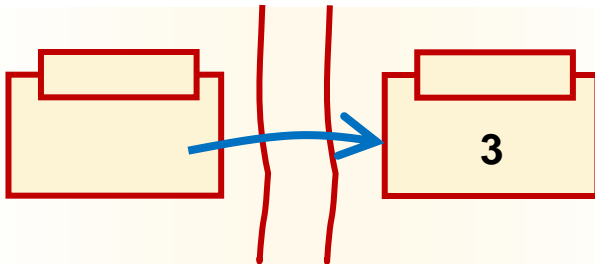
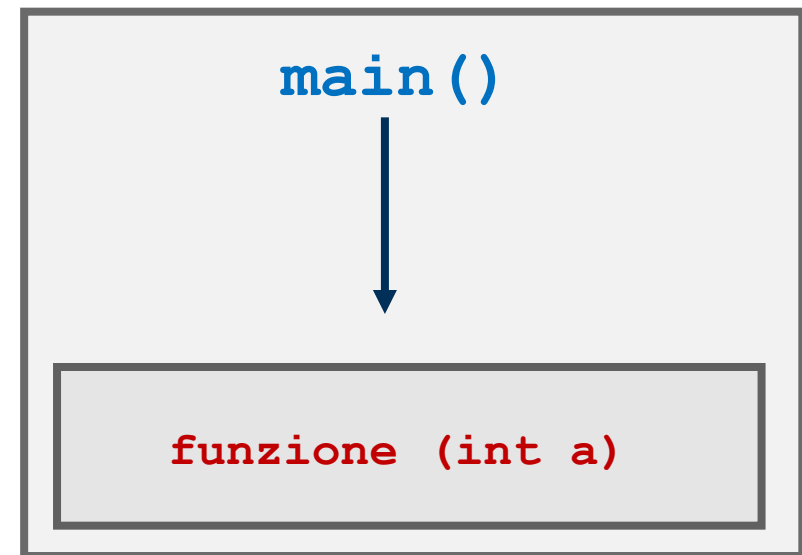


