

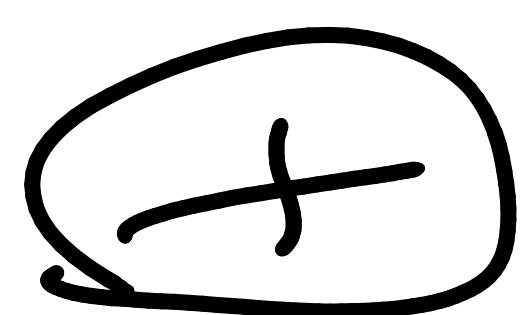
load
store

add mul
neg div
sub

if * ~~SACI~~ CONDITIONS,

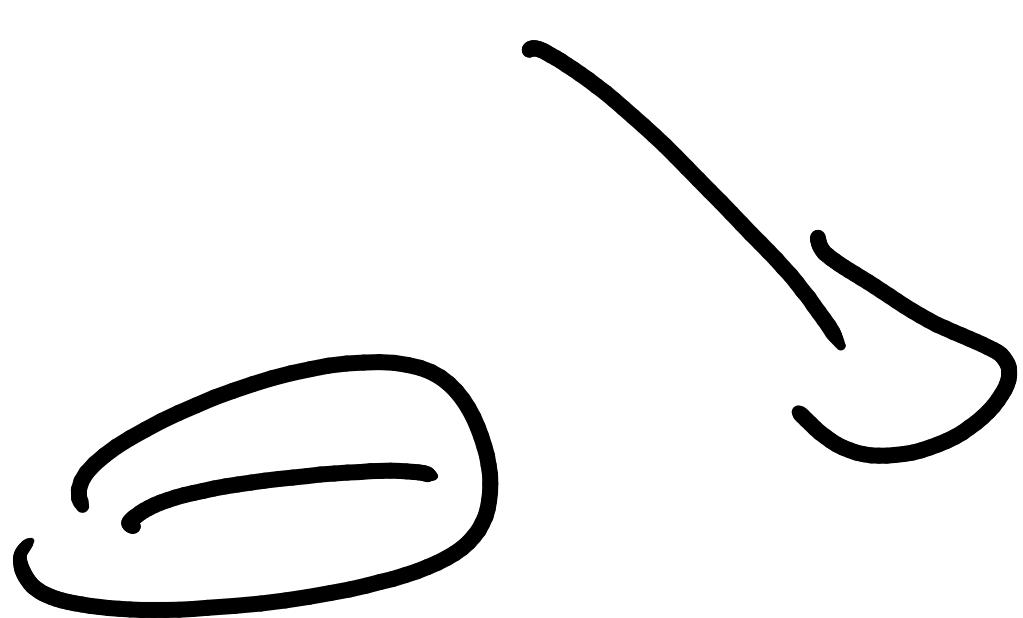
goto

invoke ~~*~~ METHOD CALL



→ PORTABILITY

(SUCH SPECIFIC H(X))



PERFORMANCE

Class

→ Field (attribute)

data

HIGHLY
COUPLED

Method (function)

functionality

Card game

Card (Briscada)

