Iterazioni VS Ricorsione

Alcuni algoritmi che sembrano necessariamente richiedere la presenza di istruzioni iterative, possono essere conveniente ridisegnati sfruttando la logica della programmazione ricorsiva.
La ricorsione e l'iterazione rappresentano due scelte alternative per risolvere problemi che richiedono l'esecuzione ripetuta di certe operazioni.

I termini di confronto sono:

- la semplicità di codifica;
- l'efficienza di esecuzione.

Il confronto va fatto caso per caso ma, in linea di massima:

- l'iterazione privilegia la semplicità di codifica;
- la ricorsione privilegia l'efficienza di esecuzione.

L'uso dell'iterazione è da preferire quando la soluzione iterativa e ricorsiva sono paragonabili dal punto di vista della complessità, oppure se l'occupazione di memoria generata dalla ricorsione viene evitata tramite la soluzione iterativa. Viceversa, l'uso della ricorsione è da preferire quando la complessità della soluzione iterativa è decisamente superiore a quella della soluzione ricorsiva, oppure se l'occupazione di memoria è necessaria alla soluzione del problema (ad ogni passo si deve tener traccia dello stato) e si verificherebbe anche nella soluzione iterativa.

Se si pensa di poter implementare una funzione secondo logica ricorsiva, non conviene cercare di ricostruire la sua esecuzione da parte dell'elaboratore; bisogna invece verificare la correttezza logica della condizione di terminazione e del passo di ricorsione.

Esercizio

Utilizzare prima la logica iterativa e poi la logica ricorsiva, per costruire una funzione in grado di calcolare il fattoriale di un numero.
Codice R

```
 fattoriale.iterativo = function(n)
{
  ris=1
  if (n>0)
    for(i in 1:n) ris=ris*i
  return(ris)
}
a=fattoriale.iterativo(3)
a
```

```
fattoriale.ricorsivo = function(n)
{
  ris=1
  if (n>0) ris=n*fattoriale.ricorsivo(n-1)
  return(ris)
 }
a=fattoriale.ricorsivo(3)
a
```

Le due soluzioni proposte producono lo stesso risultato: nel secondo caso però non vengono utilizzate istruzioni iterative e si perviene al risultato voluto mediante ricorsione, ovvero per chiamate successive della stessa funzione.

Ricorsione contro iterazione tabella riassuntiva

La ricorsione è un metodo per chiamare una funzione all'interno della stessa funzione.	L'iterazione è un blocco di istruzioni che si ripete finché la condizione data non è vera.
Complessità dello spazio	
La complessità spaziale dei programmi ricorsivi è superiore alle iterazioni.	La complessità dello spazio è inferiore nelle iterazioni.
Velocità	
L'esecuzione della ricorsione è lenta.	Normalmente, l'iterazione è più veloce della ricorsione.
Condizione	
Se non c'è una condizione di terminazione, può esserci una ricorsione infinita.	Se la condizione non diventa mai falsa, sarà un'infinita iterazione.
Pila	
In ricorsione, lo stack viene utilizzato per memorizzare le variabili locali quando viene chiamata la funzione.	In una iterazione, la pila non viene utilizzata.
Leggibilità del codice	
Un programma ricorsivo è più leggibile.	Il programma iterativo è più difficile da leggere rispetto a un programma ricorsivo.