Ingresso e uscita

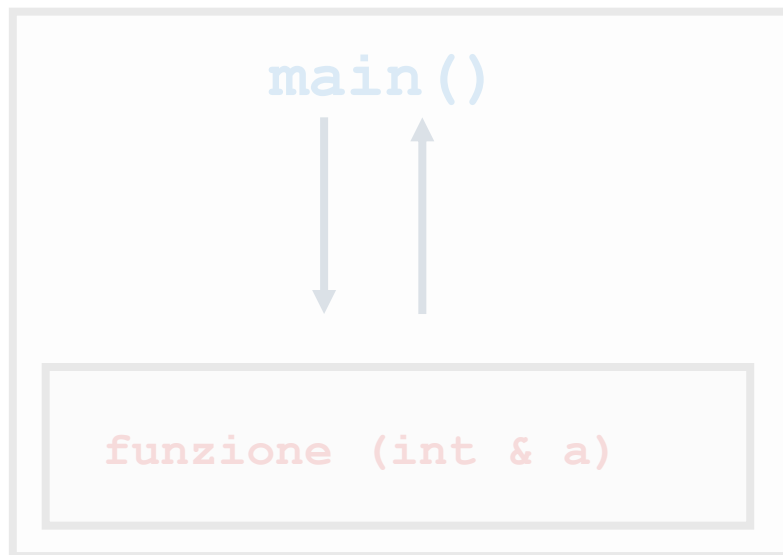
## Passaggio per indirizzo



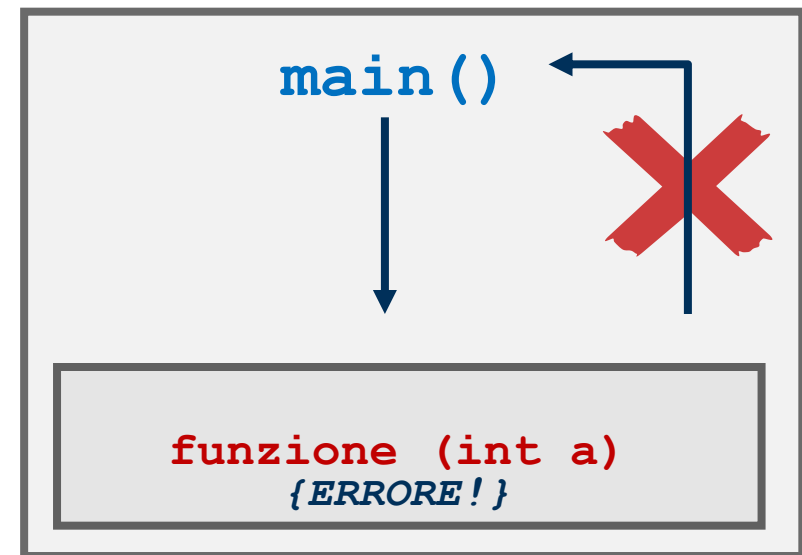
## Passaggio per valore



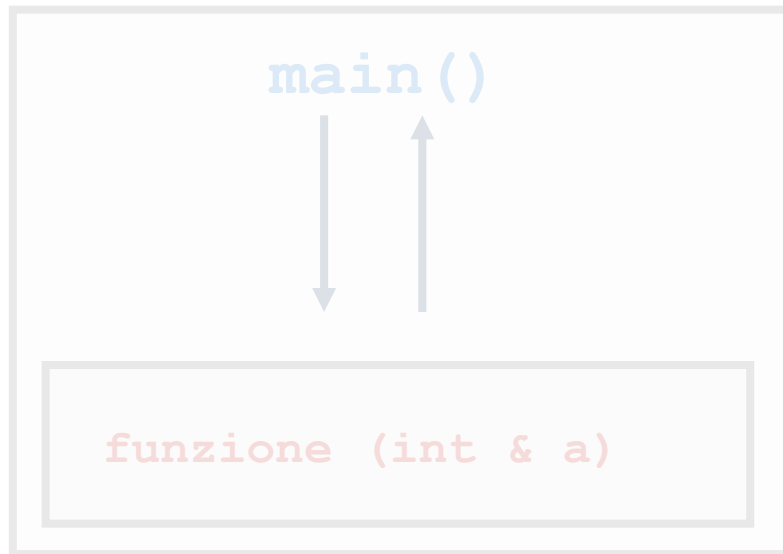
## Passaggio per indirizzo



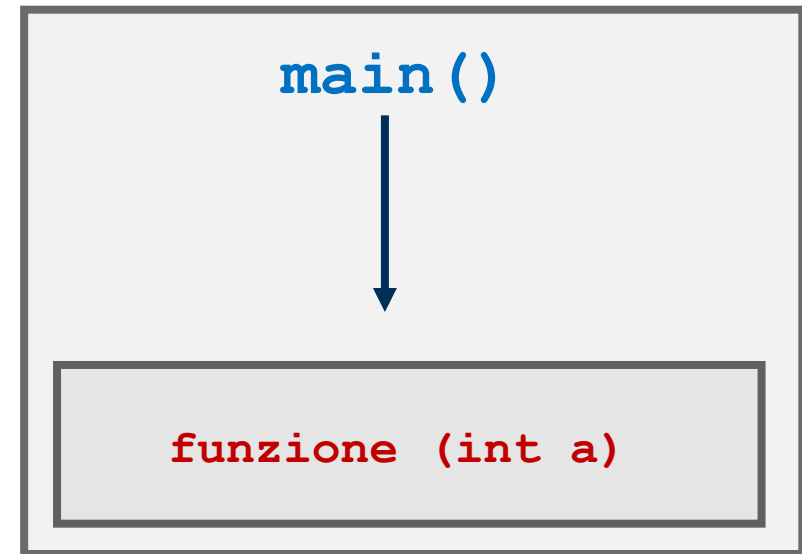
## Passaggio per valore



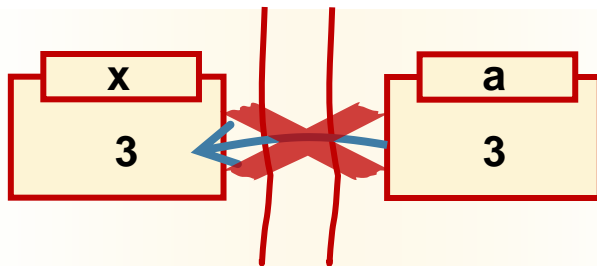
