



Fondamenti di Informatica

Alma Artis
Francesca Pratesi (ISTI, CNR)

Funzioni ricorsive

ESERCIZI

- Scrivere una funzione che chiede all'utente di inserire una serie numerica di interi, fermandosi quando viene inserito due volte consecutive lo stesso numero.
- Si scriva una funzione con un parametro numerico k. La funzione legge k valori numerici in input e calcola (e restituisce in output) true se i valori letti sono ordinati in senso crescente, false altrimenti.
 - Si scriva il corpo di un programma di verifica che legga in input un valore k, invochi la funzione definita al punto precedente e stampi il messaggio “valori ordinati in senso crescente”, oppure “valori non ordinati in senso crescente”. Si testi il programma almeno sui seguenti dati di input (6,(3,6,78,91,100,107)), (5,(23, -4, 34, 56, 90)).



ESERCIZI 2

- Si scriva una funzione senza parametri che legga in input dei valori numerici, continuando a leggere finché in numeri letti sono ordinati in senso crescente. La funzione termina non appena la sequenza dei numeri inseriti non è (più) ordinata in senso crescente. La funzione deve calcolare e restituire in output la lunghezza della sequenza dei numeri ordinati in senso crescente letti dall'input.
 - Si scriva un programma di verifica per la funzione definita al punto precedente, invocando la funzione e stampandone il risultato. Si verifichi il comportamento della funzione anche nel caso in cui si inseriscano due numeri uguali consecutivi.





RICORSIONE E FUNZIONI RICORSIVE

FUNZIONI RICORSIVE

- Una funzione ricorsiva è una funzione che contiene una chiamata a se stessa

```
function recursive_function(n){  
    ...  
    ...  
    recursive_function(n-1);  
}
```



ASPETTI FONDAMENTALI

- Una definizione ricorsiva consta di due passi fondamentali:
 - **Passo base:** condizione in cui la funzione può restituire un valore. Non sono necessarie altre chiamate ricorsive. La ricorsione finisce qui.
 - **Passo ricorsivo (induttivo):** la funzione restituisce un valore che dipende da dati semplificati o ridotti. In questo caso c'è la chiamata ricorsiva.



RICORSIONE

- Ad ogni chiamata ricorsiva i dati a cui la funzione viene applicata vengono semplificati (es. si invoca la funzione con parametro $n-1$)
- Procedendo in questo modo si arriverà ad un punto corrispondente ad un caso base
- Una chiamata ricorsiva non è differente da una chiamata di funzione standard:
 - la funzione chiamante sospende la propria esecuzione per eseguire la nuova chiamata
 - l'esecuzione della funzione chiamante riprende quando la chiamata ricorsiva termina
 - c'è solitamente una cascata di chiamate ricorsive (fino al raggiungimento del passo base)
 - si applicano gli stessi concetti già visti a proposito dello stato



ESEMPIO: FATTORIALE

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

← passo base

← passo ricorsivo


↑
i dati vengono semplificati

ESEMPIO: FATTORIALE

```
function fattoriale(n){  
  if (n==1)                ← passo base  
    return 1;  
  else  
    return n*fattoriale(n-1); ← passo induttivo  
}
```


↑
i dati vengono semplificati

ESEMPIO DI ESECUZIONE: FATTORIALE(4)

 n=4

```
function fattoriale(n){  
    if (n==1)  
        return 1;  
    else  
        return n * fattoriale(n-1);  
}
```

ESEMPIO DI ESECUZIONE: FATTORIALE(4)

 n=4

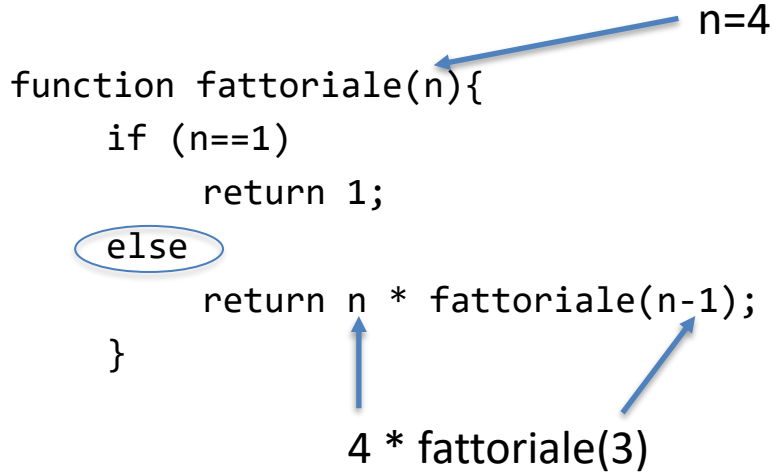
```
function fattoriale(n){  
    if (n==1)  
        return 1;  
    else  
        return n * fattoriale(n-1);  
}
```

