Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Programmazione e Laboratorio di Programmazione

Manualistica 02
Gli operatori

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore &

- **Sintassi :**

  &nome_variabile

- **Valore:**

  indirizzo della variabile
  nome_variabile

- **Esempio:**

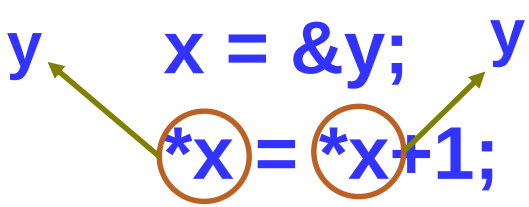
```
int *x;
int y;

x = &y;
```

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore *

- **Sintassi :**
  - **espr_di _tipo_indirizzo**
- **“Valore”:**
  - **il nome della variabile il cui indirizzo coincide con il valore di espr_di _tipo_indirizzo**
- **Esempio:**

```
int *x, y;
y = 1;
x = &y;
*x = *x+1;
```



Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di assegnamento

- **Sintassi:**

  nome_variabile = espressione

  espressione

- **Valore:**

  è il valore di espressione

- **Modifiche allo stato della memoria:**

  assegna alla variabile nome_variabile il valore di espressione

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di assegnamento

- **Attenzione:**

  il tipo di nome_variabile e quello di espressione "devono" coincidere

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di assegnamento

- **Esempio:**

```
/* sorgente: OpAss_1.c */
/* programma che mostra il comportamento
** dell'operatore di assegnamento */
#include <stdio.h>
int main ()
        {
        /* definizione e inizializzazione delle variabili */
        int x, y;
        y=2;
        /* visualizza il valore di y assegnandolo
        ** al tempo stesso a x */
        printf("\nY: %d", x=y);
        /* visualizza il valore di x */
        printf("\nX: %d", x);
        return(1);
        }
```

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di assegnamento

- **Sintassi:**

  *nome_puntatore = espressione

  espressione

- **Valore:**

  è il valore di espressione

- **Modifiche allo stato della memoria:**

  assegna alla variabile il cui indirizzo è memorizzato in nome_puntatore il valore di espressione

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di assegnamento

- **Esempio:**

```
/* sorgente: OpAss_2.c */
/* programma che mostra il comportamento
** dell'operatore di assegnamento */
#include <stdio.h>
int main ()
        {
        /* definizione e inizializzazione delle variabili */
        int x, y, *punx, *puny;
        punx=&x; puny=&y;
        *puny=2;
        /* visualizza il valore di y assegnandolo
        ** al tempo stesso a x */
        printf("\nY: %d", *punx=*puny);
        /* visualizza il valore di x */
        printf("\nX: %d", *punx);
        return(1);
        }
```

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di auto-incremento prefisso

- **Sintassi:**

  ++nome_variabile

- **Valore:**

  il valore di nome_variabile incrementato di 1

- **Modifiche allo stato della memoria:**

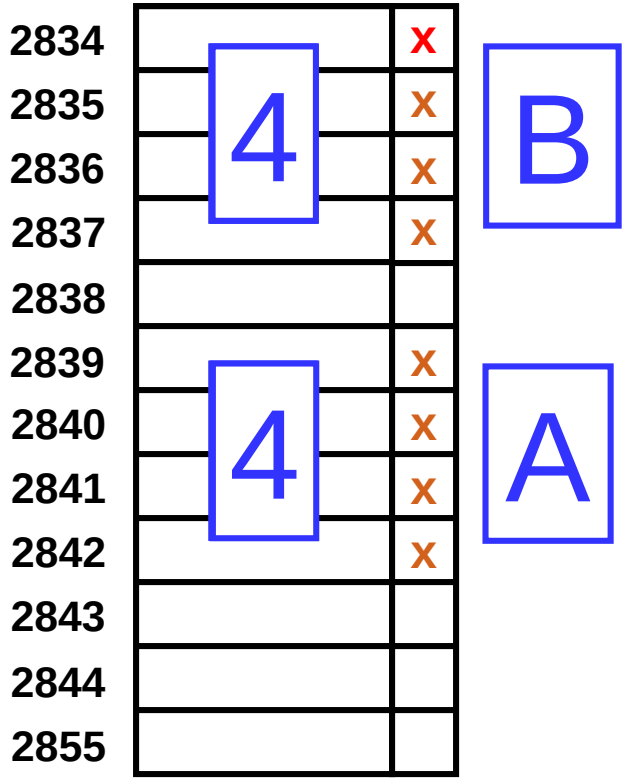
  incrementa di 1 il valore di nome_variabile