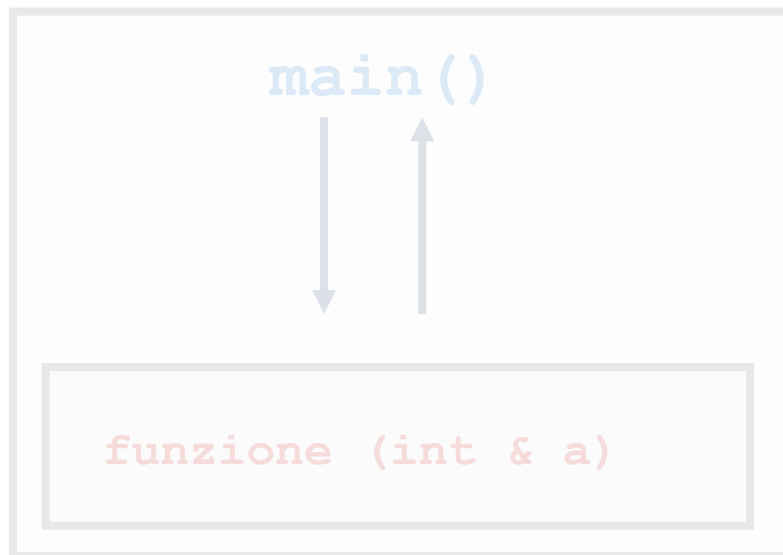
## Passaggio per indirizzo



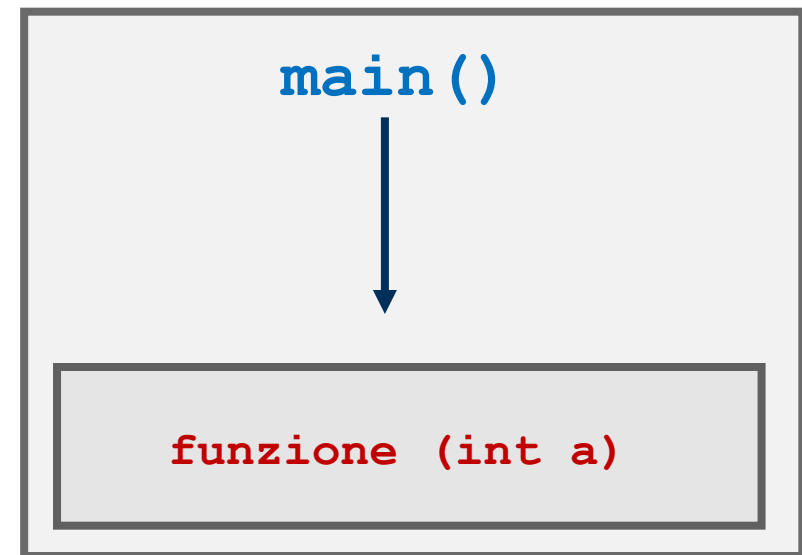
## Passaggio per valore



## Passaggio per indirizzo



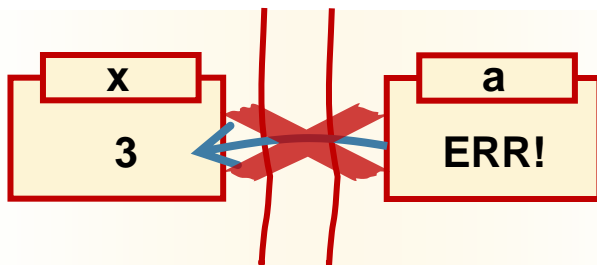
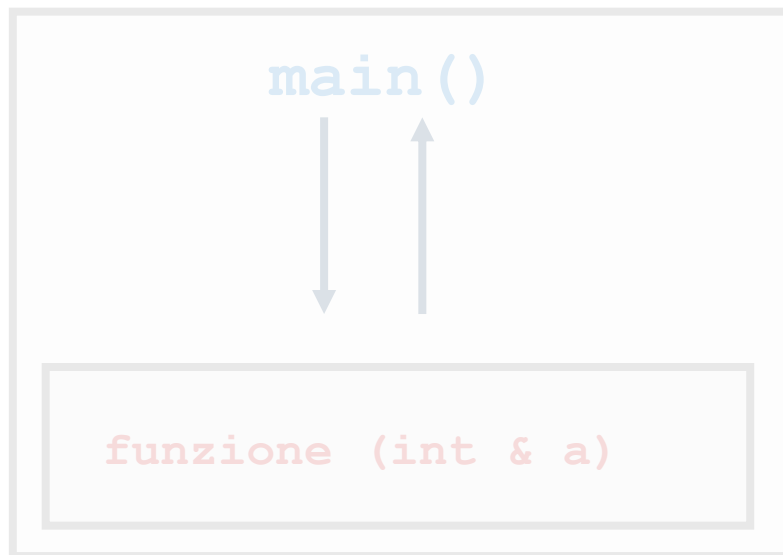
## Passaggio per valore



