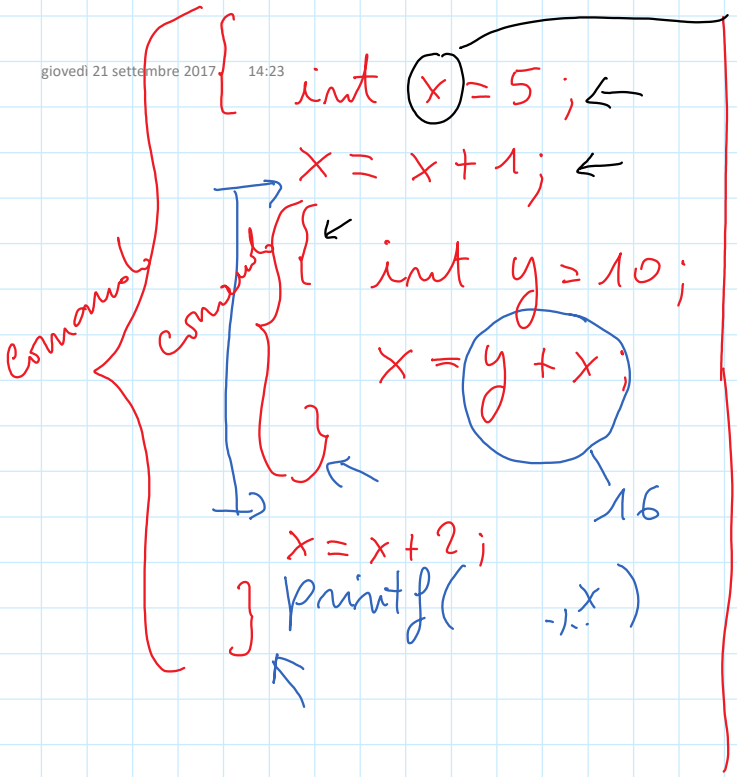
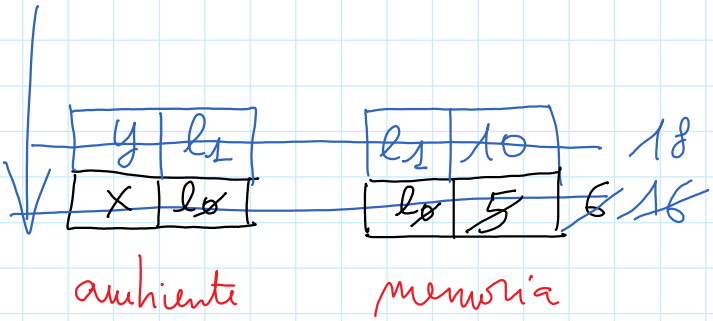
giovedì 21 settembre 2017 14:23

nomi, variabili, identificatori

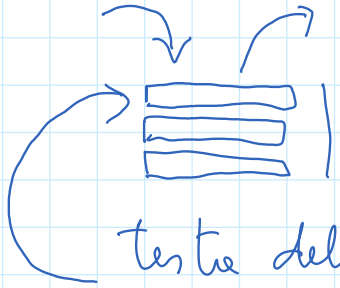




ambiente memoria

PILA (STACK)

giovedì 21 settembre 2017 14:23



LAST IN FIRST OUT

LIFO

testa della pila

push (p, v)

pop (p)

Costante particolare:

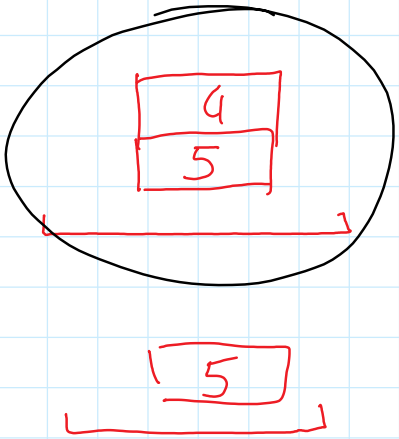
pila vuota

Ω

pile di interi $\Omega =$ ____

giovedì 21 settembre 2017 14:23

pop (push (push (Ω, 5), 4))



y	l_1
x	l_0

ambiente

frame di ambiente

l_1	5
l_0	3

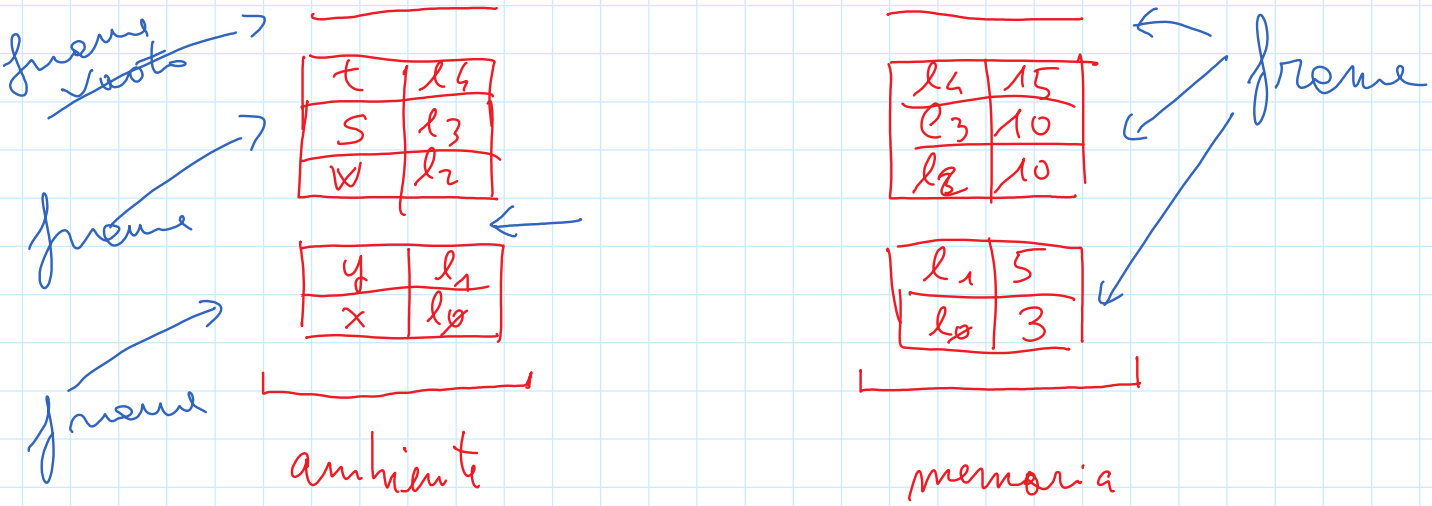
memoria

frame di memoria

Frame particolari:
frame vuoto
——→ frame

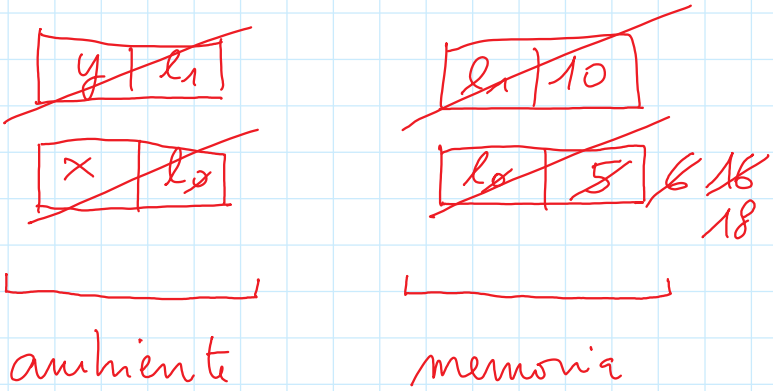
ambiente → pile di frame di ambiente
memoria → pile di frame di memoria