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di auto-incremento prefisso

- **Esempio:**

B=++A;



Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di auto-incremento postfisso

- **Sintassi:**

  nome_variabile++

- **Valore:**

  il valore di nome_variabile

- **Modifiche allo stato della memoria:**

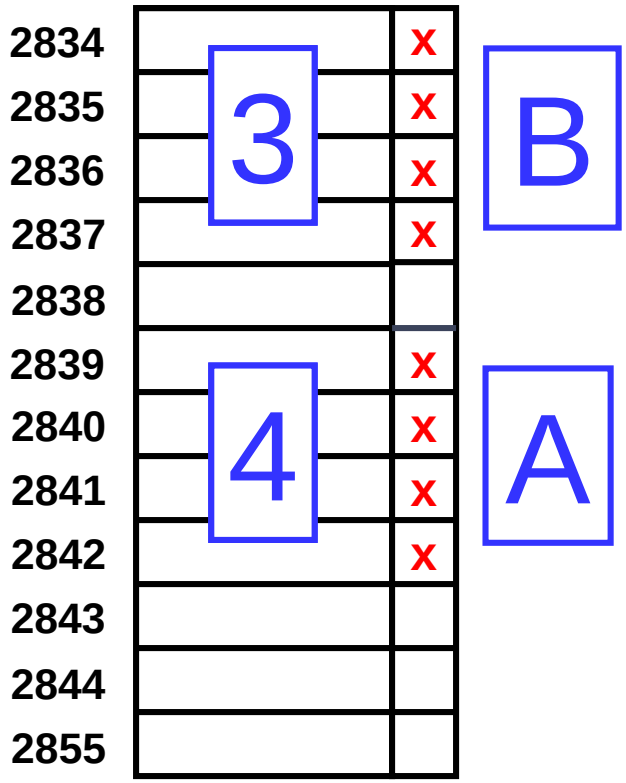
  incrementa di 1 il valore di nome_variabile

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di auto-incremento postfisso

- **Esempio:**

B=A++;



Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di auto-decremento prefisso

- **Sintassi:**

  --nome_variabile

- **Valore:**

  il valore di nome_variabile decrementato di 1

- **Modifiche allo stato della memoria:**

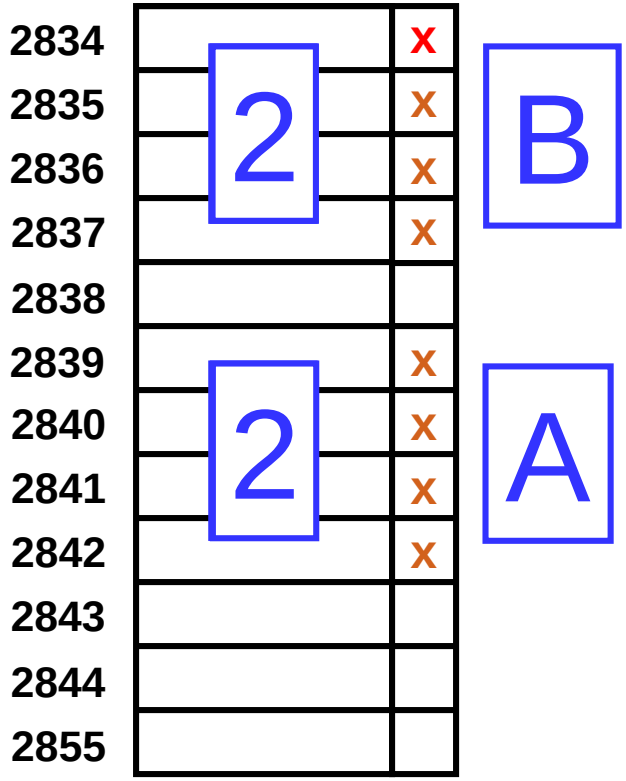
  decrementa di 1 il valore di nome_variabile

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di auto-decremento prefisso

- **Esempio:**

B=--A;



Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di auto-decremento postfisso