↳ senso: Denari, Bastoni, Spade, Cuori

↳ valore: 1-10

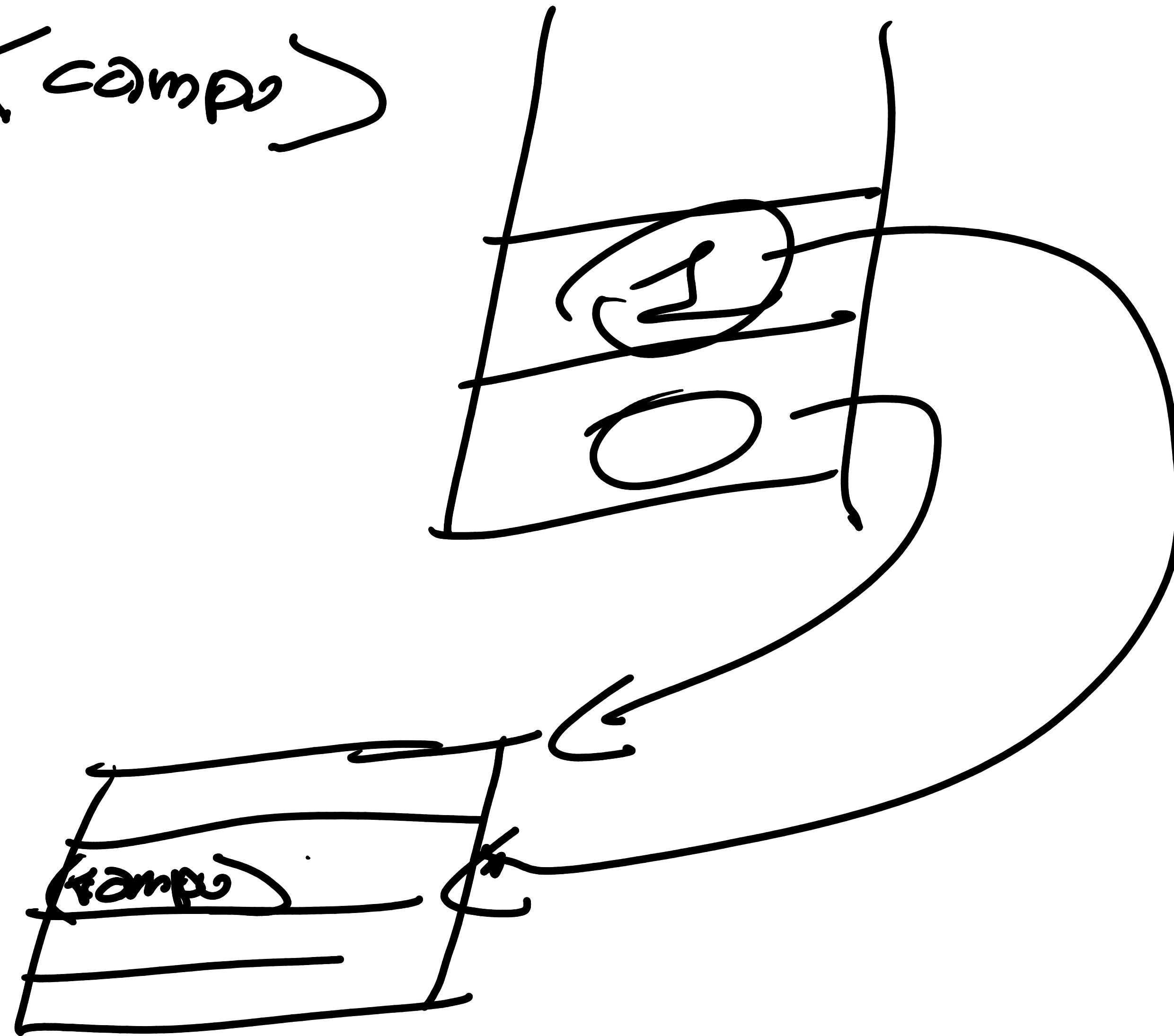
here C



heap

but fields

(campo)



Cord: seme, valore

new

(denari; forte)

new

(forte) asso

new

(denari; 3)

