

## I metodi - III

Stefano Mizzaro

Dipartimento di matematica e informatica  
 Università di Udine  
<http://www.dimi.uniud.it/mizzaro/>  
 mizzaro@uniud.it  
 Programmazione, lezione 11  
 13 aprile 2015

## Dove siamo

- Mattoni, Programmazione strutturata, Sviluppo incrementale, Array
- Intro metodi/sottoprogrammi
  - Definizione (intestazione e corpo) e uso
  - Parametri formali e attuali, associazione
  - Passaggio parametri per valore
  - Funzioni e procedure
  - Utilità dei metodi
  - Durata
  - Visibilità
  - Gestione dei metodi nella JVM
  - Metodi sovraccarichi
  - Funzioni matematiche predefinite

Stefano Mizzaro - Metodi III

2

## Oggi

- Parametri di tipo array
- Parametri del main
- Esempi
- Ricorsione
  - Definizioni
  - Esempi

Stefano Mizzaro - Metodi III

3

## Parametri di tipo array

- Sintassi simile ai tipi predefiniti
  - anche per il tipo dei valori restituiti
- Differenze
  - Se si modifica un array in un metodo, non viene modificato solo il parametro formale ma anche il parametro attuale!
  - Passaggio "per riferimento" (in realtà è per valore e viene passato per valore il riferimento)

Stefano Mizzaro - Metodi III

4

## Esempio

```
class StampaMatrice {
    static void stampaM(int[][] m) {
        for (int i = 0; i < m.length; i++) {
            for (int j = 0; j < m[i].length; j++)
                System.out.print(m[i][j] + "\t");
            System.out.println();
        }
    }

    public static void main (String[] args) {
        int[][] matrice = { {12, 23, 23, 5678},
                           {987, 87, 3, 0},
                           {12354, 34, 2, 0}};

        stampaM(matrice);
    }
}
```

Stefano Mizzaro - Metodi III

5

## Esempio

```
class ParametriArray1 {
    static void azzera(int[] a) {
        for (int i = 0; i < a.length; i++)
            a[i] = 0;
    }

    public static void main (String[] args) {
        int[] vettore = { 1, 2, 3 };
        for (int i = 0; i < vettore.length; i++)
            System.out.println(vettore[i]);
        azzera(vettore);
        for (int i = 0; i < vettore.length; i++)
            System.out.println(vettore[i]);
    }
}
```

1, 2, 3, 0, 0, 0

Stefano Mizzaro - Metodi III

6

## Diverso da...

```
class Parametri {
    static void inc(int x) {
        x++;
    }

    public static void main (String[] args) {
        int y = 0;
        inc(y);
        System.out.println(y);
    }
}
```

```
>javac Parametri.java
>java Parametri
0
>
```

Stefano Mizzaro - Metodi III

7

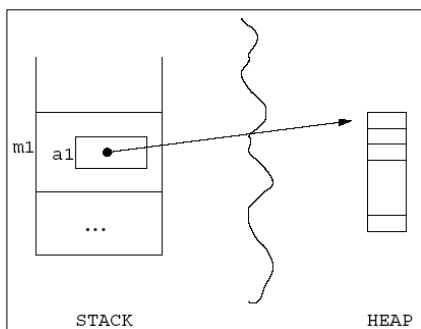
## Perché

- Passaggio per (valore del) riferimento
- Le variabili e i parametri di tipo primitivo stanno sulla pila dei record di attivazione
- Gli array no, stanno sullo heap
  - Zona di memoria separata dalla pila e gestita in modo più "disordinato"
- Sullo stack ci stanno
  - I valori delle variabili di tipo primitivo
  - I riferimenti agli array
    - Riferimento ~ indirizzo in memoria
- Dichiarazione: viene riservata la memoria sullo heap
- Allocazione: viene riservata la memoria sullo heap

