```
int fact ( int n )
{ int i;
  int f = 1;
  for (i = n; i > 1; i--)
      f = f * i;
  return f;
}
```

nome

tipo del risultato

dichiarazioni

Definizione

```
main()
{ int x = 4;
  int y = 6;
  x = fact(x);
  y = fact(y);
}
```

chiamate

espressione x+3 ← la chiamata di funzione PUÒ MODIFICARE LO STATO

dà un valore

Come si collegano gli argomenti della chiamata, x e y, con il parametro della funzione, n?

Notazione matematica

$f : \mathbb{N} \to \mathbb{N}$

$f(n) = n^2 + 2n + 2$

definizione

nome funzione

nome parametro

definizione parametrica

$f(3)$

collegamento dell'argomento al parametro

Si sostituisce al posto di n nella definizione il valore dell'argomento, 3.

$f(3) = 3^2 + 2 \cdot 3 + 2$

$= \{\text{calcolo}\}$

17

lunedì 24 settembre 2018 09:30

lunedì 24 settembre 2018 09:30

In C (in generale nei linguaggi imperativi) il collegamento tra argomento e parametro di una funzione deve essere fatto ATTRAVERSO LO STATO (ambiente e memoria)

→ MODALITÀ DI PASSAGGIO DEI PARAMETRI

In C la modalità di passaggio dei parametri si chiama:

MODALITÀ PER VALORE

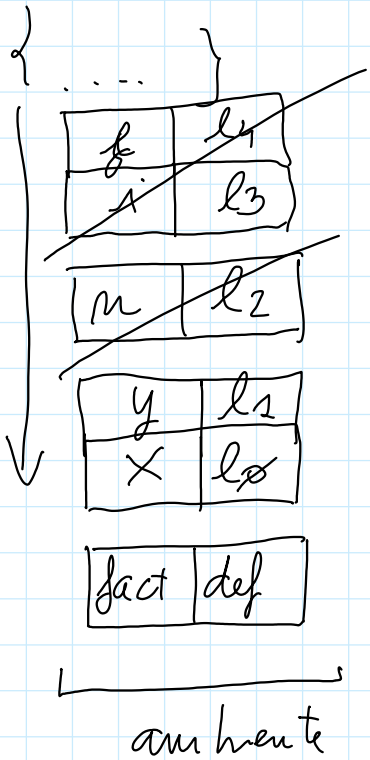
Quando si chiama una funzione si aggiunge nell'ambiente e nella memoria un NUOVO FRAME che contiene nuove variabili corrispondenti ai parametri. A queste nuove variabili viene dato il Valore degli argomenti.

Dopo si esegue il blocco che definisce la funzione

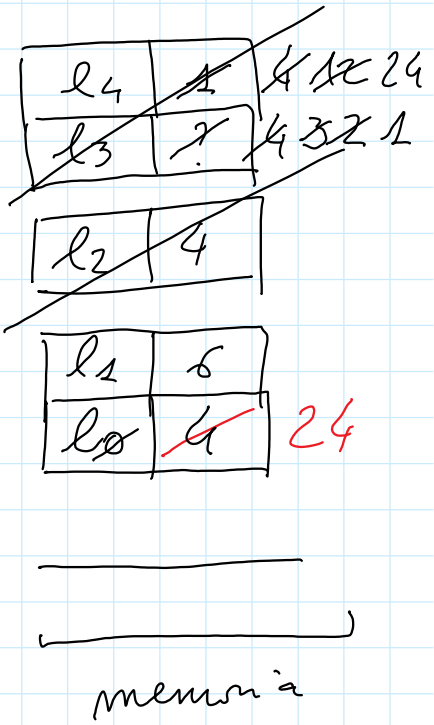
```
int fact (int n) { ... }

main ()
{ int x = 4;
  int y = 6;
  x = fact (x);
  .
  .
  .
```

=24



ambiente



memoria

lunedì 24 settembre 2018 09:47

Funzioni, quando chiamate, modif. come lo stato e RESTITUISCONO UN VALORE

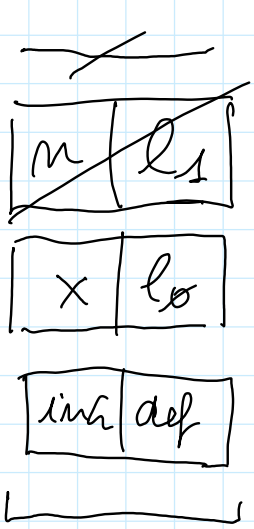
Procedure, quando chiamate, modif. come lo stato!