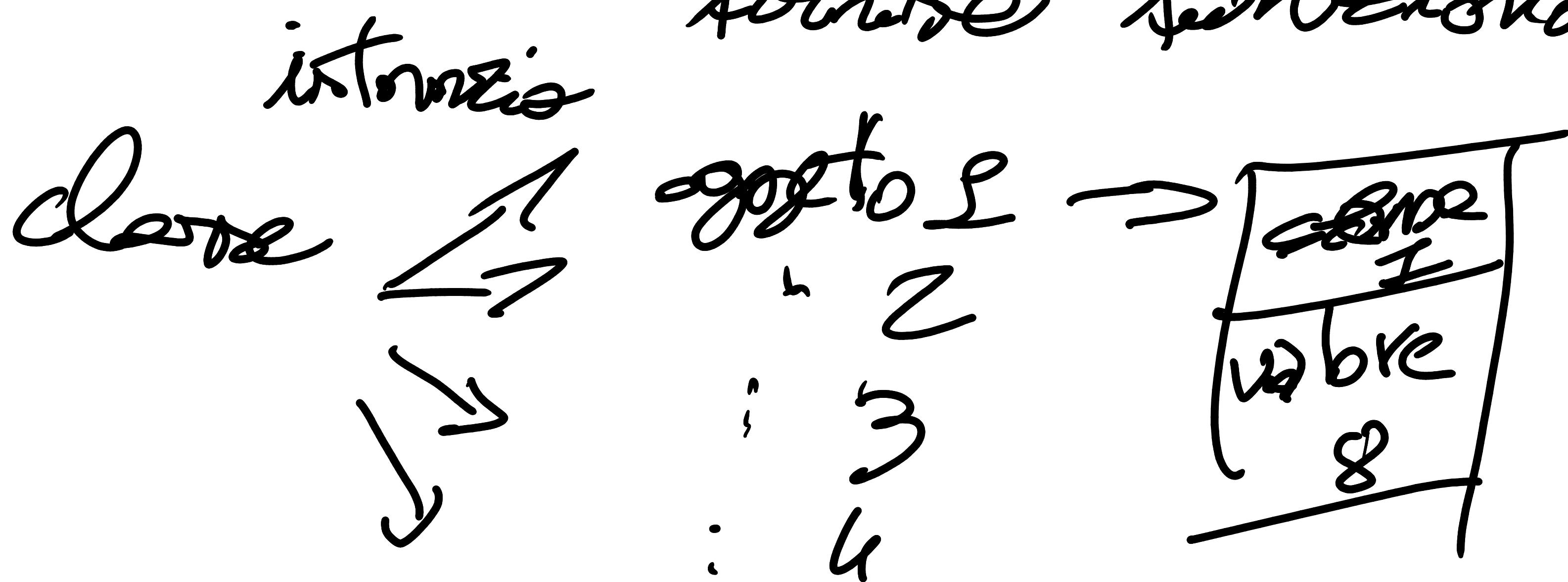
dear

oggetto

S1	C2	C3	C4
0	1	2	3
→ 18	→ -14	42	

Classe: definisce dati e funzionalità

Oggetto: istanza di una classe
contiene dati - Consiste
in varie funzionalità



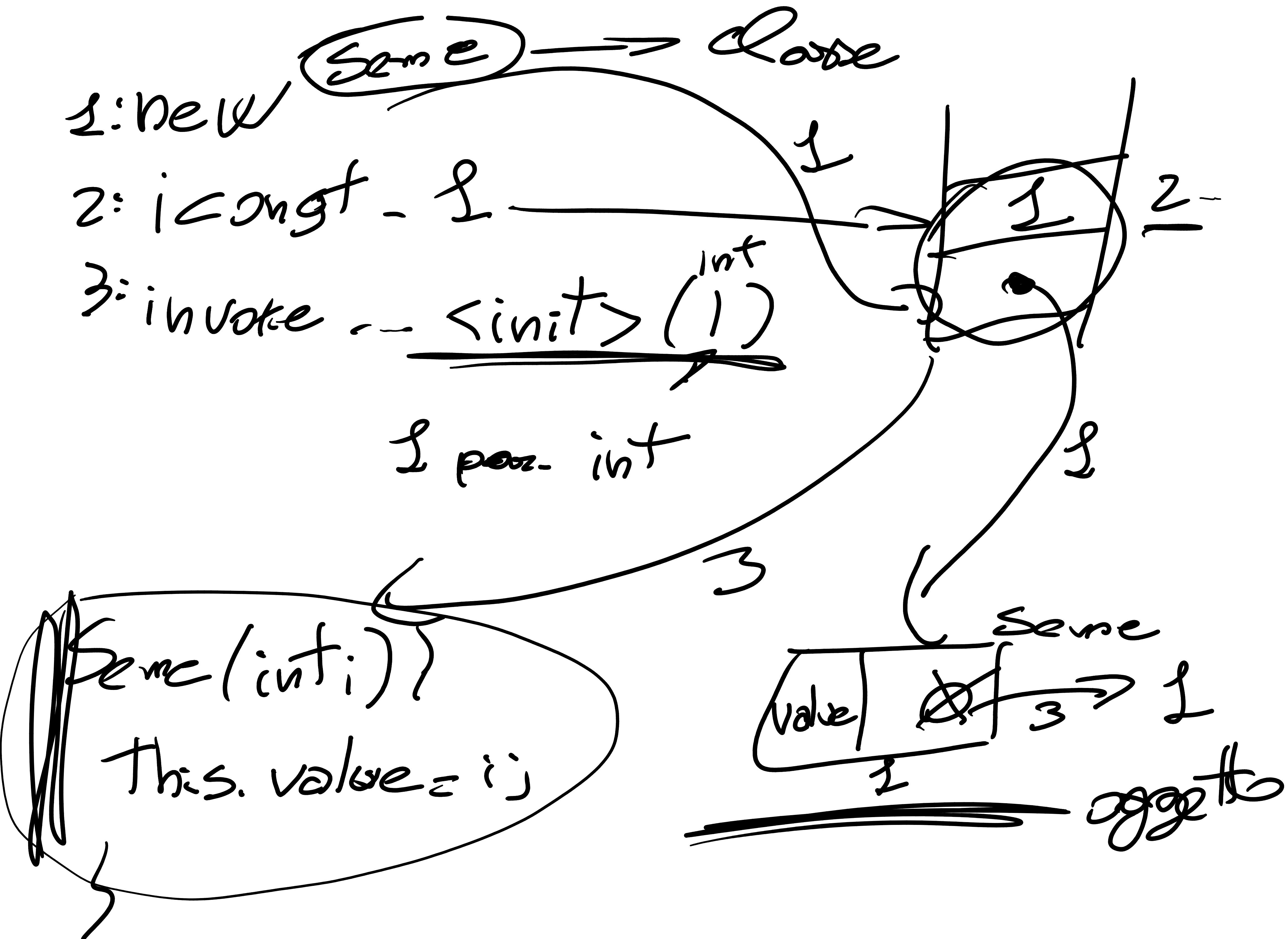
```
class Card {
    int valore;
    void stampa() {
        System.out.println("Carta di colore " + colore);
        System.out.println("con valore " + valore);
    }
}

class Main {
    public static void main(String[] args) {
        Card c1 = new Card();
        c1.colore = "rosa";
        c1.valore = 10;
        c1.stamp();
        Card c2 = new Card();
        c2.colore = "verde";
        c2.valore = 5;
        c2.stamp();
        if (c1 > c2)
            System.out.println("Carta 1 è maggiore");
        else
            System.out.println("Carta 2 è maggiore");
    }
}
```