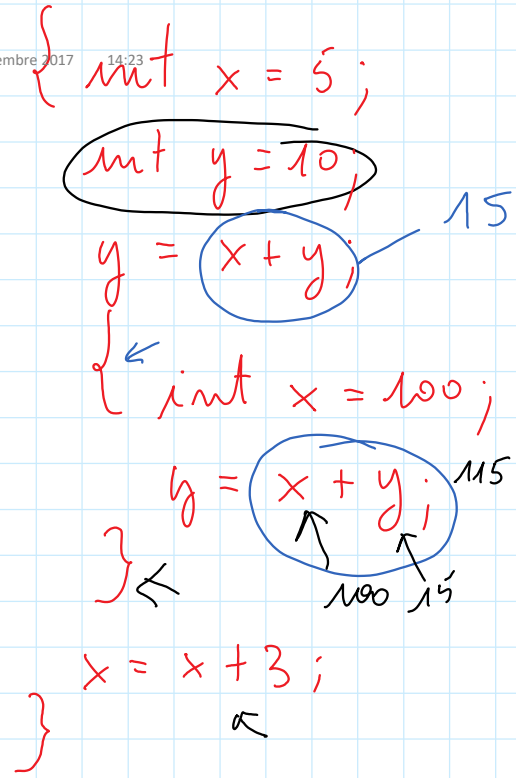
es. di possibili ambiente e memoria

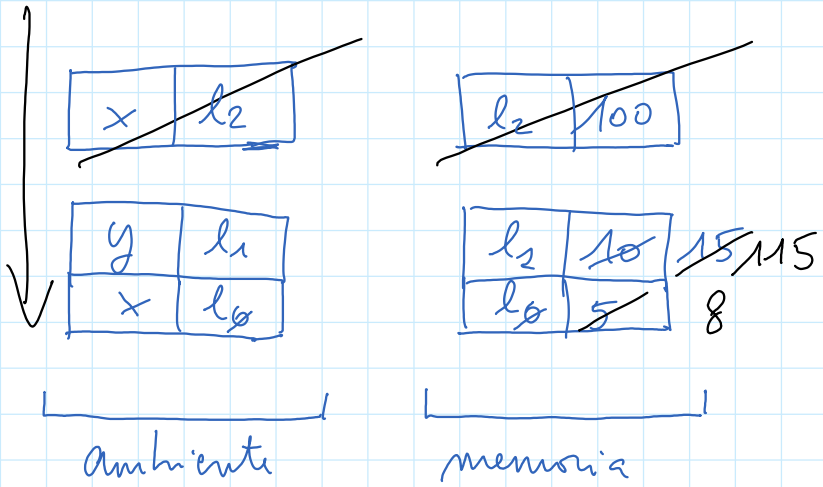


- Quando si esegue un blocco si aggiunge, sia all'ambiente che alla memoria, un frame vuoto in testa.
- La semantica (l'esecuzione) di una dichiarazione aggiunge una associazione, sia nell'ambiente che nella memoria, nel frame in testa
- Quando termina un blocco si esegue un'operazione di POP su entrambe le pile (ambiente e memoria)

```
{
    int x = 5;
    x = x + 1;
    {
        int y = 10;
        x = y + x;
    }
    x = x + 2;
}
```



