Corso di base JAVA

Mauro Donadeo
mail: mauro.donadeo@gmail.com

Le decisioni

HT Lab

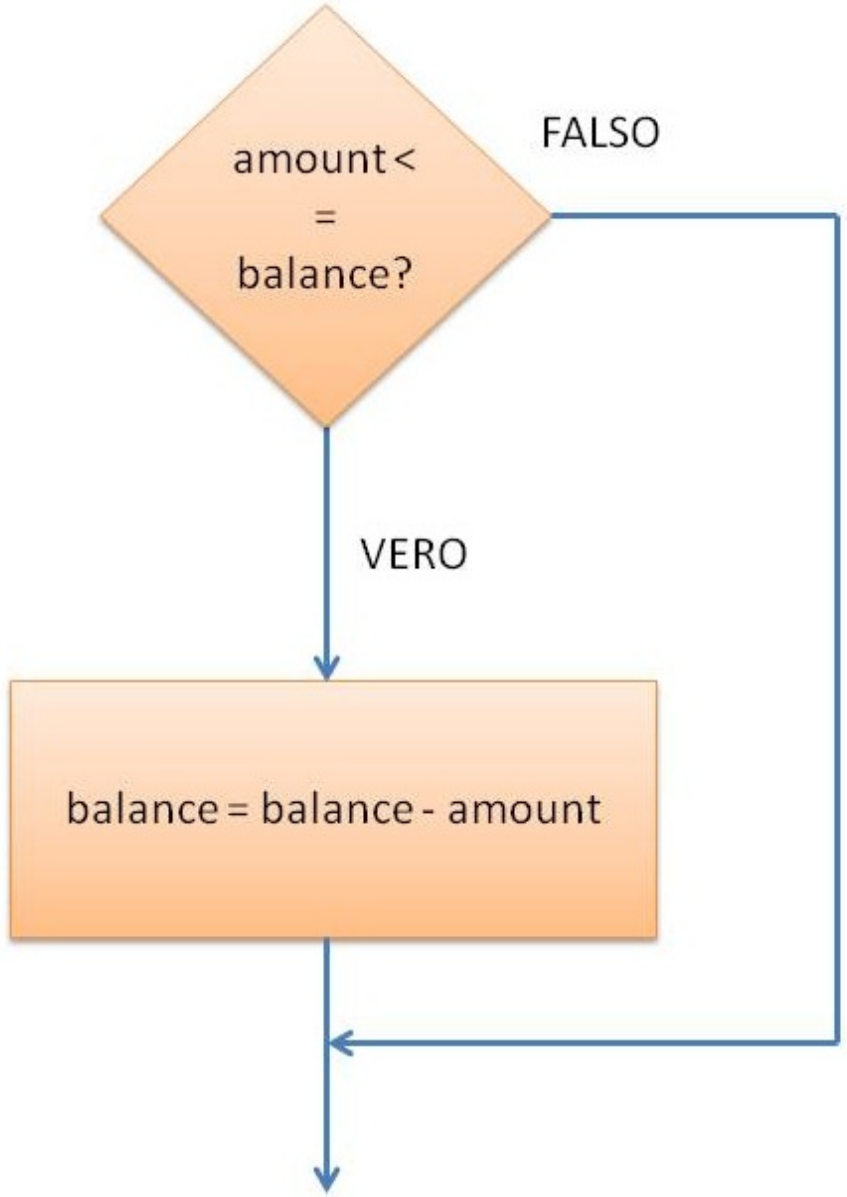
Enunciato if

Il programma **BankAccount** consente di prelevare tutto il denaro che si vuole

- il saldo `balance` può diventare **negativo**
  - **balance = balance - amount;**
- È una situazione assai poco realistica
- Quindi il programma deve **controllare** il saldo ed agire di conseguenza. *Consentire il prelievo o no.*

```
if (amount <= balance)
   balance = balance - amount;
```

- L'enunciato if si usa per realizzare una decisione ed è diviso in due parti
  - una verifica
  - un corpo
- Il corpo viene eseguito **se e solo se** **la verifica ha successo**



Tipi di enunciato in Java

- Enunciato semplice
  - **balance = balance - amount;**
- Enunciato composto
  - **if(x >= 0) x=0;**
- blocco di enunciati
  - {**zero o più enunciati di qualsiasi tipo**}

Proviamo ad emettere un messaggio d'errore in caso di prelievo non consentito:

```
if(amount <= balance)
    balance = balance - amount;
if(amount > balance)
    System.out.println("Conto scoperto");
```

Problema

Se si modifica la prima verifica, bisogna ricordarsi di modificare anche la seconda.

