Lezione 3

tipi predefiniiti

un tipo ha:

1. nome
2. insieme di valori di quel tipo che dipende dal numero di byte usati per rappresentare i valori
3. insieme di operazioni definite su quel tipo

la memoria è costituita da una sequenza di byte con indirizzi crescenti

il byte è la minima quantità di memoria che si può accedere

Tipi predefiniti

int usa 4 byte 2^{32} valori

char usa 1 byte 2

bool usa 1 byte $2^{8} = 256$ valori

float e double 4 e 8 byte

void 0 valori

ci sono anche short int, long int

int s= sizeof(tipo)

- int occupa 4 byte e si usa la rappresentazione in complemento a 2, ci sono 2^{32} valori
- bool occupa 1 byte, 2 valori, true e false

- char occupa 1 byte
  256 caratteri
  contiene i caratteri ASCII = 128 caratteri
  + altri 128

- float (4 byte) e double (8 byte)
  usano codifica floating point

[+/-, mantissa, esponente]

rappresenta valore

+/- 1 . mantissa * ($2^{\text{esponente}}$)

attenzione
rappresentazione degli interi (complemento a 2) e dei reali (floating point)
sono completamente diverse

ma alla fine il valore è sempre costituito da alcuni byte con dei bit 0 e 1

devo sapere il tipo per interpretare la sequenza di bit nel modo giusto

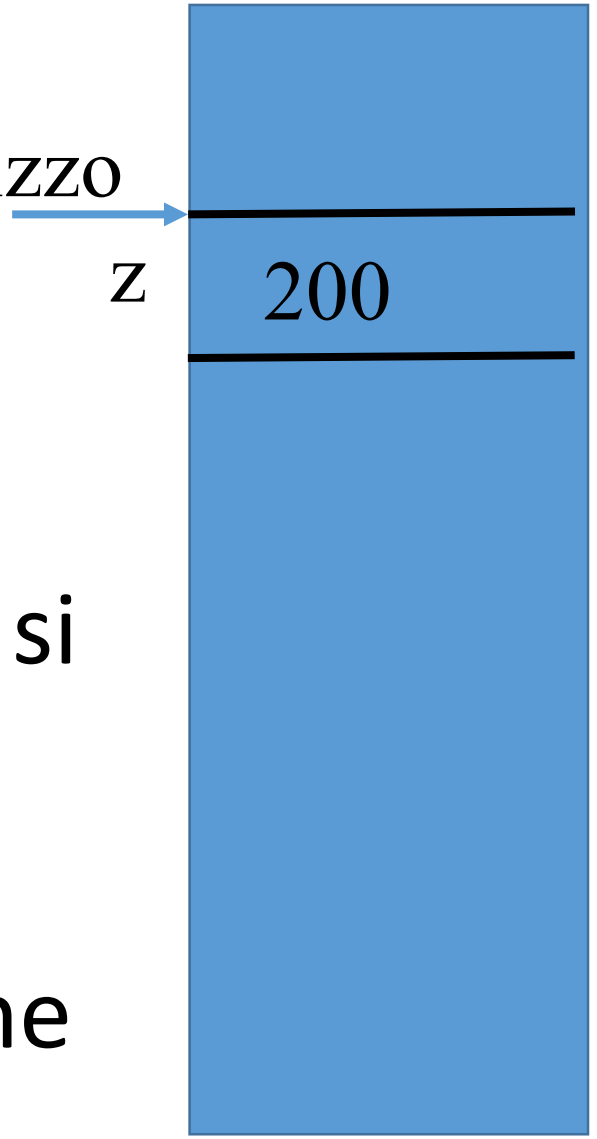
ogni variabile deve venire dichiarata
prima di essere usata:

```
int x; // indefinita attenzione
double y; // indefinita attenzione
float  z= 10; // con inizializzazione, ma ??
```

conversione automatica 10 è trasformata
in floating point in memoria

ogni variabile ha R- ed L-valore

per esempio: int z=200;



R-valore di z è 200

L-valore è l'indirizzo RAM in cui si trova questo valore

&z è un'espressione che ha come valore l'L-valore di z

espressioni più complesse che mescolano tipi diversi

12 - x * pippo + y

si vuole sempre calcolare il suo valore, cioè eseguire le operazioni

- c'è una costante intera 12 e tre variabili
- devono essere tutte intere?
- in che ordine si eseguono le operazioni?
- quale +, - e * si eseguono?

l’ordine dipende dalla precedenza degli operatori e dall’associatività:

12 - pippo * x + y
(12 - (pippo*x)) + y

* ha precedenza su + e – e tutte associano a sinistra

per le altre domande si deve capire che :

- le operazioni +, *, -, /, ecc. lavorano solo con operandi dello stesso tipo

overloading delle operazioni: c’è una + su int, una diversa + su float, un’altra su double ….

e si deve anche capire il …

…..principio di base delle conversioni:

se si deve decidere se convertire un valore di tipo T1 nel tipo T2 o vicersa, si converte sempre dal tipo che usa meno byte al tipo che usa più byte

non si perdono informazioni (o quasi)

queste conversioni sono dette promozioni
char e bool -> int
int -> float (!)
float–> double

ci sono vari casi:

1. se x, y e pippo sono tutte int allora le operazioni sono +, - e * tra int

2. e se pippo fosse char, x float e y double ?

(12 - (pippo*x)) + y

char -> float pippo convertito in float

int -> float 12 convertito in float

float -> double

risultato finale è double

anche l’assegnazione può richiedere una conversione, ma senza scelta

```
int w = 12 - pippo * x + y
```

con le ipotesi sulle variabili del caso (2), l’espressione a destra dell’= ha un valore di tipo double ed esso viene convertito in int troncando i decimali