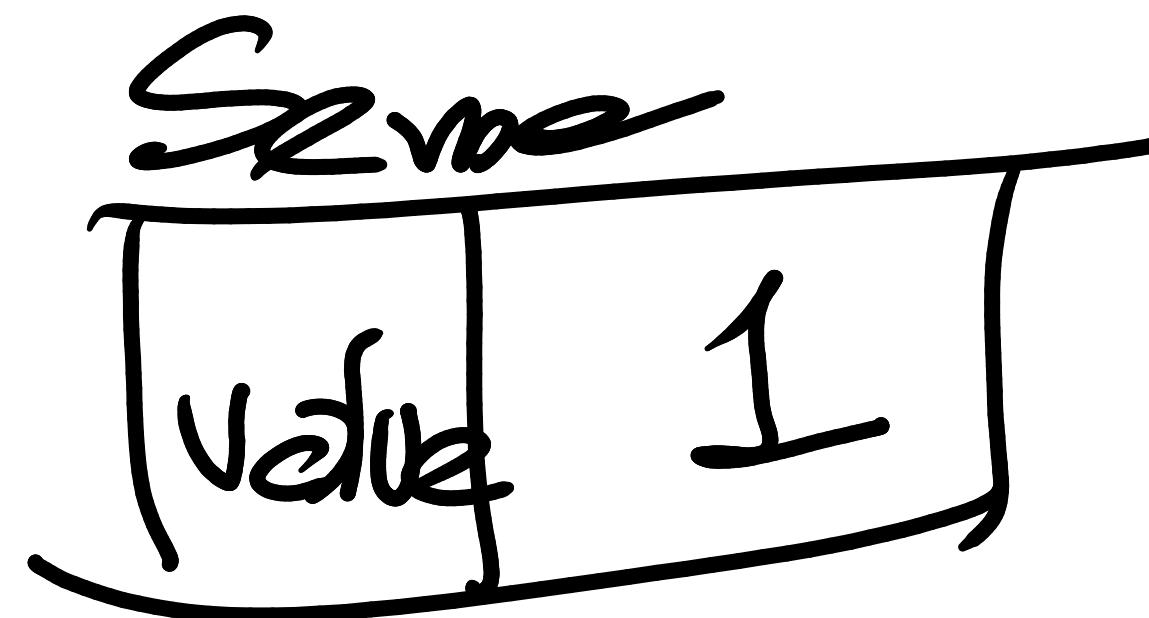
CEZIONE 4

Classe → campi dati
→ metodi funzionalità

new → oggetti



heap



CLOS

+100 visit
camera 100%
no return value

Spring -> vom e
regional
Value; -> name

camera ~~of~~ sense [int'ə] /  context
of

```
void print()
```

J met o dř

this

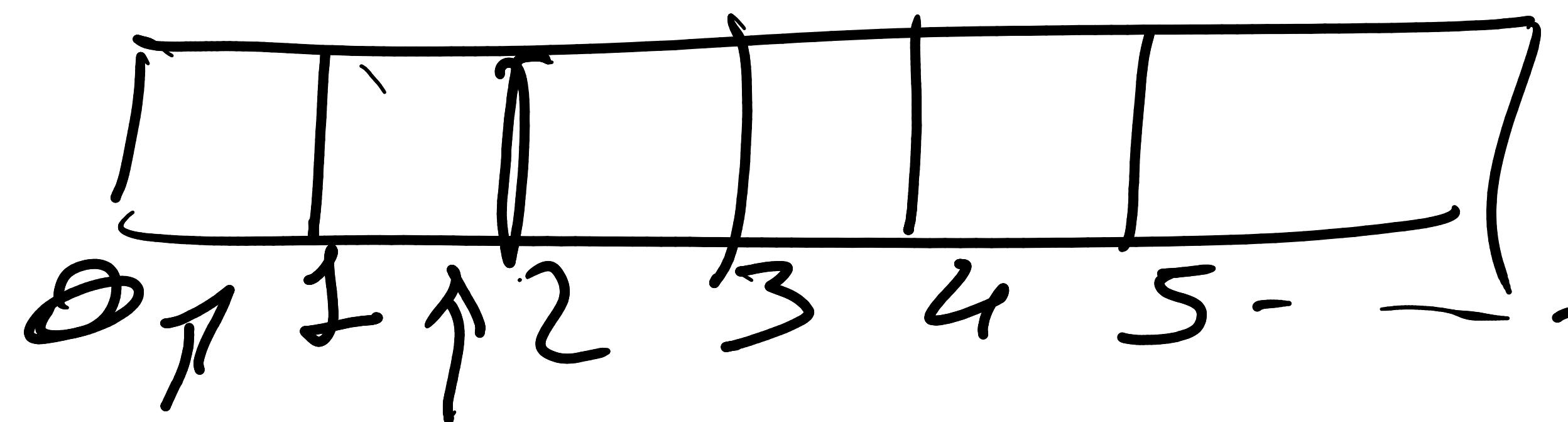
Presentato all'oggi
oggetto corrente

Seme · print()
↓
receive nasce
 metodo testo dei
 parametri

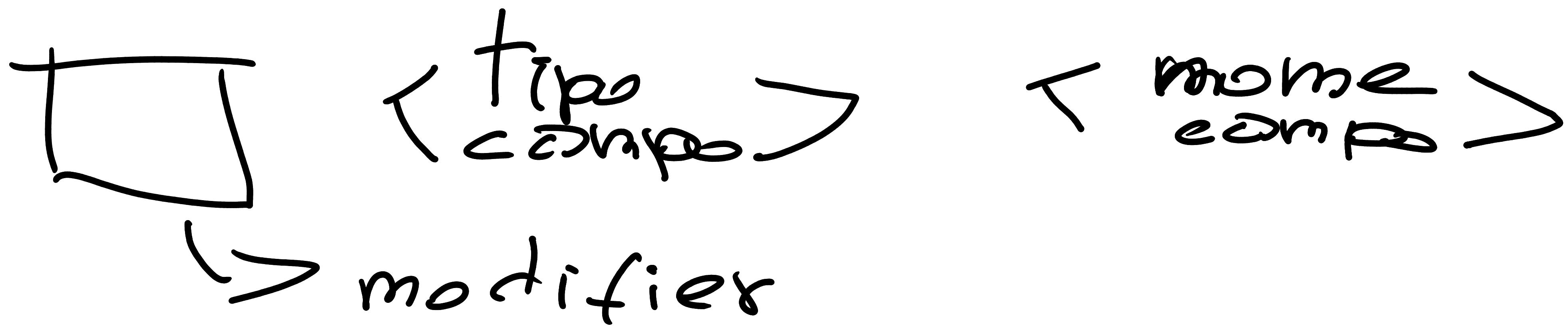
↳ Timbro il metodo se un
oggetto

↳ this dentro al metodo

Same (int i)



this i



Modifier

[public
 ↳ void
protected
private]

Access

↳ Encapsulation

[synchronized
volatile]

Concistency

[static—
final
abstract]

Others

finali compi

- ① deve essere iniziato dal costatoe una ed una sola volta
- ② tetti gli altri mette non possono assegnarli

static metodi

① metodi senza "this"

② non c'è un oggetto corrente
a cui accedere

③ "appartengono" alla classe

④ si indicano con

<nome classe>.<nome
metodo>(- - -)

LEZIONE

5

Riassunto

Classe

Campi

Metodi

Modifieri

~~Concurrenza
Volatile
Synchronized~~

access)

static
final

campi final \approx const