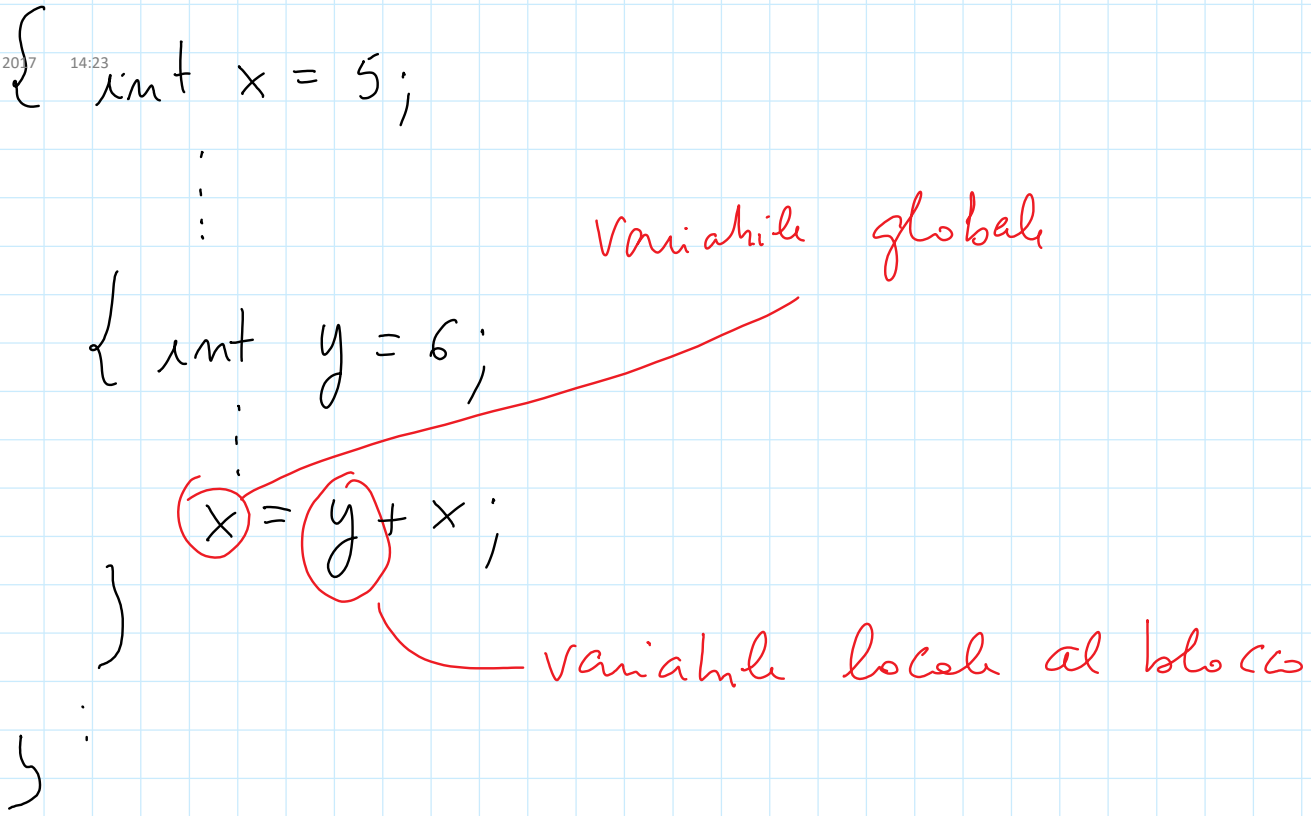

{ int x = 5;
int y = 10;
y = x + y;
15
{ int x = 100;
y = x + y;
115
100
15
}
x = x + 3;
}


x
l_2
y
l_1
x
l_6
l_2
100
l_2
10
15
115
l_6
5
8
ambiente
memoria

Regole di visibilità degli identificatori

giovedì 21 settembre 2017 14:23

Gli identificatori dichiarati in un blocco sono visibili in tutti i blocchi interni, a meno che non ne venga "coperto" da un identificatore, con lo stesso nome, dichiarato internamente (nei blocchi più interni)


int x = 5;
int y = 6;
x = y + x;
variabile globale
variabile locale al blocco

FUNZIONI E PROCEDURE

giovedì 21 settembre 2017 14:23

```
int n = 18;
int m = 15;
int x = 95;
int y = 42;

main()
{
    while (n != m)
        if (n > m) n = n - m;
        else       m = m - n;
}

while (x != y)
    if (x > y) x = x - y;
    else       y = y - x;
```

```
void mcd (int p, int s)
{ while (p != s)
    (p > s)   p = p - s;
    else      s = s - p;
}
```