- **Sintassi:**

  nome_variabile--

- **Valore:**

  il valore di nome_variabile

- **Modifiche allo stato della memoria:**

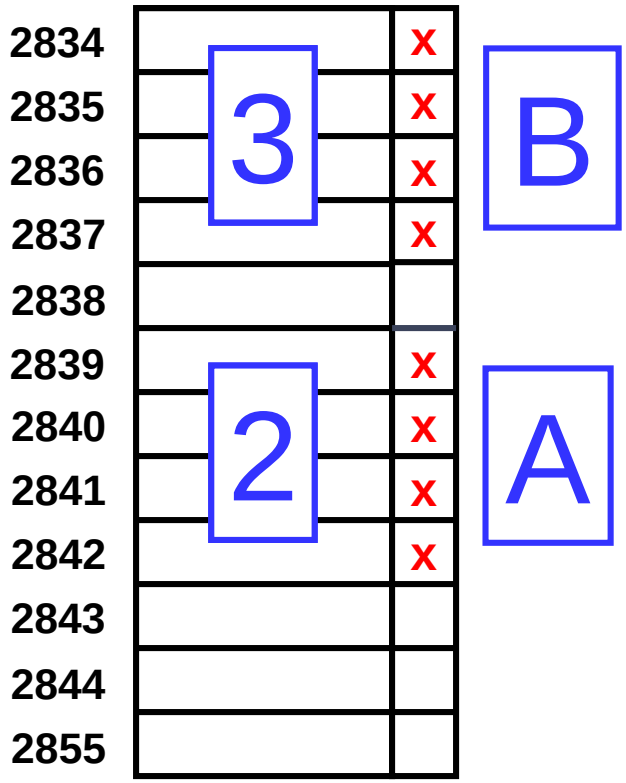
  decrementa di 1 il valore di nome_variabile

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatore di auto-decremento postfisso

- **Esempio:**

B=A--;



Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatori di auto-incremento e decremento

- **Esempio**

```
/* Mostra il comportamento degli operatori
** di autoincremento e autodecremento */

#include <stdio.h>

int main ()
  {
  int y = 1;

  printf("\nValore di y +suffisso = %d\n", y++);
  printf("Valore di y +prefisso = %d\n", ++y);
  printf("Valore di y -prefisso = %d\n", --y);
  printf("Valore di y -suffisso = %d\n", y--);
  printf("Valore finale di y = %d\n", y);

  return(1);
  }
```

1
3
2
2
1

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatori di relazione

- **Sintassi:**
  - a) espr_1 == espr_2 — uguale
  - b) espr_1 != espr_2 — diverso
  - c) espr_1 > espr_2 — maggiore
  - d) espr_1 >= espr_2 — maggiore o uguale
  - e) espr_1 < espr_2 — minore
  - f) espr_1 <= espr_2 — minore o uguale
- **Valore:**
  - 1 se i valori di espr_1 e espr_2 si trovano nella relazione specificata
  - 0 altrimenti

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatori di relazione

- **Esempio**

```
/* Sorgente: OpRel.c */

/* programma che mostra il comportamento degli
** operatori di relazione */

#include <stdio.h>

int main ()
  {
  printf ("\nValore di 3 < 5: %d\n",(3 < 5));
  printf ("Valore di 3 > 5: %d\n",(3 > 5));
  printf ("Valore di 3 == 5: %d\n",(3 == 5));
  printf ("Valore di 3 != 5: %d\n",(3 != 5));
  printf ("Valore di 5 == 5: %d\n",(5 == 5));

  printf ("Numero di relazioni soddisfatte: %d", \
        (3 < 5)+(3 > 5)+(3 == 5)+(3 != 5)+(5 == 5));

  return(1);
  }
```

```
1
0
0
1
1
3
```

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatori “logici”

- **Sintassi:**

  $espr_1$ || $espr_2$ “or”

- **Valore:**

  0 se $espr_1$ e $espr_2$ valgono entrambe 0

  1 altrimenti

- **Sintassi:**

  $espr_1$ && $espr_2$ “and”

- **Valore:**

  1 se $espr_1$ e $espr_2$ sono entrambe ≠ 0

  0 altrimenti

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatori “logici”

- **Sintassi:**

  !espr **“not”**

- **Valore:**

  1 **se espr vale 0**

  0 **altrimenti**

Carlo Gaibisso
Bruno Martino
Marco
Pietrosanto

Operatori “logici”

- **Esempio**

```
/* sorgente: OpLog.c */

/* programma che illustra il comportamento degli
** operatori logici */

#include <stdio.h>

int main ()
  {
  printf ("\nValore di !(3 > 5): %d\n", !(3 > 5));

  printf ("Valore di ((3 < 5) || (3 > 5)): %d\n", \
        ((3 < 5) || (3 > 5)));

  printf ("Valore di ((3 < 5) && (3 > 5)): %d\n", \
        ((3 < 5) && (3 > 5)));

  return(1);
  }
```

```
1
1
0
```