

Fondamenti di Java

Soluzione esercizio hashCode

Esercizio

Definire una classe con una hashCode **corretta**.
Aggiungere delle istanze "uguali" a un set, e controllare la dimensione del set ottenuto. Vi torna il valore ottenuto per la dimensione del set?

Definire una classe con una hashCode **non corretta**.
Aggiungere delle istanze "uguali" a un set, e controllare la dimensione del set ottenuto. Vi torna il valore ottenuto per la dimensione del set? Potete spiegare quel che osservate?

Implementazione corretta

```
public class Test {  
    int a=0;  
    Test(int k) {a=k;}  
    public boolean equals(Test c){  
        if (c==null) return false;  
        if (a==c.a) return true;  
        return false;  
    }  
    public int hashCode() {return a; /* return 0 */ }  
    public static void main(String a[]){  
        Test k1=new Test(2);  
        Test k2=new Test(2);  
        Collection col=new HashSet();  
        col.add(k1);  
        col.add(k2);  
        System.out.println(col.size());  
    }  
}
```

OUTPUT: 1

Implementazione sbagliata

```
public class Test {  
    int a=0;  
    Test(int k) {a=k;}  
    public boolean equals(Test c){  
        if (c==null) return false;  
        if (a==c.a) return true;  
        return false;  
    }  
    public int hashCode() {return (int)  
        (Math.random()*100);}  
    public static void main(String a[]){  
        Test k1=new Test(2);  
        Test k2=new Test(2);  
        Collection col=new HashSet();  
        col.add(k1);  
        col.add(k2);  
        System.out.println(col.size());  
    }  
}
```

OUTPUT: 2

Fondamenti di Java



Static

Modificatori: static

Variabili e metodi associati ad una Classe anziché ad un Oggetto sono definiti “static”.

Le variabili statiche servono come singola variabile **condivisa** tra le varie istanze

I metodi possono essere richiamati **senza creare una istanza**.

Variabili "static": esempio 1

```
public class S {
    static int instanceCount = 0; //variabile "di classe"
    S() {instanceCount++;}
}

public class A {
    public static void main(String a[]) {
        new A();
    }
    A() {
        for (int i = 0; i < 10; ++i) {
            S instance=new S();
        }
        System.out.println("# of instances:
            "+S.instanceCount);
    }
}
```

Output:

of instances: 10

Variabili "static": esempio 2

```
class S {
    static int instanceCount = 0; //variabile "di classe"
    S() {instanceCount++;}
    public void finalize() {instanceCount--;}
}

public class A {
    public static void main(String a[]) {
        new A();
    }
    A() {
        for (int i = 0; i < 10; ++i)
            S instance=new S();
    }
    System.out.println("# of instances:
"+S.instanceCount);
    System.gc();
    System.out.println("# of instances:
"+S.instanceCount);
}
```

Output:

```
# of instances: 10
# of instances: 0
```

Metodi "static": esempio 1

```
class S {
    static int instanceCount = 0; //variabile "di classe"
    S() {instanceCount++;}
    static void azzeraContatore() {instanceCount=0;}
}

public class A {
    public static void main(String a[]) {
        new A();
    }
    A() {
        for (int i = 0; i < 10; ++i) {
            if (i%4==0) S.azzeraContatore();
            S instance=new S();
        }
        System.out.println("instanceCount:
"+S.instanceCount);
    }
}
```

**Può agire solo su
variabili statiche!**

Output:
instanceCount: 2

Ruolo:
**Metodi che agiscono su
variabili statiche**

metodi "static": esempio 2

**Notare la
maiuscola!
(per convenzione)**

```
Math.sqrt(double x);  
System.gc();  
System.arraycopy(...);  
System.exit();  
Integer.parseInt(String s);  
Float.parseFloat(String s);
```

**Ruolo:
analogo alle
librerie del C**

Che cos'è :
`System.out.println()` ?

