3.6 L'operatore virgola

L'operatore virgola, che ha priorità più bassa di tutti gli altri, permette di inserire all'interno delle espressioni più istruzioni. Per esempio, un `for` può includere le inizializzazioni all'interno di *esp1*:

```
for(numero=1, somma=0; numero!=0;)
```

In questo caso *esp3* non è presente, ma se necessario anch'essa potrebbe contenere più di un istruzione:

```
for(i=1, j=5; i<10 && j<100; i++, j=i*j)
```

Nell'esempio, `i` viene inizializzato a 1 e `j` a 5. Il ciclo si ripete finché `i` è minore di 10 e contemporaneamente `j` è minore di 100. A ogni ciclo `i` viene incrementato di 1 e a `j` viene assegnato il prodotto di `i` per `j`. Al limite si potrebbe scrivere:

```
for(numero=1, somma=0; numero!=0; printf("Inser. intero:\t),
scanf("%d",&numero), somma=somma +numero)
  ;
```

comprendendo tutte le istruzioni che costituiscono il calcolo della somma dei numeri introdotti dall'utente all'interno di *esp3*. Vale la pena sottolineare che le istruzioni in *esp3* sono inframmezzate dalla virgola e non devono essere presenti punti e virgola.

✓ **NOTA**

Questo modo di operare porta a istruzioni lunghissime, difficilmente leggibili; consigliamo pertanto di usare l'operatore virgola essenzialmente là dove ci siano da inizializzare o incrementare più variabili che controllano il ciclo.