

Tutorato Lezione 2

Tutor: Maria Angela Pellegrino
mapellegrino@unisa.it

Parantesi bilanciate

Esercitazione

Parentesi

bilanciate

Vogliamo solo verificare se una data espressione aritmetica è ben bilanciata rispetto alle parentesi

Non ci interessa sapere se gli operatori in essa contenuti sono corretti e se hanno il giusto numero di operandi

Esercitazione


Parentesi bilanciate – modalità valutazione

CASI DI TEST:

stringa vuota, ab, (ab), ([a]) > TRUE
(ab, a)b, ([a]) > FALSE

ISPEZIONE DEL CODICE

Errori comuni - Attenzione!



- Non usa lo stack o non lo usa propriamente (e.g., uso di variabili o altre strutture di supporto)
- Non sono state definite tutte le funzioni del template fornito o non definizione nel modo corretto (isCorresponding deve verificare che i caratteri siano parentesi e siano dello stesso tipo)
- Modifica di stack per risolvere l'esercizio
- Errore di gestione di alcuni casi, principalmente stringa vuota e parentesi chiusa non aperta
- Stringa senza parentesi percepita come non bilanciata

- 1 è valutato come TRUE, quindi non occorre scrivere `if(isValid()==1)` ma basta `if(isValid())`
- Ruolo invertito di true e false (valore 0 e 1) e non uniforme
- Indentazione

Coda

Esercizio 1

Implementare una **Queue** con
due Stack.

Ricorsione

Ricorsione - FATTORIALE

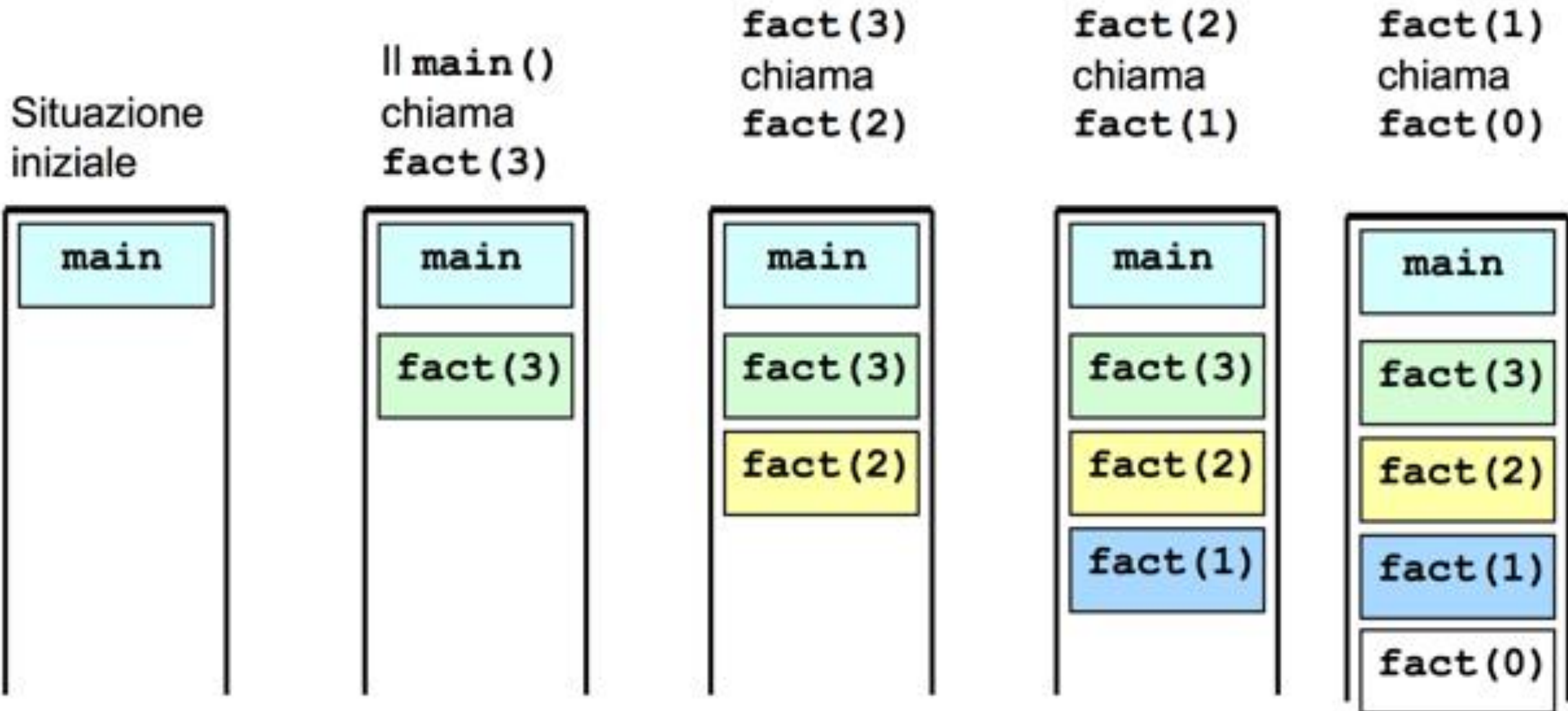
Fattoriale :

Base : $0! = 1$

Passo : $n! = n * (n-1)!$

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

Ex. Invochiamo fact(3)