Problema

Se il corpo del primo `if` viene eseguito, la verifica del secondo `if` usa il nuovo valore di `balance`, introducendo **errore logico**

- quando si preleva più della metà del saldo disponibile

La clausola else

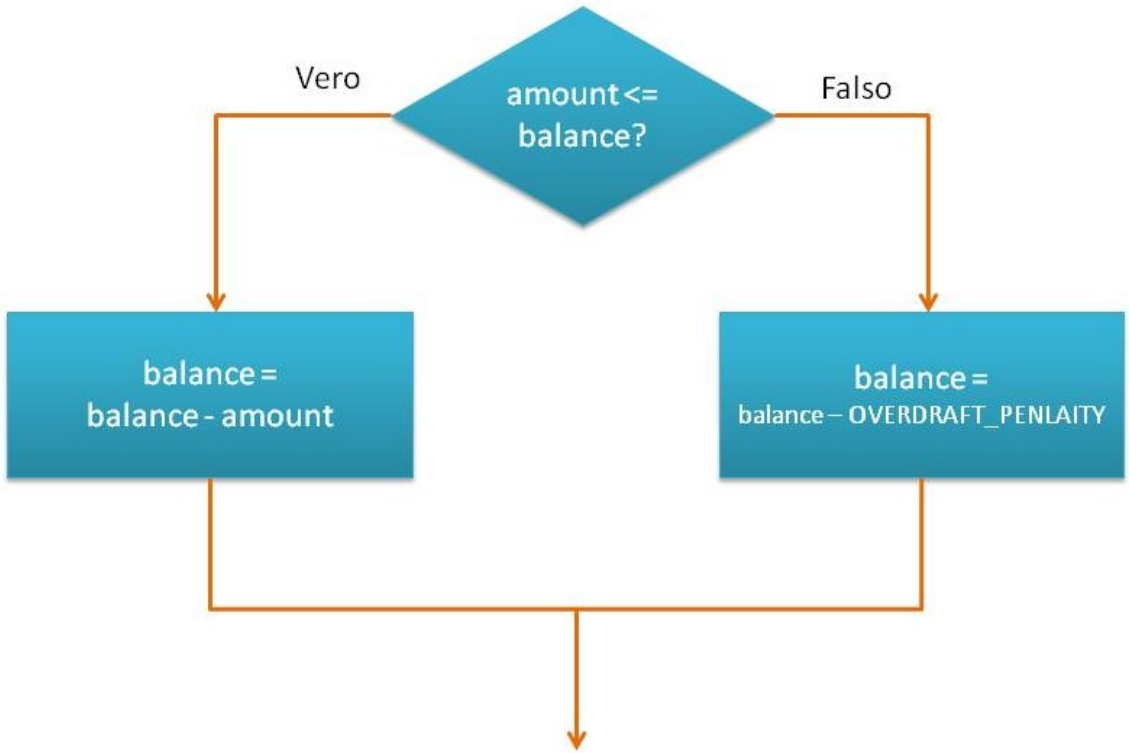
Per realizzare un'alternativa, si utilizza la clausola **else** dell'enunciato `if`

```
if(amount <= balance)
    balance = balance - amount;
else
    System.out.println("Conto aperto");
```

Vantaggio: ora c'è una sola verifica

- se la verifica ha successo, viene eseguito il primo corpo dell'enunciato `if/else`
- altrimenti, viene eseguito il secondo dopo;

```
if (amount <= balance)
    balance = balance - amount;
else{
    System.out.println("Conto scoperto");
    balance = balance - OVERDRAFT_PENALTY;
}
```



Confrontare valori

Le condizioni dell'enunciato if sono molto spesso dei confronti tra due valori

if(x >= 0)

Gli operatori di confronto si chiamano operatori relazionali

>	Maggiore
>=	Maggiore o uguale
<	Minore
<=	Minore o uguale
==	Uguale
! =	Diverso

Attenzione: negli gli operatori da due caratteri **non** vanno inseriti spazi intermedi

Operatori relazionali

Fare **molta** attenzione nella differenza tra l'operatore relazionale == e l'operatore assegnazione =

```
a = 5; //assegno ad a il valore 5

if(a == 5) //esegue enunciato
    //enunciato
```

Confrontare numeri in virgola mobile

I numeri in virgola mobile hanno una precisione limitata ed i calcoli possono introdurre errori di arrotondamento e troncamento

Tali errori sono inevitabili e bisogna fare molta attenzione nella formulazione di verifiche che coinvolgono numeri con in virgola mobile.

Affinché gli errori di arrotondamento non influenzino la logica del programma, i confronti tra numeri in virgola mobile devono avere una tolleranza

- Verifica di uguaglianza tra x ed y (di tipo double):

$$|x - y| < \epsilon \textbf{ con } \epsilon = \textbf{1E-14}$$

- Scelta migliore se x,y sono molto grandi o molto piccoli

$$|x - y| < \epsilon * max(|x|, |y|) \textbf{ con } \epsilon = \textbf{1E-14}$$

Esercizio

Calcolare la radice quadrata di 2 e confrontare il risultato con 2. Se il risultato è uguale a 2 Stampare "ok!" altrimenti "Errore"

- Per confrontare stringhe si usa il metodo **equals**

  if(s1.equals(s2))
- Per confrontare stringhe ignorando la differenza tra maiuscole e minuscole si usa **equalsIgnorecase**

  if(s1.equalsIgnorecase(s2))
- Non usare **mai** l'operatore di ugualianza per confrontare stringhe. **Usare sempre equals**