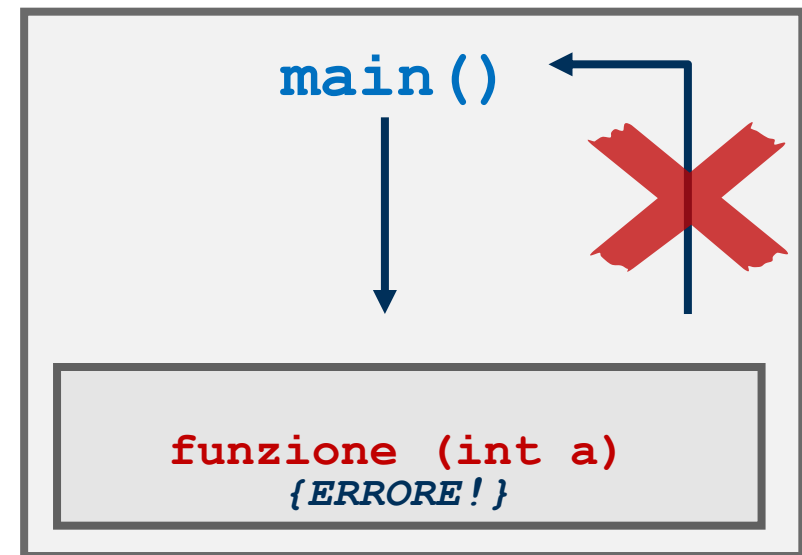
```
void funzione(int a)
{
    ...
    a = 1;
}
```