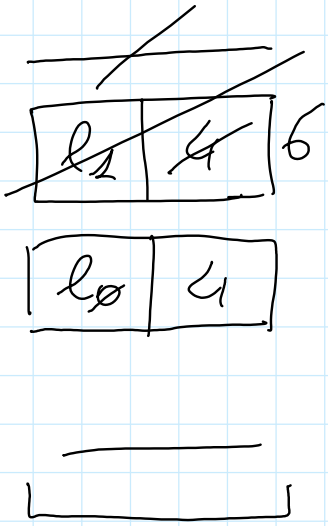
```
void incr (int n)
{
  n = n+2;
}
```

tipo del risultato

```
main()
{ int x = 4;
  incr(x);
}
```

Comando





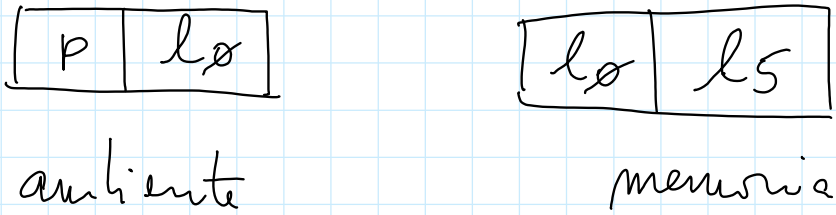
PUNTATORI

lunedì 24 settembre 2018 09:49

nello stato
- nomi
- locazioni
- valori

Il contenuto di una locazione di memoria è interpretato come un valore.

Il contenuto di una locazione di memoria può anche interpretato come una locazione



il valore della variabile p (valore alla locazione l0) in realtà è una locazione di memoria

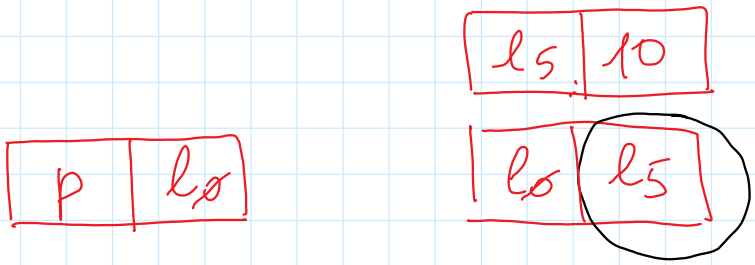
p si chiama VARIABILE PUNTATORE

lunedì 24 settembre 2018 10:16

Una variabile puntatore si dichiara:

```
int * p;
```

↑ il valore in memoria per p va interpretato come l'indirizzo di una parola di memoria che contiene un valore intero.



esempi:

p ha valore → l5

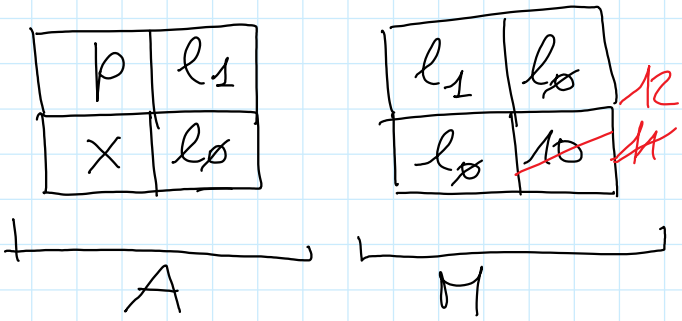
&p → l'indirizzo di memoria per p | l0

*p → il valore contenuto nella locazione che è valore di p | 10

lunedì 24 settembre 2018 10:22

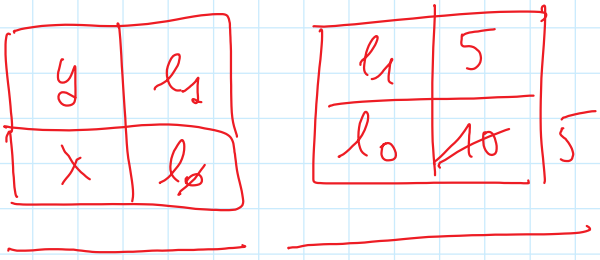
```
{ int x = 10;
  int *p = &x;
  x = x + 1;
  *p = *p + 1;
  .
  .
  .
```

attraverso il puntatore p abbiamo modificato la variabile x



Lo scambio del valore di due variabili.

```
{ int x = 10;
  int y = 5;
  x = y;
  y = x;
```



```
{
int x = 10;
int y = 5;
int t = x;
x = y;
y = t;
}
```