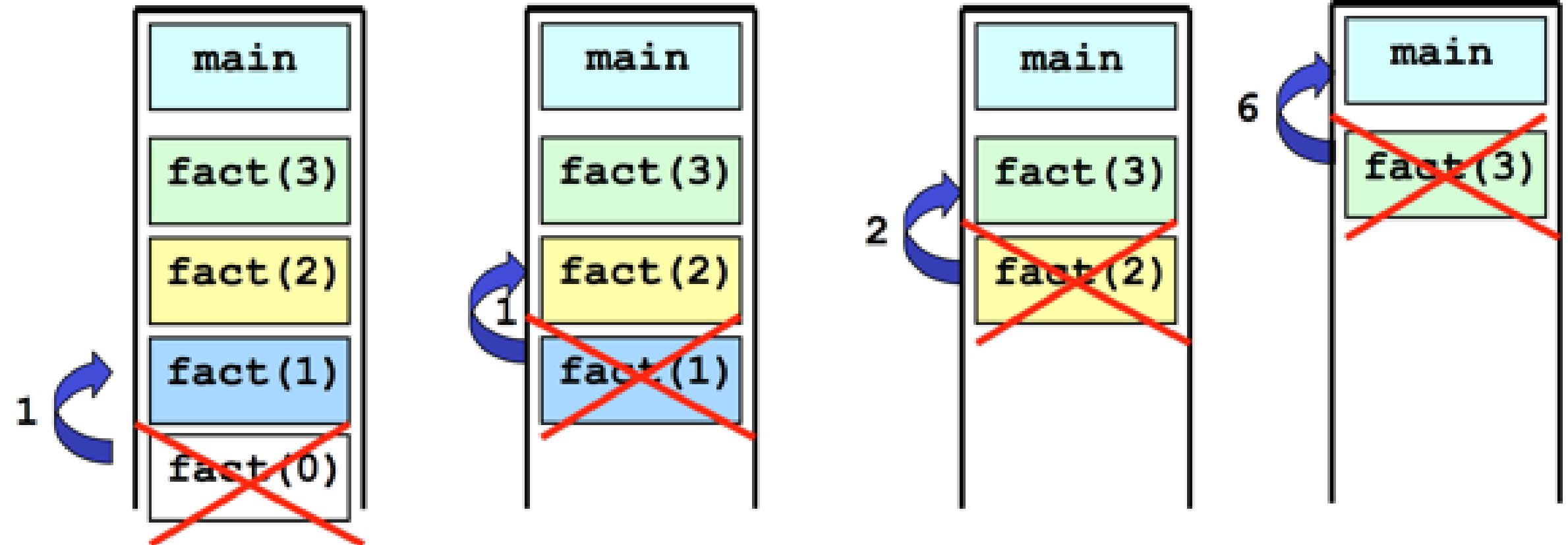


fact(0) termina restituendo il valore 1. Il controllo torna a **fact(1)**

fact(1) effettua la moltiplicazione e termina restituendo il valore 1. Il controllo torna a **fact(2)**

fact(2) effettua la moltiplicazione e termina restituendo il valore 2. Il controllo torna a **fact(3)**

fact(6) effettua la moltiplicazione e termina restituendo il valore 6. Il controllo torna al main.



Ricorsione - SOMMA PRIMI N NUMERI

Ricorsione - SOMMA PRIMI N NUMERI

```
int sommaFinoA(int n){  
    if (n==1)  
        return 1;  
    else  
        return sommaFinoA(n-1)+n;  
}
```

Ricorsione - NUMERI DI FIBONACCI

$$\text{fib}(n) = \begin{cases} 0, & \text{se } n=0 \\ 1, & \text{se } n=1 \\ \text{fib}(n-1) + \text{fib}(n-2), & \text{altrimenti} \end{cases}$$

Ricorsione - NUMERI DI FIBONACCI

```
int fibonacci(int n){  
    if(n<2)  
        return n;  
    else  
        return fibonacci(n-1)+fibonacci(n-2);  
}
```

Ricorsione - 2^n

Base : $2^0 = 1$

Passo : $2^n = 2 * 2^{n-1}$

Ricorsione - 2^n

Base : $2^0 = 1$

Passo : $2^n = 2 * 2^{n-1}$

```
int pow(int n){  
    if (n==0)  
        return 1;  
    else  
        return 2* pow(n-1);  
}
```

Ricorsione - $2*n + 5$

Base : $f(0) = 5$

Passo : $f(n) = f(n-1) + 2$

$$f(n) = 2n + 5 = 2(n-1+1) + 5 = 2(n-1) + 2 + 5 = f(n-1) + 2$$

Ricorsione - $2*n + 5$

Base : $f(0) = 5$

Passo : $f(n) = f(n-1)+2$

$$f(n) = 2n+5 = 2(n-1+1) + 5 = 2(n-1) + 2 + 5 = f(n-1) + 2$$

```
int f(int n){  
    if(n==0) return 5;  
    else return f(n-1)+2;  
}
```

Esercizio 2

Calcolo della **potenza**.

- Dati x e y ,
calcolare RICORSIVAMENTE x^y .

Esercizio 3

Calcolo del **quoziente**.

- Dati x e $y > 1$ calcolare $\frac{x}{y}$.