## Passaggio per indirizzo

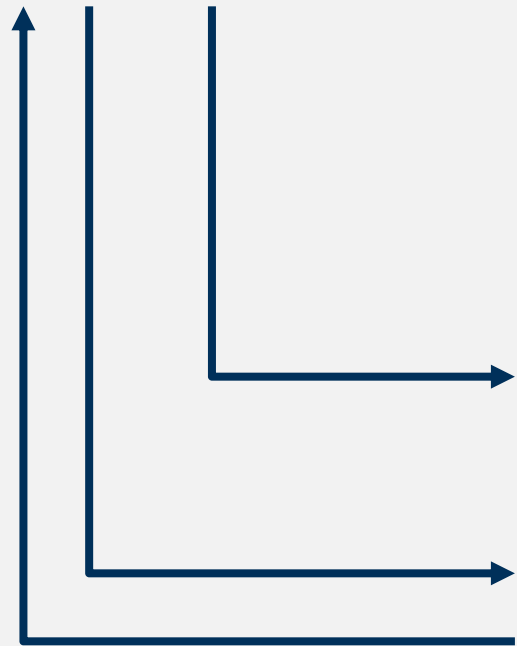


## Passaggio per valore



Passaggio dei parametri

*Problema*

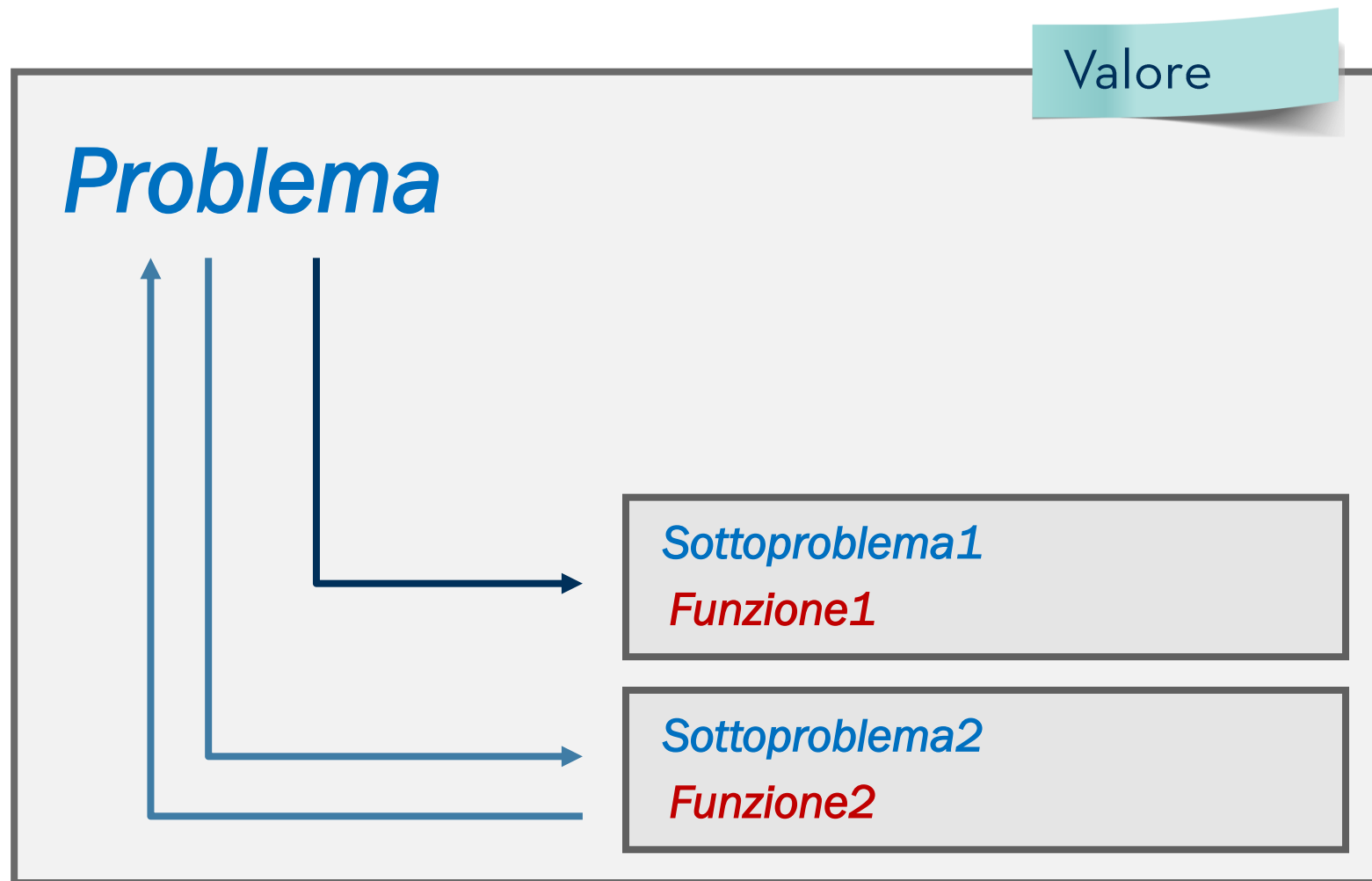


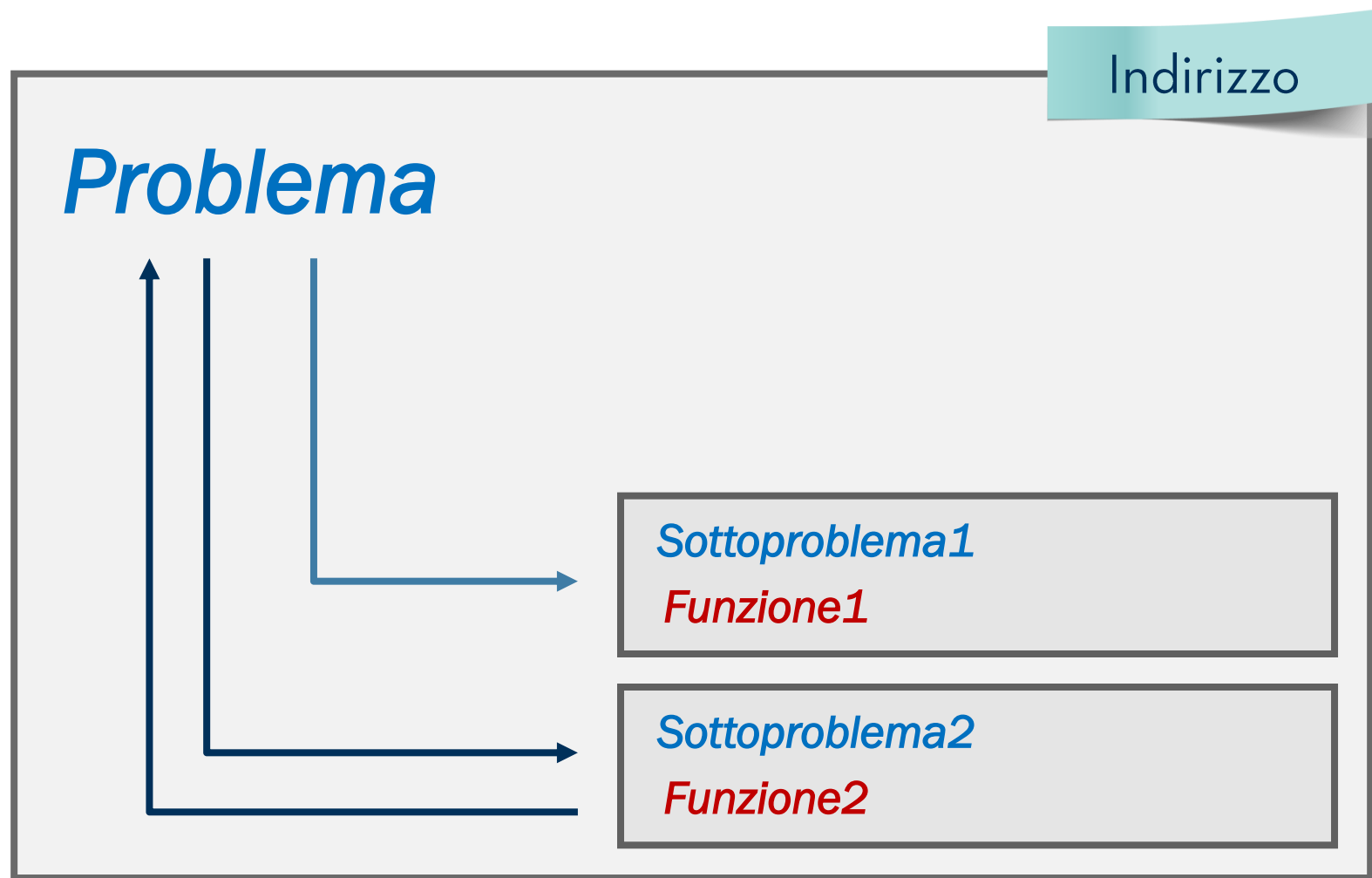
*Sottoproblema1*

*Funzione1*

*Sottoproblema2*

*Funzione2*





## Passaggio per valore

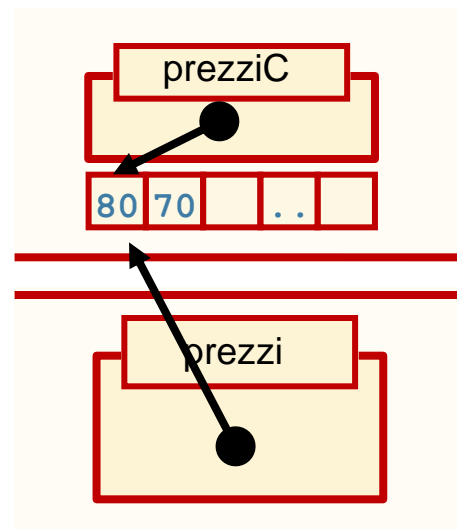
```
void visualizzaDeviazioni  
(char nomeZona[], int somma, int num, int prezzi[])
```

```
int main()  
{  
  ...  
  visualizzaDeviazioni( "centro", sommaC, numC, prezziC );  
}
```



```
void visualizzaDeviazioni  
  (char nomeZona[], int somma, int num, int prezzi[])
```

```
int main()  
{ ...  
  visualizzaDeviazioni( "centro", sommaC, numC, prezziC );  
}
```