Stefano Mizzaro - Metodi III

8

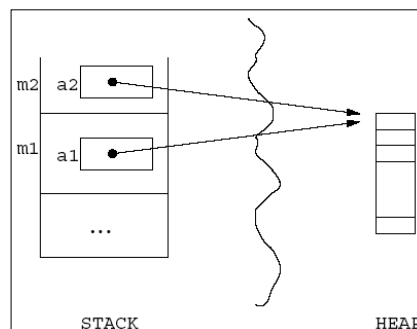
## Perché



Stefano Mizzaro - Metodi III

9

## Perché



Stefano Mizzaro - Metodi III

10

## Esempio rivisto

```
class ParametriArray1 {
    static void azzerare(int[] a, int b) {
        for (int i = 0; i < a.length; i++)
            a[i] = 0;
        b = 0;
    }

    public static void main (String[] args) {
        int[] vettore = { 1, 2, 3 };
        int x = 4;
        for (int i = 0; i < vettore.length; i++)
            System.out.println(vettore[i]);
        System.out.println(x);
        azzerare(vettore, x);
        for (int i = 0; i < vettore.length; i++)
            System.out.println(vettore[i]);
        System.out.println(x);
    }
}
```

1, 2, 3, 4, 0, 0, 0, 4

Stefano Mizzaro - Metodi III

11

## E gli array di array?

- Un array è un array
- Un tipo primitivo è un tipo primitivo
- Ergo, se l'elemento di un array è:
  - di un tipo primitivo, viene passato per valore
  - di tipo array, viene passato per (valore il) riferimento
- Esercizio
  - Scrivere un metodo void `stampaM(int[][] m)` che:
    - visualizza la matrice `m`;
    - invoca un metodo void `azzerare(int[] a)` per azzerare il vettore in `m[0]`

Stefano Mizzaro - Metodi III

12

## Quindi, riassumendo

- I parametri di tipo predefinito (**byte**, **short**, **int**, **long**, **float**, **double**, **char**, **boolean**) vengono passati per valore
  - Le modifiche non si ripercuotono sul parametro attuale
- I parametri di tipo array sono passati "per riferimento" (o meglio, il loro riferimento è passato per valore)
  - Modifiche su parametro formale → modifiche su parametro attuale
  - (anche gli oggetti, di cui parleremo...)

Stefano Mizzaro - Metodi III

13

## Domanda: cosa stampa?

```
class AssegnamentoArray {
    public static void main (String[] args) {
        int[] a = { 1, 2, 3 };
        int[] b;
        b = a;
        a[1] = 10;
        System.out.println(b[1]);
    }
}
```

- "Strano"
  - (meno se si pensa a stack e heap)
- Pericoloso
  - Viene modificato **b** senza neanche menzionarlo

Stefano Mizzaro - Metodi III

14

## Esempio

- Riprendiamo l'esempio dei numeri complessi
  - Un numero complesso rappresentato con un array di due posizioni
  - (una cosa unica, meglio di 2 variabili, gestione più unitaria)
  - Sfruttiamo
    - Passaggio parametri di tipo array
    - Passaggio per valore del riferimento
  - Un po' meglio, ma...
  - ... si potrà fare ancora molto meglio!

Stefano Mizzaro - Metodi III

15

## I numeri complessi (1/2)

```
class Complessi {
    static double[] prodC (double[] a, double[] b){
        double[] c = new double[2];
        c[0] = a[0] * b[0] - a[1] * b[1];
        c[1] = a[0] * b[1] + a[1] * b[0];
        return c;
    }
    static double[] prodC (double[] a, double[] b,
                           double[] c) {
        return (prodC(prodC(a,b),c));
    }
    static void leggiC(double[] a) {
        System.out.print("Parte reale: ");
        a[0] = Leggi.unDouble();
        System.out.print("Parte immaginaria: ");
        a[1] = Leggi.unDouble();
    }
    static void scriviC(double[] a) {
        System.out.print(" "+a[0]+"+i*"+a[1]+"");
    }
}
```

Stefano Mizzaro - Metodi III

16

## I numeri complessi (2/2)

```
public static void main (String[] args) {
    double[] x = new double[2];
    double[] y = new double[2];
    double[] z = new double[2];
    double[] w = new double[2];
    leggiC(x);
    leggiC(y);
    leggiC(z);
    w = prodC(x,y,z);
    scriviC(x);
    System.out.print(" * ");
    scriviC(y);
    System.out.print(" * ");
    scriviC(z);
    System.out.print(" = ");
    scriviC(w);
    System.out.println();
}
scriviC(prodC(x,y,z));
```

