

variabili, espressioni e conversioni

testo 2.2 e 2.4

Tutti i Linguaggi di Programmazione
usano **variabili**

le variabili sono nomi con un valore che
cambia durante l'esecuzione del
programma

**ogni variabile deve essere dichiarata
con un tipo e può essere inizializzata
con un valore del tipo appropriato**

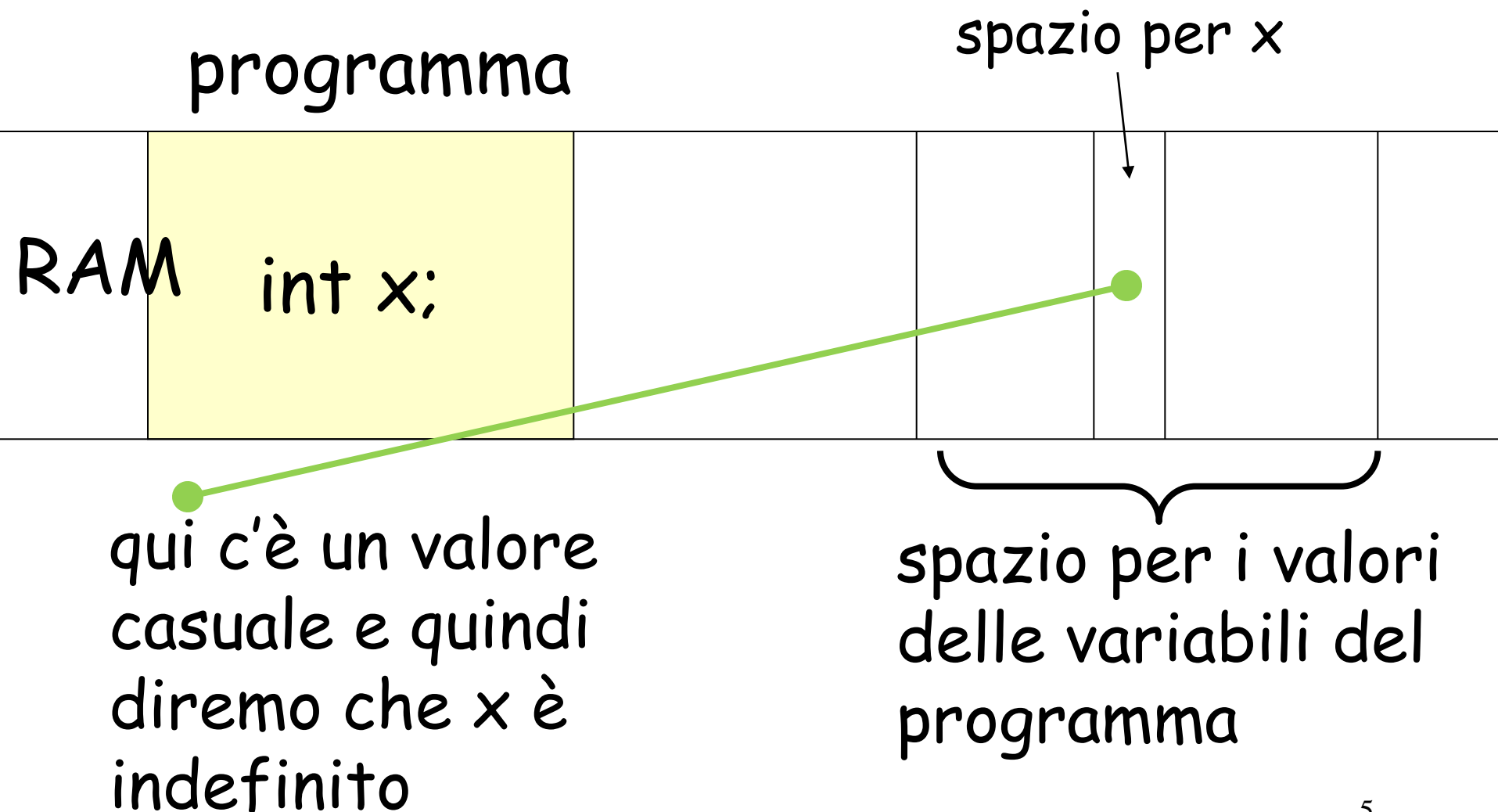
ogni variabile deve essere dichiarata prima dell'uso con un tipo