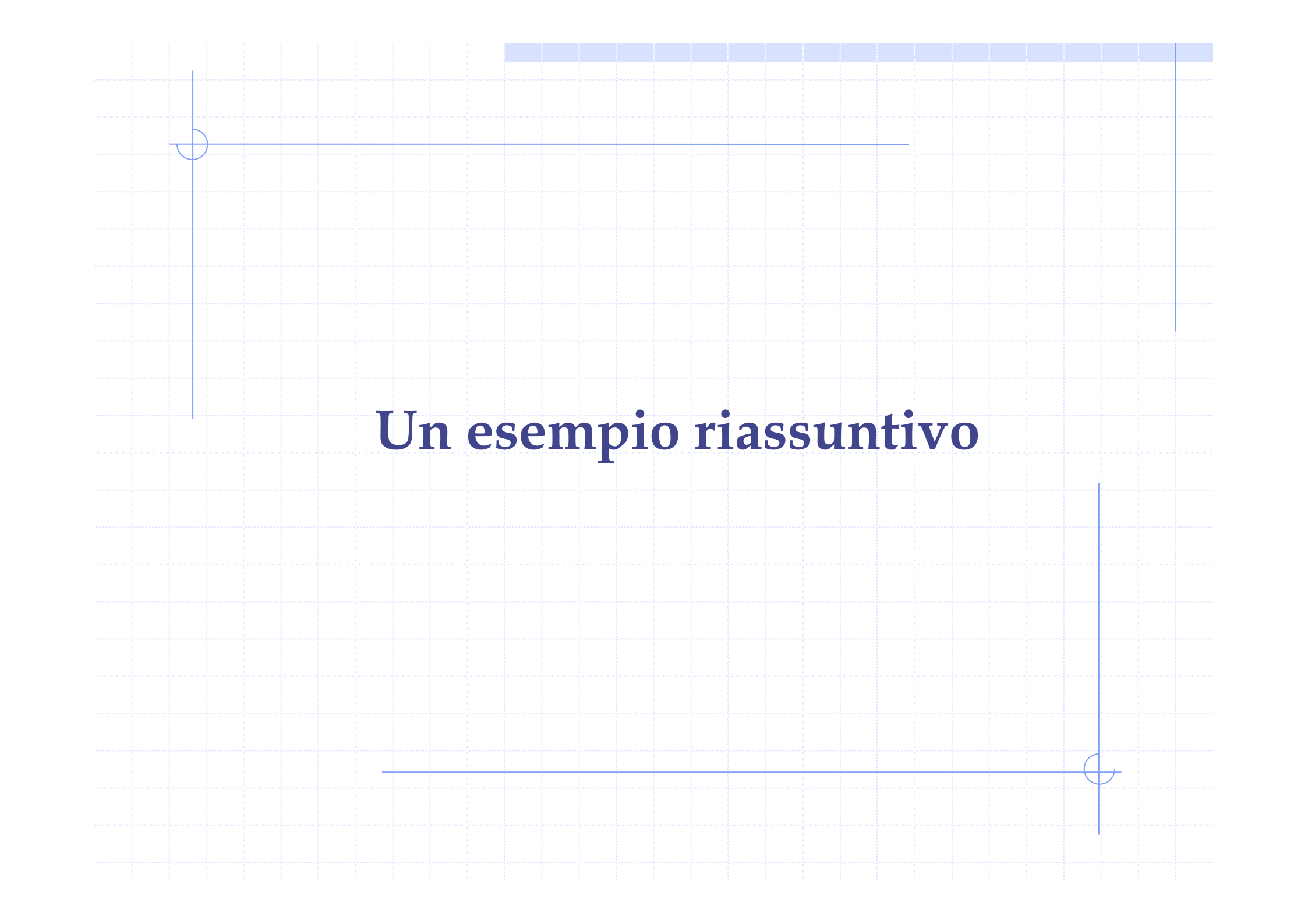
Perchè il main è "static"?

```
public class A {  
    String s="hello";  
    public static void main(String a[]) {  
        System.out.println(s);  
    }  
}
```

Non static variable s cannot be referenced from static context

```
public class A {  
    String s="hello";  
    public static void main(String a[]) {  
        new A();  
    }  
    A() {  
        System.out.println(s);  
    }  
}
```

hello



Un esempio riassuntivo

Esempio: Tombola!