metodi static

costruttore static

Class loader

- ① classpath : dove trovo le classi
- ② Quando una classe viene usata per la prima volta
 - a) la cerca
 - b) non trovata: lancia
 - c) trovata:
 - esegue costruttore
 - stessa

```
class Tester { → de classe
static void main (String[] args) {
    System.out.println ("Hello")
}
```

Y ↓ ↓ ↗
classe campo
statico de System methods
} statico non - statico

$\langle \text{mod} \rangle \text{ class } \langle \text{name Class} \rangle \langle$
 $(\langle \text{mod} \rangle \langle \text{tip} \rangle \langle \text{name} \rangle ;)^*_{\text{C\&HPI}}$
static } --
 $(\langle \text{name Class} \rangle (\sim) \{ \rightarrow \})^*_{\text{CSR.}}$
 $(\langle \text{mod} \rangle \langle \text{tip ret} \rangle \langle \text{name} \rangle (\rightarrow) \{ \rightarrow \})^*$
{}

C

char * ptr = malloc(10)

ptr → * = 10

* (ptr + 1) = 'o'

free (ptr)

Java

Card c = new Card();

c.value = 10;

NO

NO

GARBAGE COLLECTION

enum :

- classe
- I campi statici final per ogni possibile valore
- Posso aggiungere campi e metodi di parametri nel costruttore, dopo decidere il valore
Oraando definisco il caso