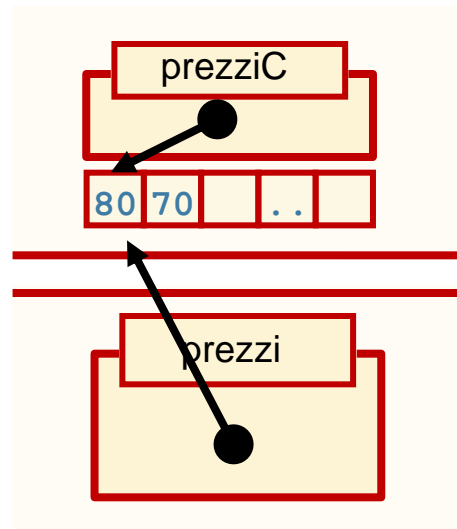


```
void visualizzaDeviazioni  
  (char nomeZona[], int somma, int num, int prezzi[])
```

```
int main()  
{  
  ...  
  visualizzaDeviazioni( "centro", sommaC, numC, prezziC );  
}
```

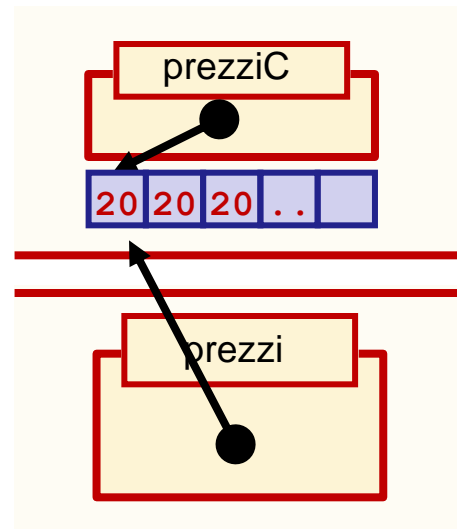
*Opero direttamente  
sul parametro effettivo*

```
void visualizzaDeviazioni  
  (char nomeZona[], int somma, int num, int prezzi[])  
{  
  ...  
  deviazione = prezzi[i] - media;  
  ...  
}
```

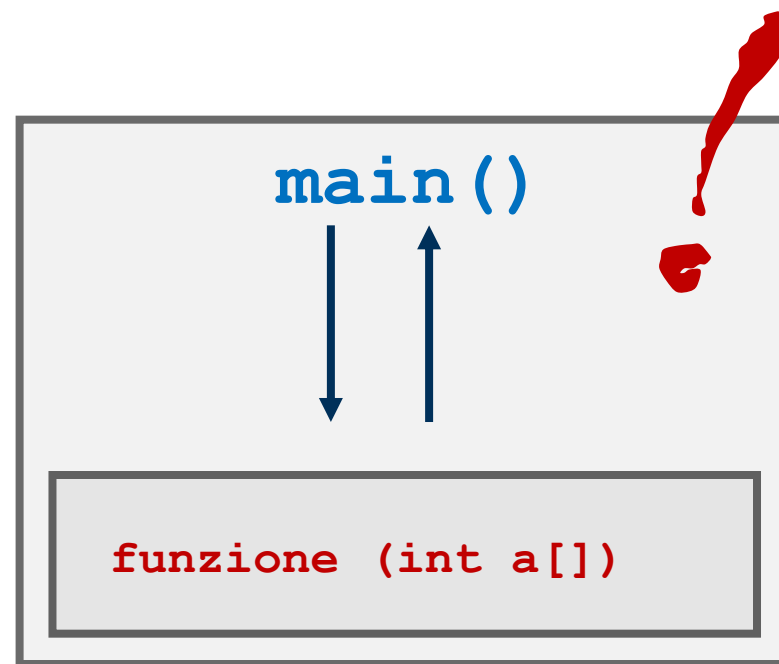




```
void visualizzaDeviazioni  
  (char nomeZona[], int somma, int num, int prezzi[])  
{ ...  
  prezzi[i] = 20;  
  ...  
}
```