- `int pippo;` // senza inizializzazione
 - `int pippo=127;`
 - `char pluto='a';`
 - `float y=3.14f;`
 - `bool ok=true;`
- } con inizializzazione

in C++ le variabili :

- iniziano con un carattere alfabetico (minu. o maiu.) e il resto può contenere ancora caratteri alfabetici, numeri, e '_' (sottolineatura)
- la variabile x è diversa dalla variabile X
- le variabili devono essere distinte dalle parole chiave del C++

quando viene eseguito un programma:



usare in
qualsiasi modo
un valore
indefinito è un
errore

assegnamo a x il valore 20

programma

spazio per x



20 è l' **R-valore** di x mentre questo indirizzo RAM è il suo **L-valore**

ogni variabile ha:

- un **R-valore** che può essere definito se esso viene assegnato dal programma o altrimenti è indefinito
- un **L-valore** che è l'indirizzo RAM in cui si trova il suo R-valore

COSTANTI

```
const int pippo=10;  
const float pi_greco= 3.14f;
```

- vanno inizializzate **SEMPRE** al momento della dichiarazione
- non esistono: il compilatore le sostituisce immediatamente col valore di inizializzazione

espressioni

- nei programmi usiamo espressioni come $a + 12 * bib$
- un'espressione va valutata per ottenere un VALORE, ma deve essere sensata
- a , bib e 12 devono "andare d'accordo"

controllo di sensatezza di un'espressione

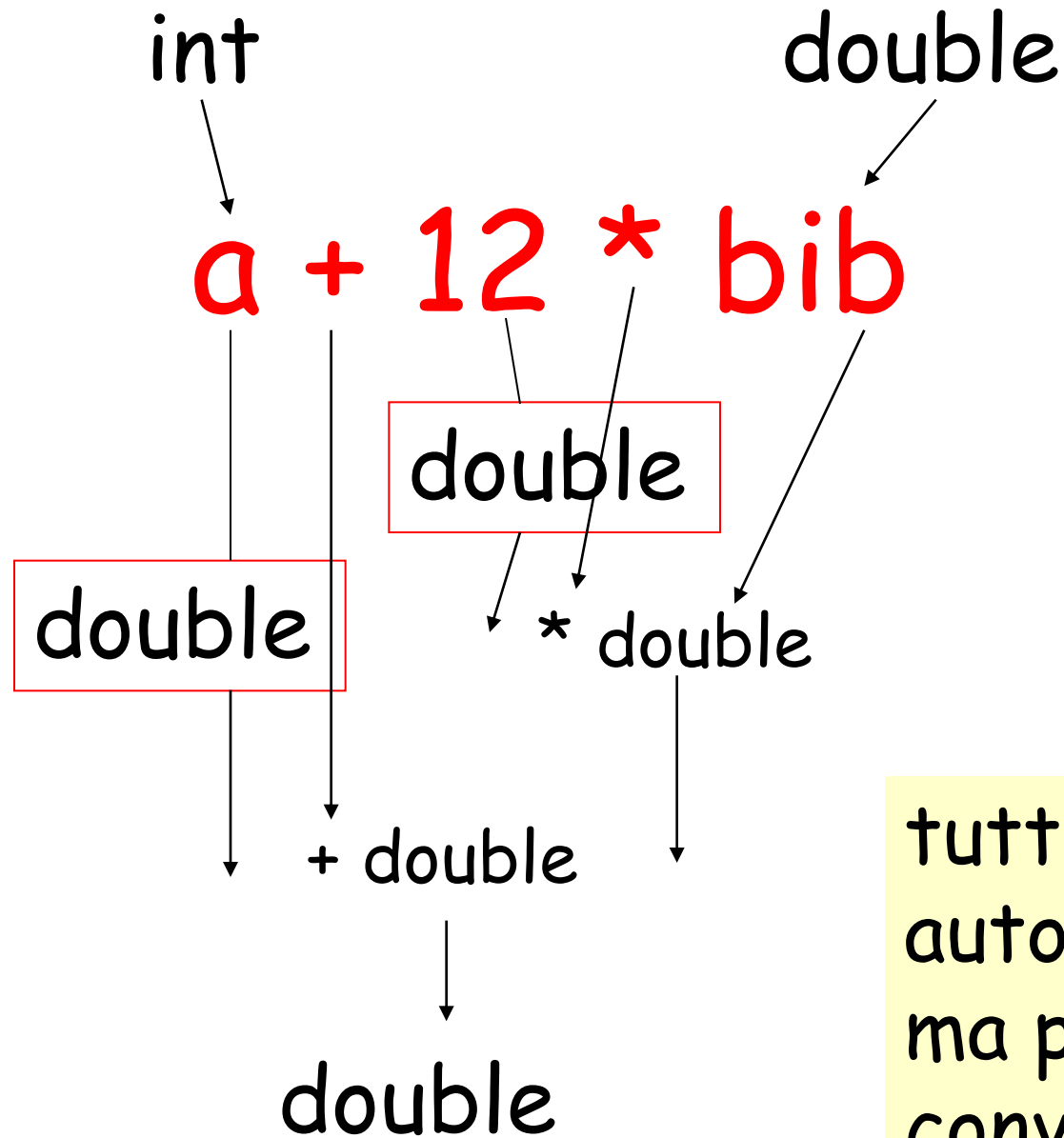
il compilatore deve stabilire se un'espressione come $a + 12 * bib$ è SENSATA o no

