



# ELEMENTI DI INFORMATICA

DOCENTE: FRANCESCO MARRA

INGEGNERIA CHIMICA

INGEGNERIA ELETTRICA

SCIENZE ED INGEGNERIA DEI MATERIALI

INGEGNERIA GESTIONALE DELLA LOGISTICA E DELLA PRODUZIONE

INGEGNERIA NAVALE

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II  
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE

# FUNZIONI RICORSIVE





# AGENDA

- Le funzioni ricorsive
- 

# FUNZIONI RICORSIVE

- Funzione che chiama se stessa...
  - Finchè non esiste un caso risolvibile in maniera diretta che termina la ricorsione
- Si applica il principio di induzione
  - Individuazione del passo base
    - Il valore della funzione è direttamente determinato
  - Determinazione del passo induttivo (o *ricorsivo*)
    - Il valore della funzione è determinato dal valore della funzione applicata ai predecessori dei suoi argomenti

# FUNZIONI RICORSIVE: ESEMPI

- Fattoriale di un numero  $n$

$$n! = \begin{cases} 1 & \text{se } n = 0 \\ n \cdot (n - 1)! & \text{se } n > 0 \end{cases}$$

- Elevazione a potenza di un numero  $x$

$$x^n = \begin{cases} 1 & \text{se } n = 0 \\ x \cdot x^{n-1} & \text{se } n > 0 \end{cases}$$

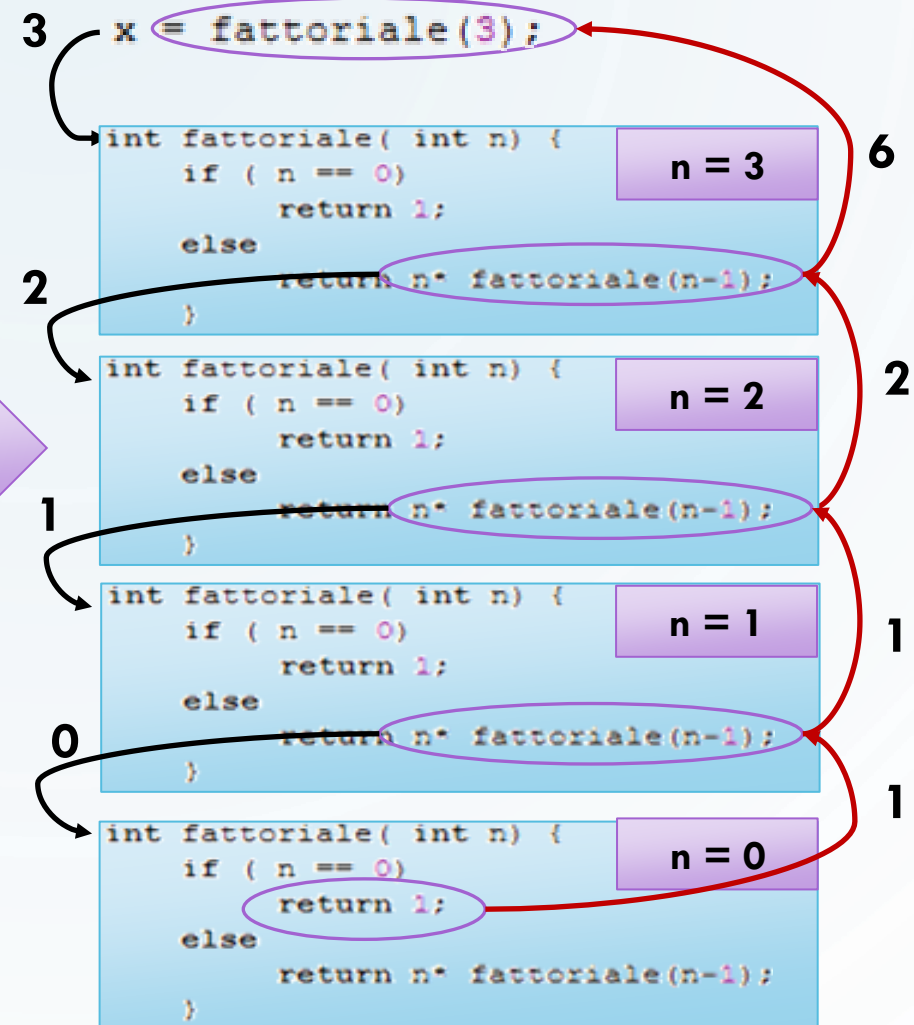
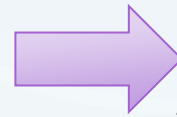
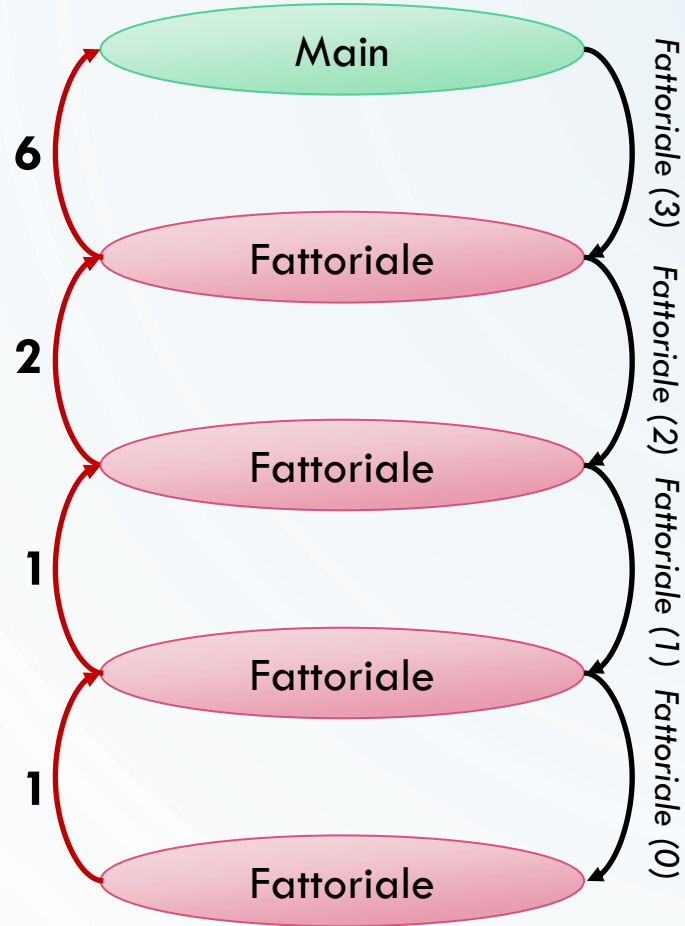
# ESEMPIO: CALCOLO DEL FATTORIALE