ESEMPIO DI ESECUZIONE: FATTORIALE(4)


```
function fattoriale(n){  
    if (n==1)  
        return 1;  
    else  
        return n * fattoriale(n-1);  
}
```

$n=4$


$4 * \text{fattoriale}(3)$



ESEMPIO DI ESECUZIONE: FATTORIALE(4)


 n=4

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=3

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```

ESEMPIO DI ESECUZIONE: FATTORIALE(4)


 n=4


```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=3

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


ESEMPIO DI ESECUZIONE: FATTORIALE(4)


function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}



function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}


3 * fattoriale(2)


ESEMPIO DI ESECUZIONE: FATTORIALE(4)

 n=4

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=3

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=2

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```



ESEMPIO DI ESECUZIONE: FATTORIALE(4)

 n=4

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=3


```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=2

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```

ESEMPIO DI ESECUZIONE: FATTORIALE(4)

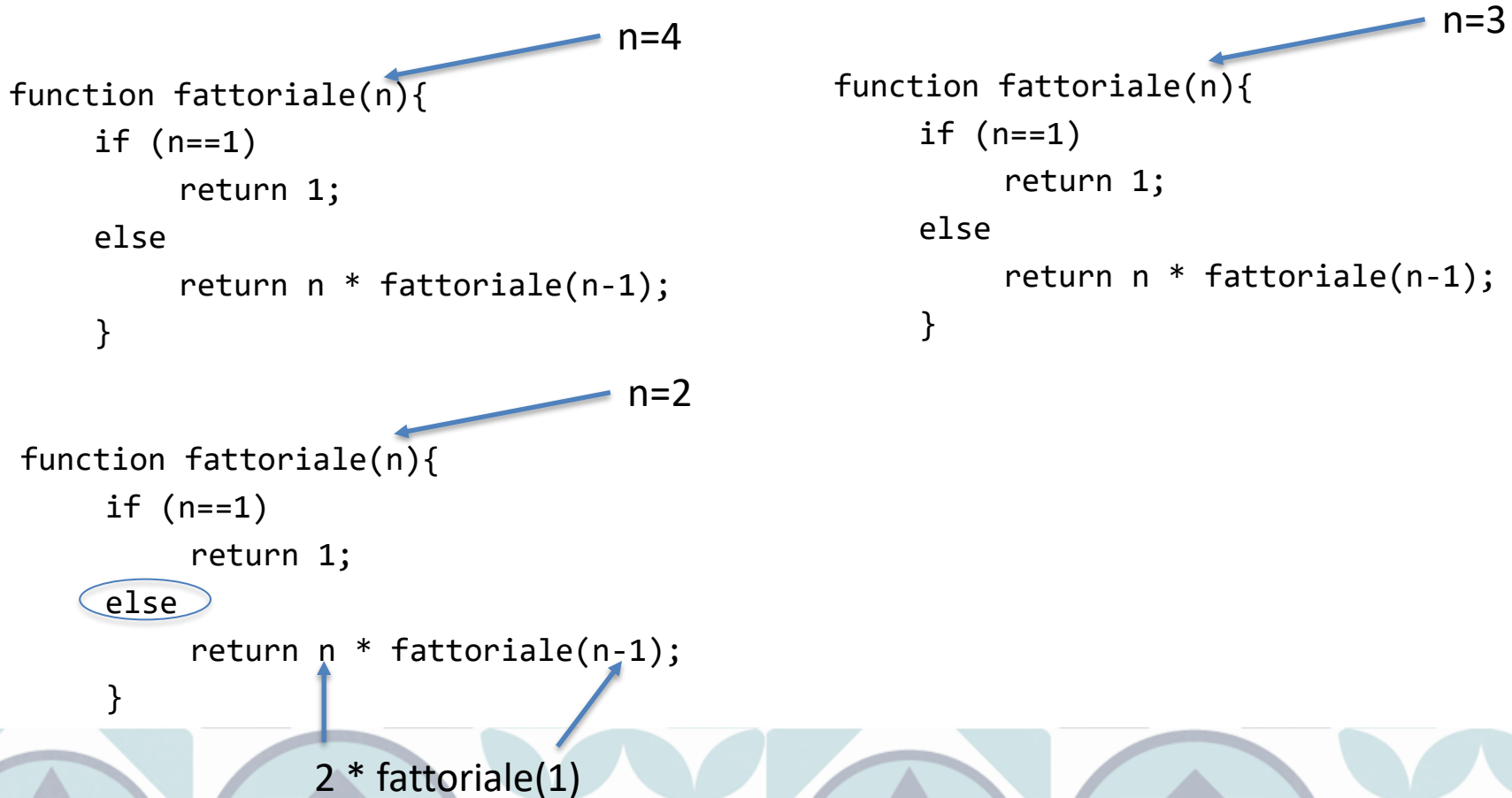

function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}


function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}



function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}


2 * fattoriale(1)


ESEMPIO DI ESECUZIONE: FATTORIALE(4)

 n=4


```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```

 n=3

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=2

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=1

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


ESEMPIO DI ESECUZIONE: FATTORIALE(4)

 n=4


```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```

 n=3

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=2

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=1

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


ESEMPIO DI ESECUZIONE: FATTORIALE(4)

 n=4

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```

 n=3


```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```


 n=2


```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```

ESEMPIO DI ESECUZIONE: FATTORIALE(4)