Stefano Mizzaro - Metodi III

17

## Parametri del main

- Il main è un metodo!
  - parametri formali: un array di **String**
  - parametri attuali specificati all'invocazione della JVM

```
class Main {
    public static void main (String[] args) {
        System.out.println("Numero argomenti: " +
            args.length);
        for (int i = 0; i < args.length; i++)
            System.out.println(args[i]);
    }
}
```

```
>java Main
Numero argomenti: 0
>java Main pippo pluto minni
Numero argomenti: 3
pippo
pluto
minni
>
```

Stefano Mizzaro - Metodi III

18

## Scaletta

- Parametri di tipo array
- Parametri del main
- Esempi
- Ricorsione
  - Definizioni
  - Esempi

Stefano Mizzaro - Metodi III

19

## Ricorsione

- Definizione ricorsiva in matematica:
  - Definizione di un concetto usando il concetto stesso
- Definizione ricorsiva in Java:
  - Definizione di un metodo usando il metodo stesso

Stefano Mizzaro - Metodi III

20

## Ricorsione in matematica

- Somma (+)
  - $x + 0 = x$
  - $x + succ(y) = succ(x + y)$   
(*succ* è il successore)
- Fattoriale
  - $0! = 1$
  - $n! = n * (n - 1)!$
- Sono definizioni operative
  - $4 + 3 = 4 + succ(2) = succ(4 + 2) \dots$
  - $5! = 5 * 4! = 5 * \dots$
- Caso base e passo

Stefano Mizzaro - Metodi III

21

## Def. ricorsiva vs. circolare

- Somma (+) ricorsiva:
  - $x + 0 = x$
  - $x + succ(y) = succ(x + y)$
- Somma (+) circolare:
  - $x + 0 = x$
  - $x + y = pred(succ(x) + y)$
  - Def. "corretta", ma non operativa:
    - $4 + 3 =$   
 $pred(succ(4)+3) =$   
 $pred(pred(succ(succ(4))+3)) =$   
 $pred(pred(pred(succ(succ(succ(4))))+3))) = \dots$

Stefano Mizzaro - Metodi III

22

## Funzione di Fibonacci

- $F(0) = 1$
- $F(1) = 1$
- $F(n) = F(n - 1) + F(n - 2)$
- $F(5) = F(4) + F(3) = \dots$

Stefano Mizzaro - Metodi III

23

## Algoritmi ricorsivi

- Anche gli algoritmi possono essere definiti ricorsivamente
- Es.: date  $N (= 3^m)$  palline, di cui una più pesante, e una bilancia a 2 bracci, trovare la pallina pesante
  - Divido in 3 gruppi equinumerosi
  - Ne peso 2
  - Se  $=$ , li scarto e scelgo il 3o gruppo, altrimenti scelgo il gruppo più pesante. Poi ripeto ricorsivamente sul gruppo selezionato (che avrà  $3^{m-1}$  palline...)

Stefano Mizzaro - Metodi III

24

## Fattoriale in Java

- Metodo ricorsivo per il calcolo del fattoriale

- $0! = 1$
- $n! = n * (n - 1)!$

```
static int fatt(int n) {
    if (n == 0)
        return 1;
    else
        return n * fatt(n - 1);
}
```

Stefano Mizzaro - Metodi III

25

## Fibonacci in Java

- Metodo ricorsivo per il calcolo della funzione di Fibonacci

- $F(0) = 1$
- $F(1) = 1$
- $F(n) = F(n - 1) + F(n - 2)$

```
static int F(int n) {
    if (n == 0 || n == 1)
        return 1;
    else
        return (F(n - 1) + F(n - 2));
}
```

Stefano Mizzaro - Metodi III

26

## Perché la ricorsione funziona

- La ricorsione e la pila dei record di attivazione
  - Ogni *invocazione/esecuzione* di metodo ha un'alocazione di un record di attivazione sulla pila della JVM
  - I record di attivazione allocati mantengono i risultati intermedi. Vediamo un es.

```
class Fattoriale {
    static int fatt(int n) {
        int res = 0;
        if (n == 0)
            res = 1;
        else
            res = n * fatt(n - 1);
        return res;
    }
    public static void main (String[] args) {
        int x = 4;
        System.out.println(fatt(x));
    }
}
```

