INTRODUZIONE ALLA RICORSIONE

La funzione fattoriale costituisce il classico esempio di funzione ricorsiva

Sui testi di matematica, il fattoriale è spesso indicato con il simbolo “!”, e definito come segue:

(i) 0! = 1

(ii) n! =n * (n-1)!

Per maggiore leggibilità e per omogeneità al modo “informatico” di denotare le funzioni, questa definizione può essere riscritta come segue

(i) fact(0) = 1

(ii) fact(n) = n * fact(n-1)

Tali *definizioni sono ricorsive, perchè adottano una funzione nella definizione della funzione stessa*. Tuttavia, esse sono corrette, e permettono di definire, se pur “implicitamente”, tutti i possibili valori assunti dalla funzione.

(1) fact(0) = 1 per definizione, parte (i)

(2) fact(1) = 1 * fact(0) per definizione, parte (ii). Ma, dato (1), fact(0)=1, e quindi, sostituendo, fact(1)=1*1=1

(3) fact(2) = 2 * fact(1) per definizione, parte (ii). Ma, dato (2), fact(1)=1, e quindi, sostituendo, fact(2)=2*1=2

(4) fact(3) = 3 * fact(2) per definizione, parte (ii). Ma, dato (3), fact(2)=2, e quindi, sostituendo, fact(3)=3*2=6

(5) .....

Si notino le analogie fra definizioni ricorsive come quella del fattoriale e la nozione matematica di *INDUZIONE*. Informalmente parlando, volendo ad esempio dimostrare per induzione che una proprietà P vale su un insieme di elementi, si dimostra che tale proprietà vale per un elemento (passo BASE della dimostrazione), e poi, assumendo che P valga su n elementi (ipotesi induttiva), si dimostra che P vale anche su n+1 elementi (passo di induzione). Ad esempio, per dimostrare che P vale sui numeri interi, si dimostra che

(1) P vale su 0 (passo base)

(2) Se P vale (ipotesi induttiva) su un generico numero naturale n, allora (passo di induzione) vale su n+1

In C, come in (quasi) tutti i linguaggi di programmazione, è possible dare una definizione ricorsiva delle funzioni. Ad esempio, in C, la funzione fattoriale puo' essere correttamente definite come segue:

```
 /* calcolo ricorsivo del fattoriale */

int fact(int n)

{

  if (n==0)  return 1;

  else  return n*fact(n-1);

}

/* passo base: fact(0)=1 */

/* ipotesi induttiva: fact(n-1) restituisce (n-1)! */

/* passo induttivo: n! = n*(n-1)! */

/* terminazione: ad ogni richiamo, n è decrementata di uno, fino a raggiungere il valore 0 del passo base */

/* la complessità   tempo: O(n)   spazio: O(n) */
```

Naturalmente, la funzione fattoriale puo' essere definite in C anche in modo iterativo

```
 int fact_iter(int n)

{

  int product = 1;

  while(n>1)

  {

    product=product*n;

    n=n-1;

   }

returnproduct;

}
```

Si noti tuttavia lo stile completamente differente di soluzione del problema. Nel caso della soluzione ricorsiva, il problema e' stato definito "in astratto" per induzione, descrivendo il passo base ed il passo induttivo. Al contrario, nella definizione iterativa, sono state esplicitate le operazioni (un ciclo di prodotti e sottrazioni, finche' il parametro n non diventa 1) da eseguire per raggiungere il risultato.

NOTA. Questa differenza puo’ apparire poco significativa su problem semplici quail la definizione del fattoriale, ma puo’ essere invece cruciale per problem piu’ complessi, quali ad esempio “le Torri di Hanoi” o le funzioni di ordinamento (es. Quicksort, Mergesort), o ancora problemi concernenti strutture dati complesse quail alberi e grafi.

Il fatto che, tramite una definizione ricorsiva di funzione, non sia necessario esplicitare tutti i passi di computazione necessari alla risoluzione del problema (dal momento che ci si puo’ avvalere, all’atto della definizione, dell’ipotesi induttiva), non significa che, a livello di ESECUZIONE di una funzione ricorsiva da parte del computer, tali passi non vengano eseguiti. Per andare ad analizzare piu’ in dettaglio come avvenga l’esecuzione di una funzione ricorsiva possiamo avvalerci del modello dei RECORD DI ATTIVAZIONE.