PROCEDURE

```
mcd (m, m);
mcd (x, y);
```

$f(x,y) = x+y+2$ ← def

nome — parametri — valore in funzione dei parametri

$f(2,3) = 7$ — chiamata di funzione / applicazione di f

$f(3,4) = 9$

Calcolo del fattoriale

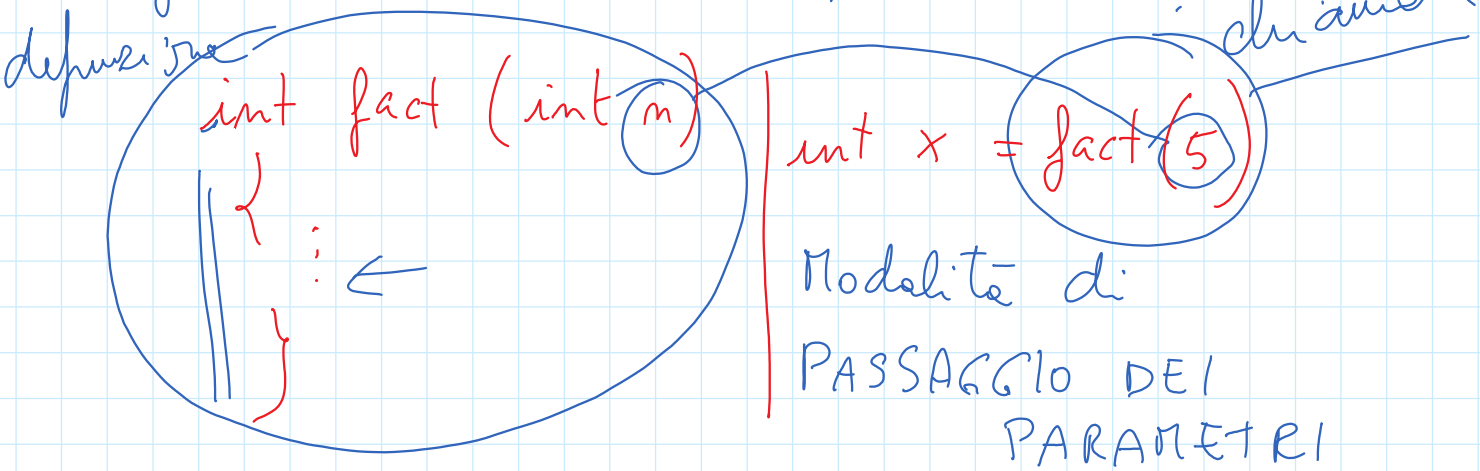
giovedì 21 settembre 2017 14:23

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

FUNZIONI

giovedì 21 settembre 2017 14:23

Come si collegano, attraverso lo stato, gli argomenti di una procedura o funzione, con i parametri?

