Equals e Hashset / 23-04

- equals permette di confrontare i riferimenti di due variabili

- hashCode sfrutta la funzione di hash
  (final per non farlo ridefinire da sottoclassi)

Con una ridefinizione di equals posso controllare se l'oggetto passato è un'istanza di un tipo compatibile

```
if (obj instanceof SimpleVariable sv)
    return nome.equals(sv.nome)
```

↑ var this — ↑ var passata

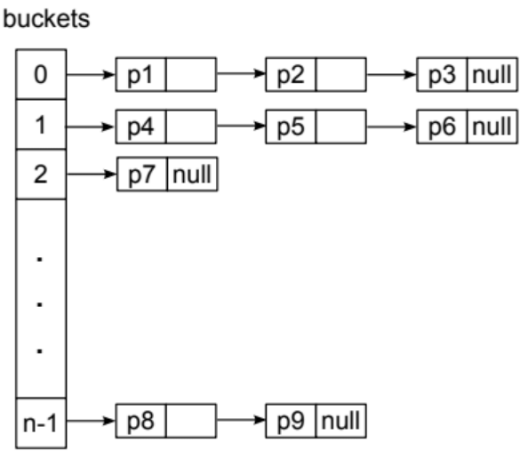
ovvero se passo il confronto, sono lo stesso oggetto se hanno stesso nome

L'HashCode() restituisce il cod hash associato a quel nome ovvero calcola con una funzione di hash un valore distribuito in modo omogeneo.
Utile quando si implementano tabelle di hash, HashSet<E> (HashMap)

Nota: se ridefinisco "equals()" devo ridefinire "hashCode()", il comportamento è anomalo

```
import java.util.HashSet;
class SimpleTest {
  public static void main(String[] args) {
    var pointSet = new HashSet<Point>();
    var p1 = new CartesianPoint();
    var p2 = new CartesianPoint();
    pointSet.add(p1);
    assert p1.equals(p2);
    // most likely fails if 'hashCode()' is not redefined in 'CartesianPoint'
    assert pointSet.contains(p2);
    }
}
```

Esempio di hashSet




```
hash(p1)==hash(p2)==hash(p3)==0
hash(p4)==hash(p5)==hash(p6)==1
hash(p7)==2
hash(p8)==hash(p9)==n-1
```

La funzione di hash spesso utilizzata è quella del modulo

è un insieme di valori, utilizza una hashMap

l'Hashmap contiene coppie <key, value> in cui ad ogni chiave è associato un valore

Utilizzando un decorator pattern ColoredPoint che non eredita da Point. Ciò permette di definire un equal simetrico e transitivo, inoltre non duplica il codice fra punti cartesiani e polari.
Dunque il confronto è:

point.getX() == coloredpoint.getX()

└──── ≠ ────┘

ottimenti potrei fare equal, ma Point non è compatibile con ColoredPoint (no eredità (estensione))

Implementazioni di operazioni su alberi

Vi sono due approcci:

- data-oriented: focus sui tipi di nodi, metodi implementati per ogni tree class
- object oriented: una classe con tutti i metodi per visitare diversi tipi di nodi (metodi generici)

↳ soluzione generale
↳ non deve modificare il codice esistente quando aggiungo operazioni
↳ meno efficiente