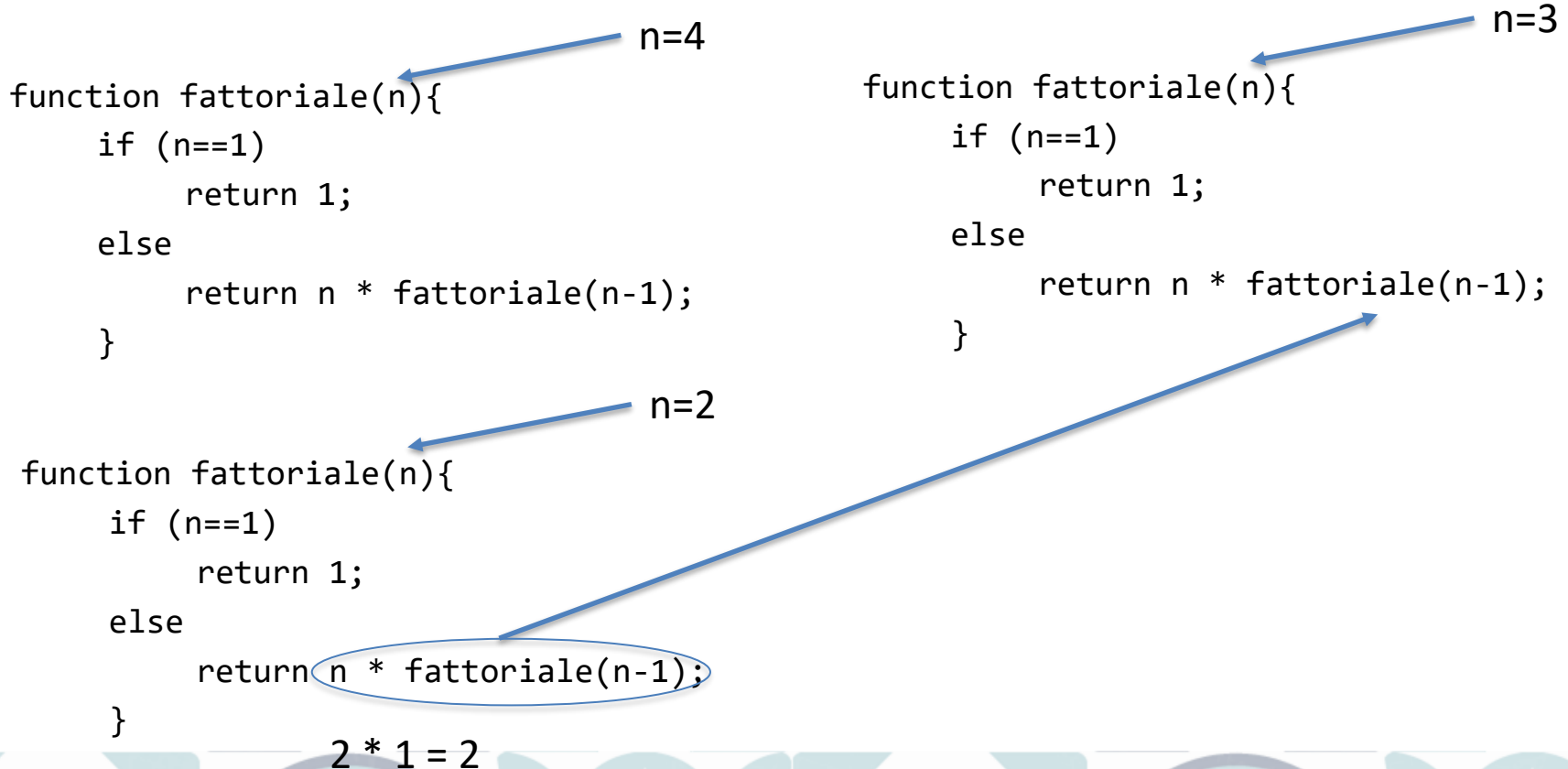

function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}


function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}



function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}

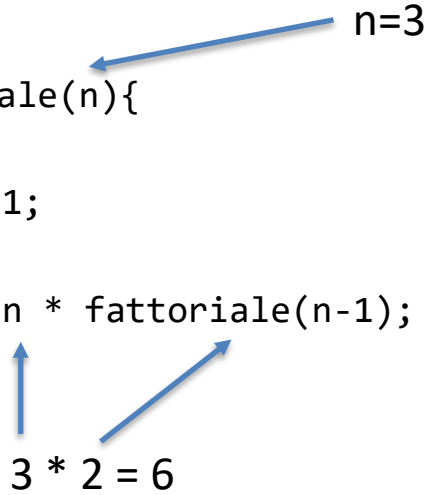

 $2 * 1 = 2$

ESEMPIO DI ESECUZIONE: FATTORIALE(4)



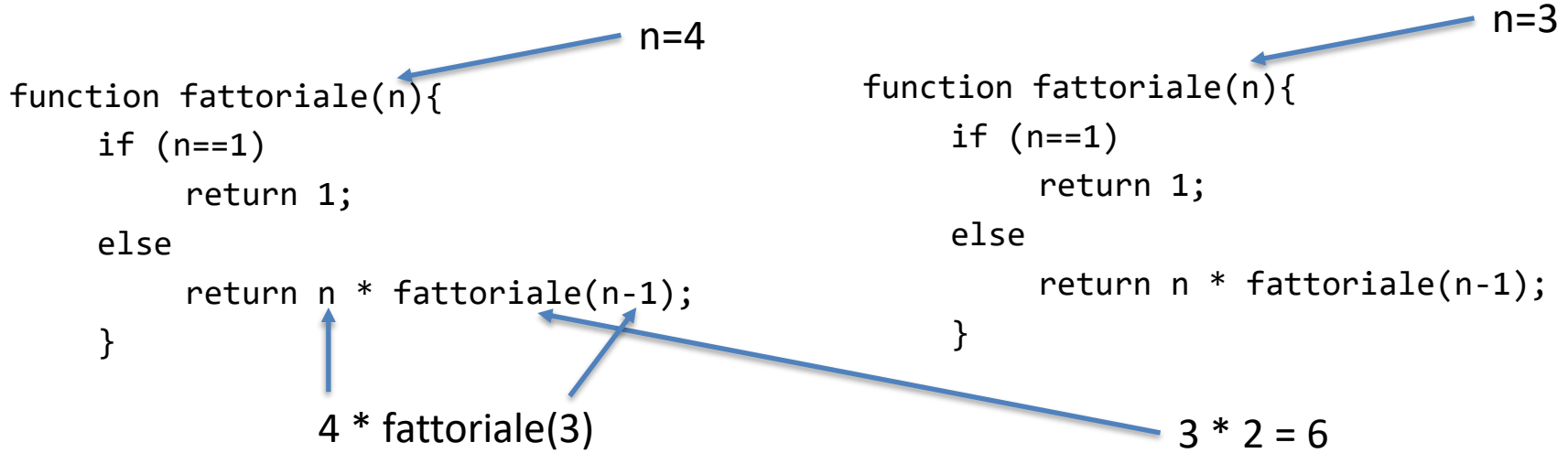
ESEMPIO DI ESECUZIONE: FATTORIALE(4)


function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}


function fattoriale(n){
 if (n==1)
 return 1;
 else
 return n * fattoriale(n-1);
}

3 * 2 = 6

ESEMPIO DI ESECUZIONE: FATTORIALE(4)



ESEMPIO DI ESECUZIONE: FATTORIALE(4)

```
function fattoriale(n){  
  if (n==1)  
    return 1;  
  else  
    return n * fattoriale(n-1);  
}
```

$n=4$

$4 * 6 = 24$