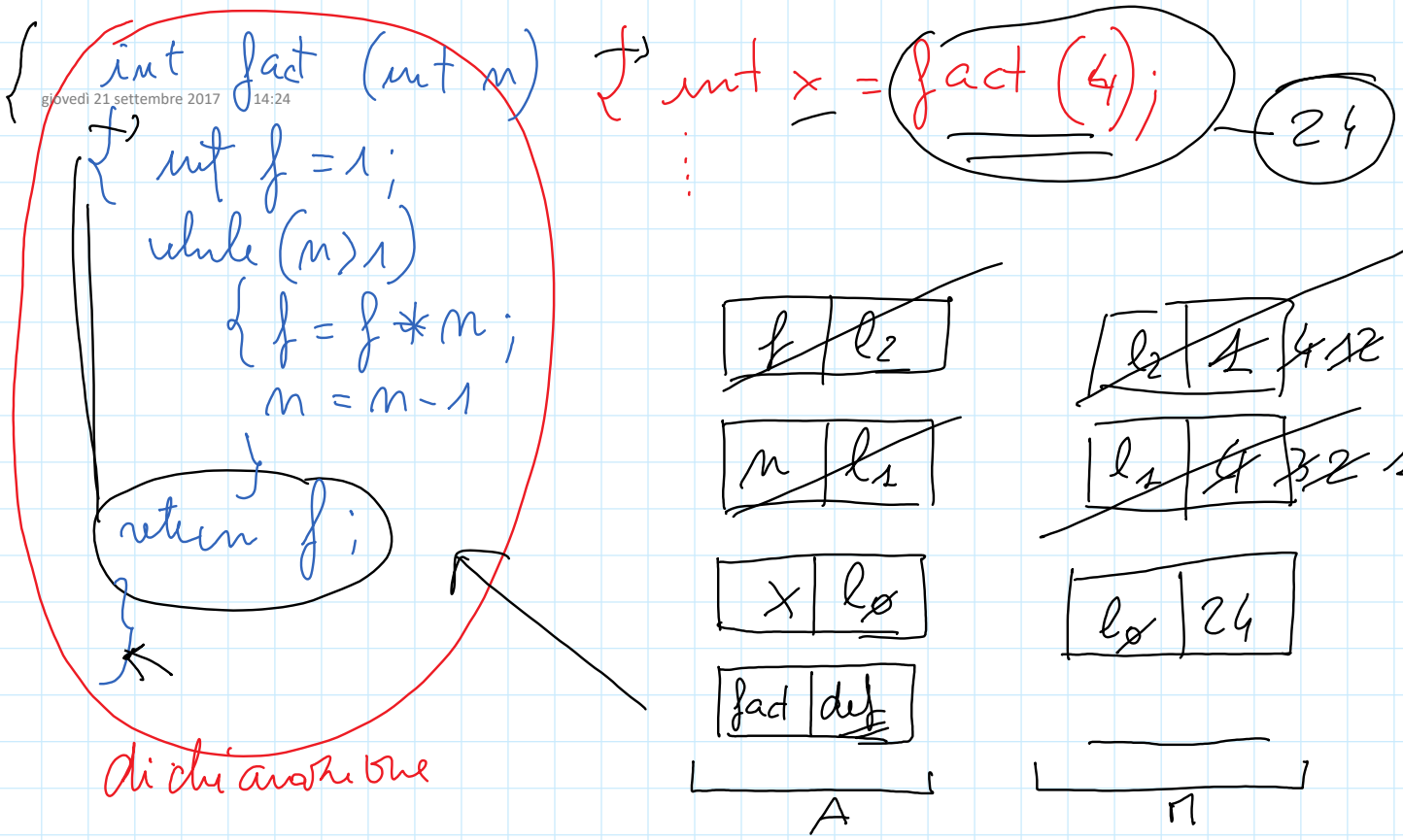


Modalità di PASSAGGIO DEI PARAMETRI

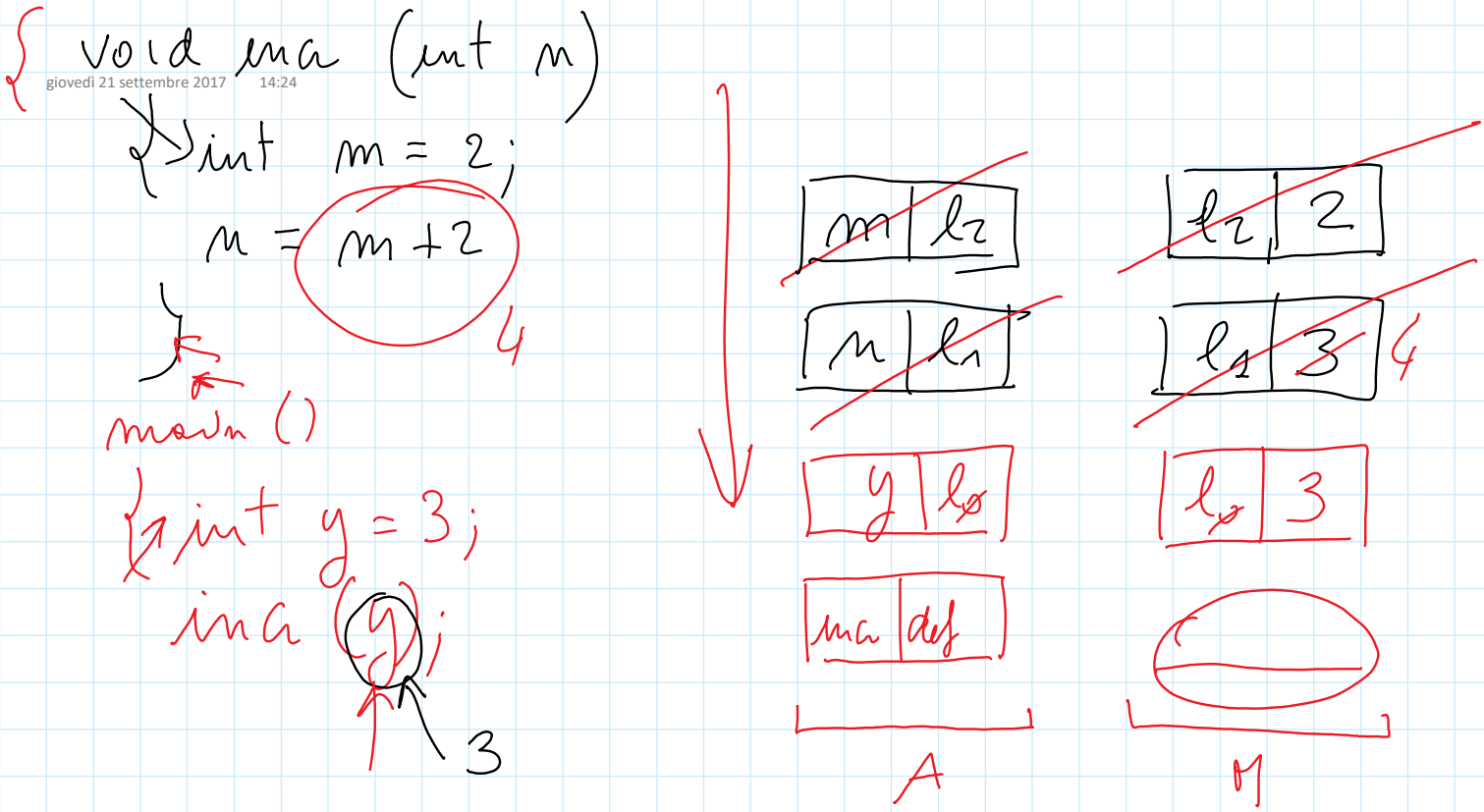
giovedì 21 settembre 2017 14:23

Il passaggio dei parametri in C
è fatto con la modalità "PER VALORE"

Quando una procedura o funzione viene
"chiamata" viene aggiunto, sia all'ambiente
che alla memoria, un frame con le
variabili corrispondenti ai parametri.
A queste variabili viene dato inizialmente il
VALORE degli argomenti.


int fact (int n)
{ int f = 1;
while (n > 1)
{ f = f * n;
n = n - 1
}
return f;
}
int x = fact (4);
24
dichiarazione
f | l2
n | l1
x | l0
fact | def
A
l2 | 1 | 4 12 24
l1 | 4 3 2 1
l0 | 24
n

giovedì 21 settembre 2017 14:24



giovedì 21 settembre 2017 14:24

Blocco

è un comando { Sequenza di dichiarazioni
Sequenza di comandi }

comandi modificano lo stato ma non creano nuove associazioni nello stato (ambiente + memoria)