

RICORSIONE

- La ricorsione è la capacità di un metodo di richiamare se stesso.
- Una definizione ricorsiva consiste in due parti:
 - *Caso base*: vengono elencati gli elementi di base che sono i blocchi costitutivi dell'insieme;
 - *Caso induttivo/ricorsivo*: vengono fornite le regole per costruire nuovi elementi a partire dagli elementi di base o da elementi che sono già stati costruiti.

RICORSIONE

- La funzione che definisce l'elevamento di un numero x ad una potenza non negativa n è un esempio di funzione ricorsiva:

$$x^n = \begin{cases} 1 & \text{se } n = 0 \text{ (caso base)} \\ x \cdot x^{n-1} & \text{se } n > 0 \text{ (caso ricorsivo)} \end{cases}$$

- Implementiamo questa funzione in Java e analizziamo come avviene un'invocazione ricorsiva.

ESEMPIO

```
/*102*/ double power(double x, int n){
/*103*/     if (n==0)
/*104*/         return 1;
/*105*/     return x*power(x,n-1);
}

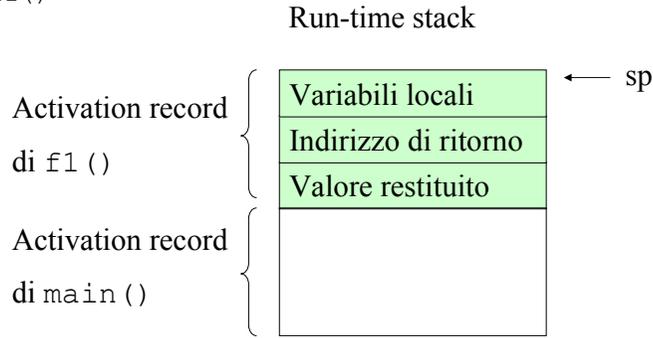
public static void main(String[] args) {
    ...
/*136*/     y = power(5.6,2);
    ...
}
```

INVOCAZIONE DI UN METODO

- Cosa succede quando un metodo viene invocato?
- Il *sistema* deve memorizzare lo stato del chiamante, creare lo spazio per lo stato del chiamato e sapere dove riprendere l'esecuzione del programma dopo che il metodo è terminato.
- Lo stato del metodo, cioè la documentazione di attivazione (*activation record*), viene allocato sulla pila di esecuzione (*run-time stack*) e un puntatore (*stack pointer*) tiene traccia della posizione nella pila.

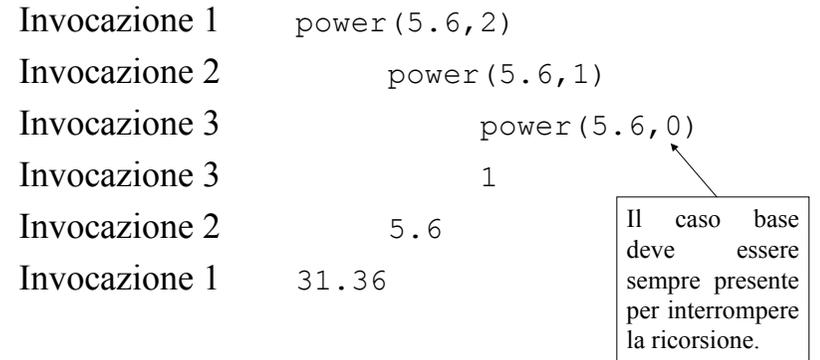
INVOCAZIONE DI UN METODO

Per esempio se il `main()` chiama il metodo `f1()`



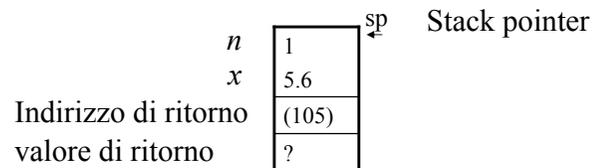
CHIAMATA RICORSIVA

Una traccia delle chiamate ricorsive dell'esempio:

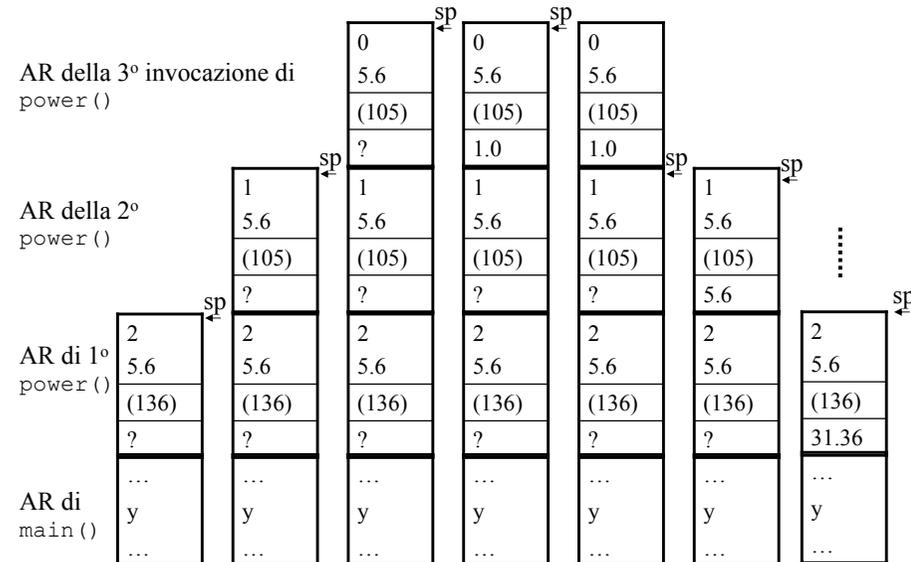


CHIAMATA RICORSIVA

- Vediamo una possibile rappresentazione grafica della pila di esecuzione dell'esempio precedente:
 - I numeri nei commenti (per es. `/*136*/`) rappresentano l'indirizzo dato dal sistema a quella "linea di codice";
 - Il ? indica il valore di ritorno non ancora calcolato;



CHIAMATA RICORSIVA



RICORSIONE IN CODA

- L'esempio visto rappresenta un tipo di ricorsione chiamato ricorsione in coda.
- La ricorsione in coda è caratterizzata dall'uso di una sola invocazione ricorsiva al termine del metodo.
- Tale *ricorsione* può essere trasformata in *iterazione* attraverso l'uso di un ciclo.
- Quale vantaggio ad usare la ricorsione? La ricorsione sembra essere più intuitiva, perchè più simile alla definizione originale, e permette di scrivere codice conciso.

RICORSIONE IN CODA

- Tuttavia bisogna porre attenzione alle *risorse* che vengono utilizzate per produrre le chiamate ricorsive: lo spazio di memoria utilizzato e il tempo necessario ad attivare nuove chiamate.
- In generale la ricorsione in coda non è una caratteristica consigliabile.
- La potenza della ricorsione è legata a particolari strutture dati o ad algoritmi specifici.

RICORSIONE NON IN CODA

- Un esempio di tale ricorsione e la visualizzazione di una linea di ingresso in ordine inverso:

```
F:\temp\reverse
Inserisci una stringa: 123456
654321
```

```
F:\temp\reverse
Inserisci una stringa: qwert
trewq
```

```
/*200*/ void reverse() {
/*201*/     char ch=getchar();
/*202*/     if (ch!='\n') {
/*203*/         reverse();
/*204*/         printf("%c",ch);
    } }
```

RICORSIONE NON IN CODA

Vediamo la pila di esecuzione con la stringa "ABC".