Tombola!

	17	22		45		66		83
1	19		30	48			70	
		29	37		53		74	86
		58	31		23		14	89

Tombola

Il croupier(banco) ha a disposizione un tabellone sul quale sono riportati tutti i numeri da 1 a 90, e un sacchetto riempito con pezzi numerati in modo analogo.

Il suo compito consiste nell'estrarre i pezzi in modo casuale, e annunciare agli altri giocatori il numero uscito.

I giocatori dispongono di una o più cartelle precedentemente acquistate, composte da 3 righe, su ciascuna delle quali sono riportati cinque numeri compresi tra 1 e 90.

Ogni volta che il numero estratto è presente su una o più delle sue schede, il giocatore "copre" la casella corrispondente.

Lo scopo ultimo del gioco è quello di realizzare la tombola, ovvero arrivare per primi a coprire tutti i numeri presenti su una delle proprie cartelle.

Estratto da <https://it.wikipedia.org/wiki/Tombola>

Tombola

Il **croupier(banco)** ha a disposizione un **tabellone** sul quale sono riportati tutti i numeri da 1 a 90, e un **sacchetto** riempito con **pezzi** numerati in modo analogo.

Il suo compito consiste nell'**estrarre** i pezzi in modo casuale, e **annunciare** agli altri giocatori il **numero uscito**.

I **giocatori** dispongono di una o più **cartelle** precedentemente acquistate, composte da 3 **righe**, su ciascuna delle quali sono riportati cinque numeri compresi tra 1 e 90.

Ogni volta che il numero estratto è presente su una o più delle sue schede, il giocatore "**copre**" la casella corrispondente.

Lo scopo ultimo del gioco è quello di realizzare la tombola, ovvero **arrivare per primi a coprire** tutti i numeri presenti su una delle proprie cartelle.

Estratto da <https://it.wikipedia.org/wiki/Tombola>

Diagramma ad oggetti...



Diagramma ad oggetti...



Random numbers

```
package java.util;
```

```
Random(long seed)
```

Creates a new random number generator using a single long seed

```
public int nextInt(int n)
```

Returns a pseudorandom, uniformly distributed int value between **0** (**inclusive**) and the **specified value** (**exclusive**), drawn from this random number generator's sequence.

Common

```
package tombola;  
import java.util.Random;  
public class Common {  
    static final int NCELLS=3;  
    static final int MAXNUM=10;  
    static final Random generatore =  
        new Random(System.currentTimeMillis());  
}
```

Banco

```
package tombola;
```

```
import java.util.LinkedList;  
import java.util.List;
```

```
public class Banco {  
    List sacchetto;
```

```
    public Banco() {  
        sacchetto= new LinkedList();  
        for (int i=1; i<=Common.MAXNUM;i++) {  
            sacchetto.add(new Integer(i));  
        }  
    }  
}
```

Banco

```
public int getNextNumber() {  
    if (sacchetto.size()==0) {  
        System.out.println("NUMERI FINITI!");  
        System.exit(1);  
    }  
    int index=Common.generatore.nextInt(sacchetto.size());  
    Integer num=(Integer)sacchetto.get(index);  
    sacchetto.remove(index);  
    System.out.println("====> ESTRATTO: "+num );  
    return num.intValue();  
}
```

Banco – unit test

```
public static void main(String[] args) {  
    Banco banco = new Banco();  
    while (true) {  
        banco.getNextNumber();  
    }  
}
```

run:

```
====> ESTRATTO: 8  
====> ESTRATTO: 2  
====> ESTRATTO: 6  
====> ESTRATTO: 3  
====> ESTRATTO: 7  
====> ESTRATTO: 9  
====> ESTRATTO: 4  
====> ESTRATTO: 5  
====> ESTRATTO: 10  
====> ESTRATTO: 1
```

NUMERI FINITI!