St

27

## Ricorsione mutua (indiretta)

- Un metodo **m1** ne chiama un altro **m2** (che ne chiama un altro **m3** ...) che chiama **m1**
- Es. (giochino!): Definiamo, usando la ricorsione mutua, due metodi **pari** e **dispari**

- Idea:
  - $pari(n) = dispari(n-1)$
  - $dispari(n) = pari(n-1)$
- Base della ricorsione:
  - $pari(0) = true$
  - $dispari(0) = false$

Stefano Mizzaro - Metodi III

28

## Il codice

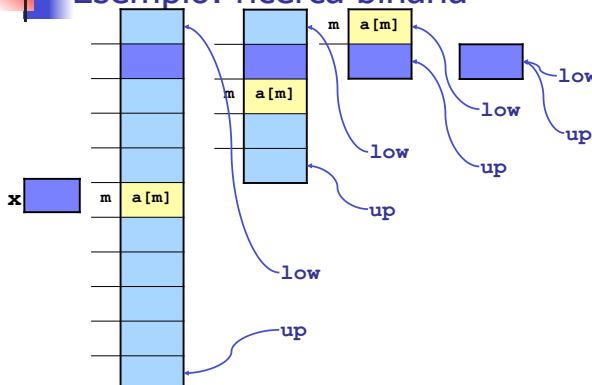
```
static boolean pari(int n) {
    if (n == 0)
        return true;
    else
        return dispari(n - 1);
}

static boolean dispari(int n) {
    if (n == 0)
        return false;
    else
        return pari(n - 1);
}
```

Stefano Mizzaro - Metodi III

29

## Esempio: ricerca binaria



Stefano Mizzaro - Metodi III

30

## Lo pseudocodice

- Cerco in tutto l'array:
  - Individuo l'elemento mediano m
  - Se  $x == m$ , ho finito
  - Se  $x > m$  cerco (ricorsivamente) nella seconda metà
  - Se  $x < m$  cerco (ricorsivamente) nella prima metà

Stefano Mizzaro - Metodi III

31

## Il codice

```
static int ricerca(char[] a, char x, int low,
                  int up) {
    int m;
    m = (up + low) / 2;
    if (up < low)
        return -1;
    else if (a[m] == x)
        return m;
    else if (a[m] < x)
        return ricerca(a, x, m + 1, up);
    else //a[m] > x
        return ricerca(a, x, low, m - 1);
}
```

Stefano Mizzaro - Metodi III

32

## Esercizi

- Per capire meglio, vediamo esempi
  - Ex. 1. Metodo ricorsivo per assegnare i ad a[i]
    - (suggerimento: usare un parametro ausiliario, come per ricerca binaria)
    - "Assegna i da i in poi"
  - Ex. 2. Metodo ricorsivo per calcolare il MCD con l'algoritmo di Euclide
  - Ex. 3. Metodo ricorsivo per invertire un array
  - ...
- Vediamo le soluzioni (non testate)

Stefano Mizzaro - Metodi III

33

## Ex. 1

- Invocazione: **assegna(x, 0)**
- "Assegna i da i in poi"
- Schema tipico

```
static void assegna(int[] a, int i) {
    if (i < 0)
        assegna(a, 0);
    else if (i < a.length) {
        a[i] = i;
        assegna(a, i+1);
    }
}
```

Stefano Mizzaro - Metodi III

34

## Ex. 2

```
static int mcd(int x, int y) {
    if (x == y)
        return x;
    if (x < y)
        return mcd(x, y-x);
    //if (x > y)
        return mcd(x-y, y);
}
```

```
static int mcd(int x, int y) {
    if (x == 0 || y == 0)
        return (x==0?y:x);
    if (x < y)
        return mcd(x, y%x);
    // if (x > y)
        return mcd(x%y, y);
}
```

Stefano Mizzaro - Metodi III

35

## Ex. 3

```
static void inverti(int[] a, int i) {
    if (i < 0)
        inverti(a, 0);
    else if (i < a.length / 2) {
        scambia(a, i, a.length - 1 - i);
        inverti(a, i+1);
    }
}
```

```
static void scambia(int[] a, int i, int j) {
    int temp = a[i];
    a[i] = a[j];
    a[j] = temp;
}
```