$$n! = \begin{cases} 1 & \text{se } n = 0 \\ n \cdot (n-1)! & \text{se } n > 0 \end{cases}$$



```
/* calcolo ricorsivo del fattoriale */  
  
int fattoriale(int n) {  
    if (n == 0)                // passo base  
        return 1;  
    else                        // passo ricorsivo  
        return n* fattoriale(n-1);  
}
```

# ESEMPIO: CALCOLO DEL FATTORIALE



# RICORSIONE ED ITERAZIONE

- Rappresentano soluzioni alternative per risolvere problemi che richiedono l'esecuzione ripetuta di operazioni


Ricorsione	Iterazione
Favorisce la semplicità della codifica	Comporta una maggiore complessità della soluzione
Richiede molta memoria poiché mantiene uno stack dei valori da processare da ogni chiamata di funzione	Privilegia l'efficienza di esecuzione poiché comporta una minore occupazione di memoria
Utilizza una selezione nella condizione di terminazione	Gestisce esplicitamente le condizioni iniziali e di terminazione
La ripetizione si ottiene come effetto delle ripetute chiamate di funzione	La ripetizione è insita nella struttura di controllo di flusso ( <i>while</i> o <i>for</i> )




# RICORSIONE ED ITERAZIONE: CALCOLO DEL MASSIMO COMUN DIVISORE

- Il Massimo Comun Divisore tra due numeri  $m$  e  $n$  è definito dall'algoritmo di Euclide come:

$$mcd(m, n) = \begin{cases} m & \text{se } m = n \\ mcd(m - n, n) & \text{se } m > n \\ mcd(m, n - m) & \text{se } n > m \end{cases}$$



```
int mcdIterativo(int m, int n){  
    while (m != n)  
        if(m > n)  
            m = m - n;  
        else  
            n = n - m;  
    return m;  
}
```



```
int mcd(int m, int n){  
    if (m == n)  
        return m;  
    else  
        if(m > n)  
            return mcd(m-n, n);  
        else  
            return mcd(m, n-m);  
}
```

# ESERCIZIO 1

- Scrivere una funzione ricorsiva che calcola la potenza  $k$ -esima di un numero  $x$  ( $k > 0$ ). Riscrivi l'esercizio del calcolo dell'integrale definito di un polinomio, usando le funzioni e la funzione ricorsiva della potenza

## ESERCIZIO 2

- Scrivere una funzione ricorsiva che controlla se una stringa è palindroma, ovvero se “rigirandola” non cambia
  - Ad es. “otto” e “anna ” sono palindromi

## ESERCIZIO 3

- Scrivere un programma che trova gli zeri di una funzione continua con il metodo della bisezione

**DOMANDE, DUBBI, PERPLESSITÀ**