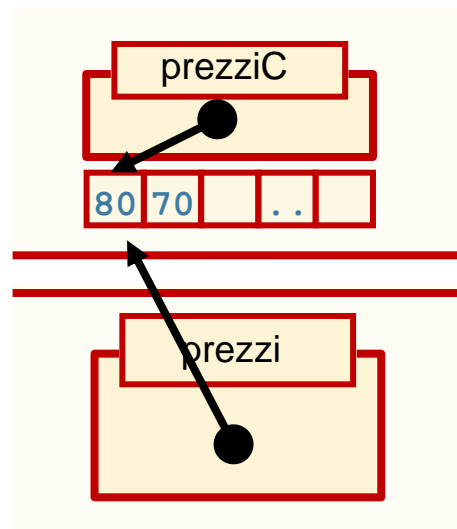


```
void visualizzaDeviazioni  
  (char nomeZona[], int somma, int num, int prezzi[])
```



const

```
void visualizzaDeviazioni  
  (char nomeZona[], int somma, int num, int prezzi[])
```



const

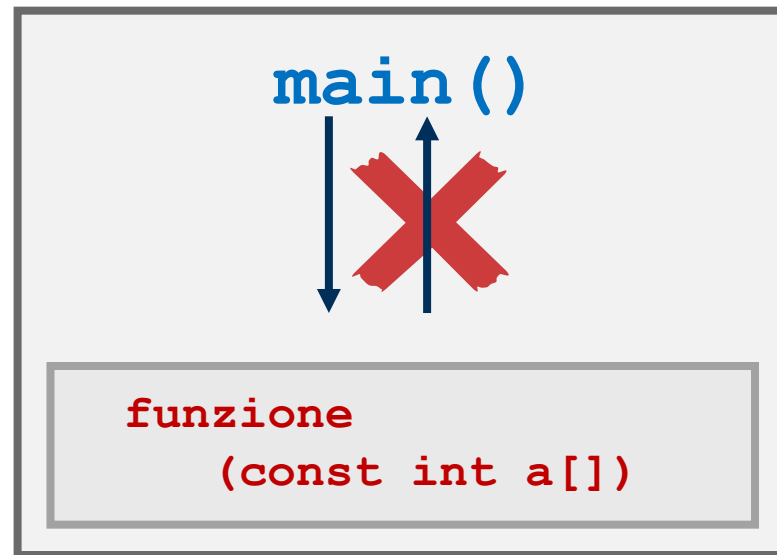
```
void visualizzaDeviazioni  
(const char nomeZona[], int somma, int num, const int prezzi[])
```

const

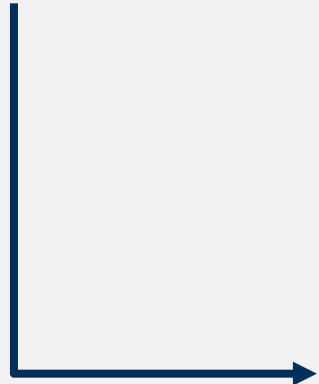
```
void visualizzaDeviazioni  
(const char nomeZona[], int somma, int num, const int prezzi[])  
{  
    ...  
    prezzi[i] = 20;  
    ...  
}
```

const

```
void visualizzaDeviazioni  
(const char nomeZona[], int somma, int num, const int prezzi[])
```



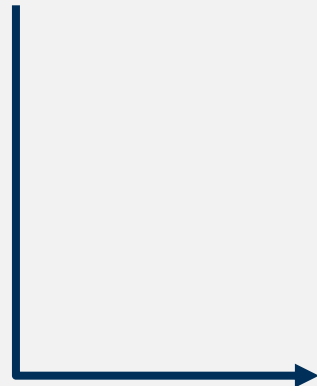
*Problema*



*Sottoproblema1*

`visualizzaDeviazioni(...)`

*Problema*



*Calcolo e stampa*

`visualizzaDeviazioni(...)`



*Problema*

Valore

*Calcolo deviazione*

`funzione(... scarti[])`

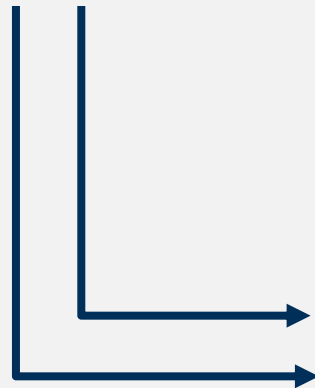
## *Problema*

Valore senza "const"

*Calcolo deviazione*

`funzione(... scarti[])`

## *Problema*



*Calcolo deviazione*

```
calcolaDeviazioni(...scarti[])
```