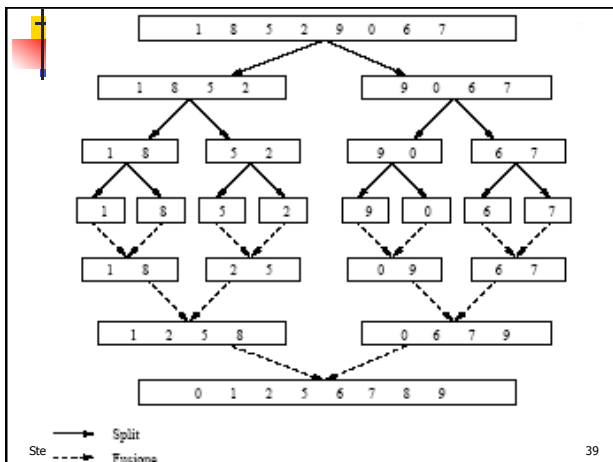
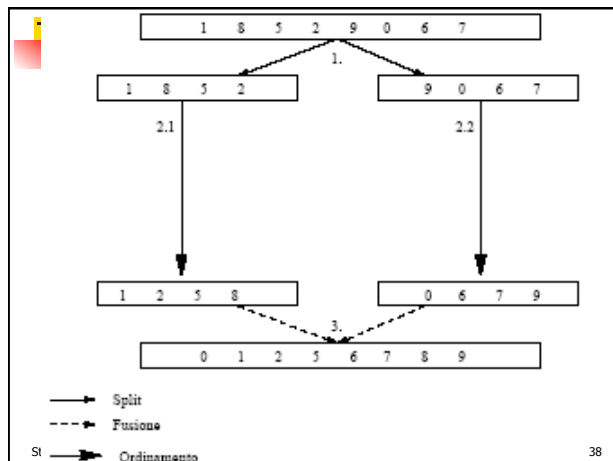
Stefano Mizzaro - Metodi III

36

### Esempio: ordinamento per fusione (*merge sort*)

- Ragioniamo in modo ricorsivo
  - 1. **spezza** l'array a metà;
  - 2.1. **ordina** la prima metà;
  - 2.2. **ordina** la seconda metà;
  - 3. **fonde** le due metà
- La ricorsione è sull'operazione **ordina**
  - che viene effettuata su un pezzo di array più corto...
  - ...che verrà ordinato con lo stesso procedimento!
- C'è anche la fusione...

Stefano Mizzaro - Metodi III 37



### Il codice (1/2)

- Non servono array ausiliari
- Ipotizziamo lunghezza =  $2^n$

```

/** Ordina l'array a da x a u */
static void ordina(char[] a, int x, int u) {
    int m;
    if (x != u) {
        m = (x + u) / 2;
        ordina(a, x, m);
        ordina(a, m + 1, u);
        fondi(a, x, m, u);
    }
}
    
```

Stefano Mizzaro - Metodi III 40

### Il codice (2/2)

```


/** Fonde l'array a, da x a m e da m + 1 a u */
static void fondi (char[] a, int x, int m, int u) {
    char[] b = new char[u - x + 1];
    int i = x, j = m + 1, k = 0;
    while (i <= m && j <= u)
        if (a[i] <= a[j])
            b[k++] = a[i++];
        else
            b[k++] = a[j++];
    while (i <= m)
        b[k++] = a[i++];
    while (j <= u)
        b[k++] = a[j++];
    for (k = 0; k <= u - x; k++)
        a[k + x] = b[k];
}
    
```

Stefano Mizzaro - Metodi III 41

### Sulla potenza della ricorsione

- Espressività
  - A volte la definizione più semplice e naturale di un algoritmo è ricorsiva (cfr. ricerca binaria, fatt, mcd...)
  - Es.: algoritmi di visita su alberi... ASD...
  - Le istruzioni di iterazione e la ricorsione sono equivalenti!
- È difficile
  - Si capisce pensando all'implementazione
  - domanda x lode
  - ne riparlerete
- N.B. Ricorsione possibile grazie a:
  - Metodi
  - 1 record di attivazione per ogni invocazione

Stefano Mizzaro - Metodi III 42



## Riassunto

- Metodi
  - Parametri di tipo array
  - Parametri del main
  - Esempi
  - Ricorsione
- Libro: fino a cap. 6
- Eserciziario: fino a § 3.2

Stefano Mizzaro - Metodi III 43