IF E UNLESS

IF

```
if n == 1 then
    puts "N vale uno"
elsif n == 2 then
    puts "N vale due"
else
    puts "N non vale ne uno ne due"
end
```

È possibile omettere, qualora non necessario, sia il ramo elseif sia il ramo else e arrivare alla forma base.

```
if espressione then
   istruzione
end
```

è anche possibile sostituire alla parola chiave then i due punti (:), o anche ometterla del tutto se il nostro if è scritto su più linee. L'esempio di prima diventa dunque:

```
if n == 1
   puts "N vale uno"
elsif n == 2
   puts "N vale due"
else
   puts "N non vale ne uno ne due"
end
```

Possiamo inserire un'espressione if anche in coda ad una normale istruzione

UNLESS

Opposto all'if abbiamo lo statement unless che esegue le istruzioni associate all'espressione che risulta falsa:

```
unless n > 10
   puts "N non è maggiore dieci"
else
   puts "N è maggiore di dieci"
end
```

Anche dell’unless ne esiste una versione modificatrice di istruzione:

```
puts "N è maggiore di dieci" unless n < 10
```

Una forma ancora più ermetica prevede l’uso dell’operatore ? nel seguente modo:

```
segno = n >= 0 ? "positivo" : "negativo"
```

se la condizione è vera viene eseguita la prima espressione, se è falsa la seconda. È buona norma usare questa sintassi solo nei casi in cui non va a scapito della leggibilità, uno dei punti di forza del Ruby.