COME FA ??? usa i tipi di a e di bib e quello di 12 (int)

consideriamo alcuni casi possibili

$a + 12 * bib$

- a è int e bib è di tipo stringa : il compilatore dà errore di tipo
- a e bib sono entrambi int: è sensata e si usano $+$ e $*$ interi
- a è int e bib è double: il compilatore trasforma 12 in double, usa $*$ per i double per $(12*bib)$, poi trasforma l'R-valore di a in double e applica la $+$ dei double per $(a+...)$



tutto avviene automaticamente, ma perché queste conversioni ??

consideriamo $12 * \text{bib}$ con bib: double

1) uno dei 2 operandi deve assumere il tipo dell'altro: esiste solo $*$ tra interi o tra double o tra float, ma non tra un int e un double !!

2) quindi si potrebbe trasformare

R-valore di $\text{bib} \rightarrow \text{int}$ oppure

$12 \rightarrow \text{double}$

e si sceglie questa seconda conversione

PERCHE' ??

principio: si operano automaticamente conversioni che non fanno perdere informazioni

int (4 byte) → double (8 byte)

char → int

bool → int

int → float (!!)

se l'espressione è sensata allora il compilatore produce codice che eseguito calcola il valore dell'espressione
 $a + 12 * bib$ con $a: \text{int}$ e $bib: \text{double}$ in

```
LOAD bib R0;  
LOAD 12 R1  
CONV_INT_DOUBLE R1  
MULT_DOUBLE R0 R1
```

.....

esempio:

2345 + 'a'

converte 'a' in int

si passa da 1 a 4 byte : è sicuro

il viceversa da 4 a 1 byte,

porterebbe $2345 \% 256 = 41 = ')$

con evidente perdita di informazioni

$(2345 + 'a') * 23.56$

prima 'a' si converte in int

$2345 + 97 = 2442$

poi 2442 si converte in double

double usa 8 byte, int solo 4 : OK

risultato è di tipo double

MA che vuol dire convertire ? **Dipende**

char → int

'a' =

97

 intero 97 in un byte

viene convertito nel valore intero:



che occupa quattro byte

la cosa si complica se il codice ASCII del carattere è un intero negativo:

per esempio 'è' = -24

come char è rappresentato dal byte

232

che in complemento a 2 rappresenta -24

convertirlo a int significa trasformarlo in -24
in complemento a 2 con 4 byte:

xxx

xxx

xxx

xxx

il valore contenuto nei 4 byte è = $4294927272 = 2^{32} - 24$

convertire int \rightarrow float

int = 102101 \rightarrow segno = 1 bit, e = 8, m = 23 bit

segno=0

e = 132 (-127)

m = 02101 (1. è implicito)

$2^5 * 1.02101$

int \rightarrow double simile con il doppio dei bit