t	l_2
y	l_1
x	l_0

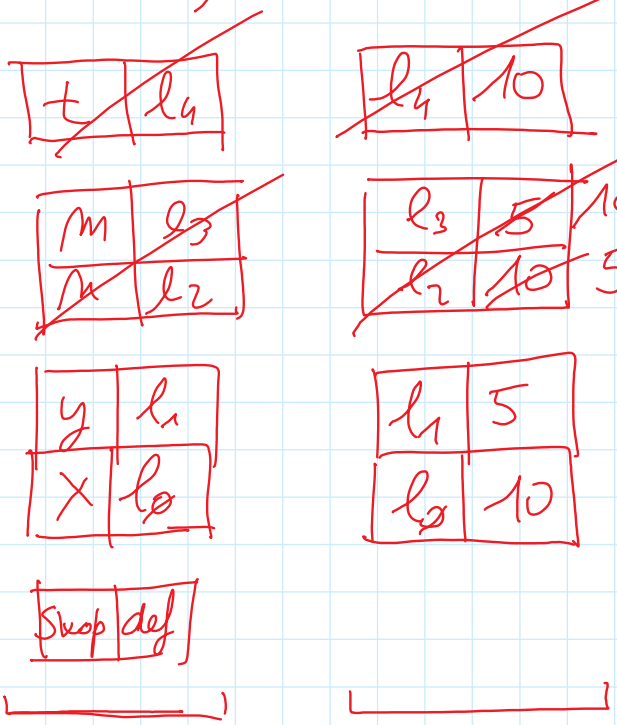
l_2	10
l_1	5
l_0	10

10

5

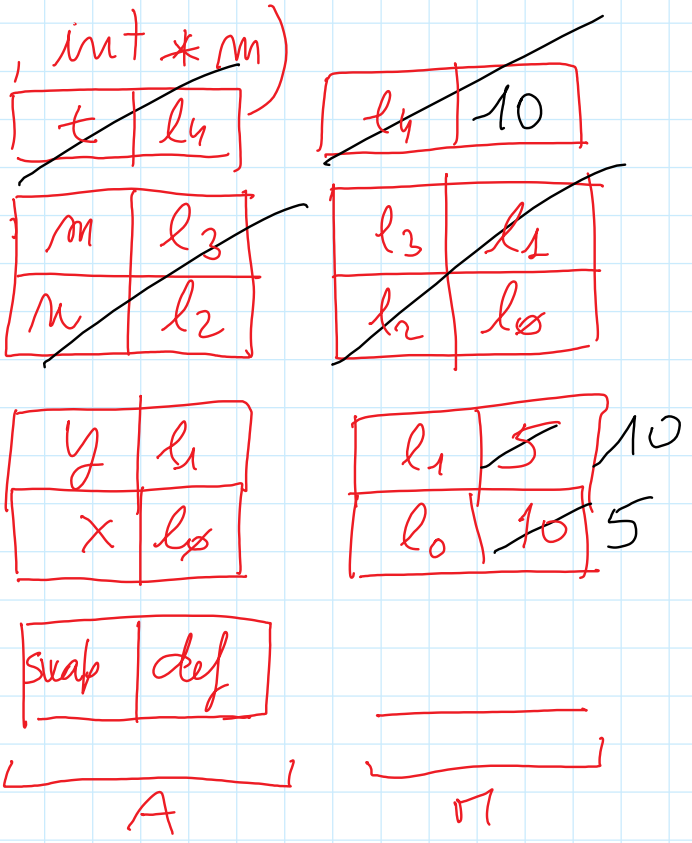
```
void swap (int n, int m)
{ int t = n;
  n = m;
  m = t;
}

main()
{ int x = 10;
  int y = 5;
  swap (x, y);
```



lunedì 24 settembre 2018 10:34

```
void swap(int *n, int *m)
{ int t = *n;
  *n = *m;
  *m = t;
}
main()
{ int x=10;
  int y = 5;
  swap(&x, &y);
```



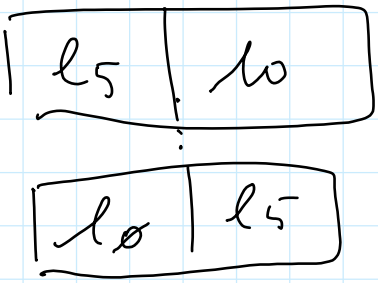
Puntatori: variabili il cui valore è un indirizzo.

Espressioni:

$\&p = l_0$

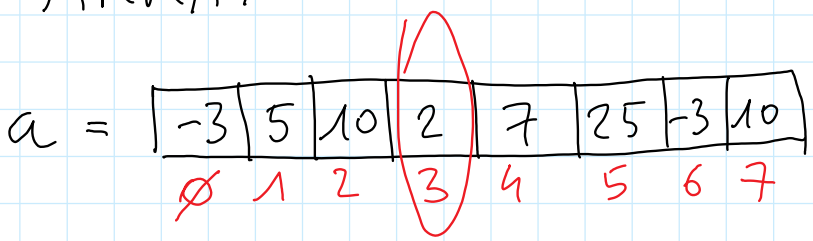
$p = l_5$

$*p = 10$



lunedì 24 settembre 2018 10:46

ARRAY




```
int a [8]
```

- `int`: tipo degli elementi
- `a`: nome array
- `8`: numero degli elementi

`a[3]`: l'elemento di a di indice 3

~~`int a[x];`~~

```
a[3] = 5;
a[4] = a[3] + 1;
```

sia alla sinistra che alla destra di un assegnamento

modifica con il singolo elemento

```
main()
{ int a[10];
  int i;
  for (i=0; i<10; i++)
      a[i] = 0;
  ⋮
```

---

leggere i valori di un array dall'esterno.

```
printf( ... );
```

```
scanf( , );
```

scanf("%d", &x);

Tipo del valore da leggere

Variabile da modificare con il valore letto

```
main()
{ int x;
  scanf("%d", &x);
```

scanf e printf sono procedure predefinite nella libreria stdio

standard input/output

lunedì 24 settembre 2018 10:58

```
main()
{
    int a[10];
    int i;
    for(i=0; i<10; i++)
        scanf("%d", &a[i]);
```