

```
boolean r = (profession.equals("teacher") || profession.equals("soldier")) && age >= 18 && !origin.equals("north");  
---
```

è una ****congiunzione**** (AND) di tre componenti:

1. `(profession.equals("teacher") || profession.equals("soldier"))` - disgiunzione
2. `age >= 18` - confronto relazionale
3. `!origin.equals("north")` - negazione di un confronto

Traccia di Valutazione: Valori Forniti

****Assunzioni sui valori delle variabili:****

- `profession = "teacher"`
- `age = 45`
- `origin = "south"`

Fase 1: Valutazione della Prima Componente (Disgiunzione)

Operando Sinistro: `profession.equals("teacher")`

Questa espressione invoca il metodo `equals()` dell'oggetto `profession` con argomento `"teacher"`.

****Processo di valutazione:****

- Il metodo `equals()` confronta il contenuto della stringa memorizzata in `profession` con il letterale `"teacher"`
- `profession` contiene `"teacher"`
- Il contenuto è identico

- **Risultato: `true`**

Operando Destro: `profession.equals("soldier")`

Questa espressione invoca il metodo `equals()` dell'oggetto `profession` con argomento `"soldier"`.

Processo di valutazione:

- Il metodo `equals()` confronta il contenuto della stringa memorizzata in `profession` con il letterale `"soldier"`
- `profession` contiene `"teacher"`
- Il contenuto è diverso
- **Risultato: `false`**

Operatore OR (`||`) sulla Prima Componente

L'operatore OR combina i risultati:

...

true || false

...

Tavola di verità applicata:

Dalla tavola di verità dell'OR, quando un operando è `true`, il risultato è `true` indipendentemente dall'altro operando.

Inoltre, a causa della **valutazione pigra (short-circuit)**, poiché il primo operando di OR è `true`, il compilatore non valuta nemmeno il secondo operando (anche se in questo caso lo abbiamo valutato per completezza espositiva).

Risultato della prima componente: `true`

Fase 2: Valutazione della Seconda Componente (Confronto Relazionale)

Operatore di Confronto: `age >= 18`

Questa espressione confronta il valore della variabile `age` con il letterale numerico `18`, utilizzando l'operatore di confronto relazionale `>=`.

****Processo di valutazione:****

- `age` contiene il valore intero `45`
- Il letterale `18` rappresenta il valore intero `18`
- L'operatore `>=` valuta se `45` è maggiore o uguale a `18`
- La relazione è vera
- ****Risultato: `true`****

Fase 3: Valutazione della Terza Componente (Negazione)

Operando della Negazione: `origin.equals("north")`

Questa espressione invoca il metodo `equals()` dell'oggetto `origin` con argomento `"north"`.

****Processo di valutazione:****

- Il metodo `equals()` confronta il contenuto della stringa memorizzata in `origin` con il letterale `"north"`
- `origin` contiene `"south"`
- Il contenuto è diverso
- ****Risultato di `origin.equals("north")`: `false`****

Operatore NOT (!) sulla Terza Componente

L'operatore NOT nega il risultato:

...

!false

...

****Processo di valutazione:****

- L'operatore NOT trasforma il valore booleano secondo la sua tavola di verità

- `false` negato diventa `true`
- ****Risultato della terza componente: `true`****

Fase 4: Combinazione Finale (Congiunzione)

Ora abbiamo i tre risultati intermedi:

...

true && true && true

...

Prima Congiunzione: `true && true`

L'operatore AND combina i primi due risultati:

...

true && true

...

****Dalla tavola di verità di AND:**** Quando entrambi gli operandi sono `true`, il risultato è `true`.

****Risultato intermedio: `true`****

Seconda Congiunzione: `true && true`

L'operatore AND combina il risultato precedente con il terzo operando:

...

true && true

...

****Dalla tavola di verità di AND:**** Quando entrambi gli operandi sono `true`, il risultato è `true`.

****Risultato finale: `true`****

Rappresentazione Strutturale

L'espressione può essere rappresentata come un ****albero sintattico**** dove ogni nodo rappresenta un'operazione:

...

```
      &&
     / \
    &&  !
   /\   |
  ||  >= equals
 /\   |  /\
== == 45 18 origin "north"
 /\ /\
prof "t" prof "s"
```