Java Result: 1

```
package tombola;
```

Cartella

```
import java.util.HashSet;  
import java.util.Iterator;
```

```
public class Cartella {
```

```
    private HashSet numeri = new HashSet();  
    private HashSet mancanti = new HashSet();  
    private Giocatore proprietario=null;  
    private int id=0;  
    static int nCartelle=0;
```

```
    Cartella(Giocatore g) {
```

```
        id=++nCartelle;
```

```
        proprietario=g;
```

```
        for (int i = 1; i <= Common.NCELLS; i++) {
```

```
            boolean creatoNuovoNumero = false;
```

```
            do {
```

```
                int x = Common.generatore.nextInt(Common.MAXNUM)+1;
```

```
                creatoNuovoNumero = numeri.add(new Integer(x));
```

```
                if (creatoNuovoNumero) System.out.println("aggiunto "+ x);
```

```
            } while (!creatoNuovoNumero);
```

```
        }
```

```
        mancanti.addAll(numeri);
```

```
    }
```

Cartella

```
public boolean checkNumber(int x) {  
    boolean result = mancanti.remove(new Integer(x));  
    if(proprietario!=null) {  
        if (result) proprietario.annunciaNumero(x, id);  
        if (mancanti.isEmpty()) proprietario.annunciaVittoria(id);  
    }  
    return result;  
}
```

```
private void print(HashSet list) {  
    Iterator iter = list.iterator();  
    while (iter.hasNext()) {  
        System.out.print(iter.next()+" ");  
    }  
    System.out.println();  
}
```

```
public void printOriginale() {print(neri);}  
public void printCurrent() {print(mancanti);}
```

Cartella – unit test

```
public static void main(String a[]) {  
    Cartella x=new Cartella(null);  
    x.printCurrent();  
    while (!x.mancanti.isEmpty()) {  
        int k=Common.generatore.nextInt(Common.MAXNUM)+1;  
        if (x.checkNumber(k)) System.out.println("==> Trovato "+k);  
        else System.out.println(k);  
    }  
    System.out.println("Finito!");  
    x.printOriginale();  
}
```

aggiunto 2	4	...
aggiunto 7	...	1
aggiunto 9	6	==> Trovato 2
2 7 9	==> Trovato 9	Finito!
==> Trovato 7	7	2 7 9

Player

```
package tombola;  
public class Giocatore {  
    public String name;  
    private Cartella cartella;  
    Giocatore(String name){  
        this.name=name;  
        cartella=new Cartella(this);  
    }  
    void checkNumber(int x){  
        cartella.checkNumber(x);  
    }  
}
```

Player

```
void annunciaNumero(int num, int cartellaId){  
    System.out.println(name+" ha il numero  
"+num+" in cartella "+cartellaId);  
}
```

```
void annunciaVittoria(int cartellaId) {  
    System.out.println(name+" ha vinto con  
cartella "+cartellaId);  
    cartella.printOriginale();  
    System.exit(1);  
}
```

```
}
```

Tombola

```
package tombola;

public class Tombola {
    public Tombola() {
        Banco banco = new Banco();

        Giocatore p = new Giocatore("Pippo");
        while (true) {
            int x = banco.getNextNumber();
            System.out.println(
                "Il numero estratto é " + x);
            p.checkNumber(x);
        }
    }

    public static void main(String[] args) {
        Tombola x=new Tombola();
    }
}
```

```
aggiunto 5
aggiunto 4
aggiunto 1
====> ESTRATTO: 1
Il numero estratto é 1
Pippo ha il numero 1 in cartella 1
====> ESTRATTO: 5
Il numero estratto é 5
Pippo ha il numero 5 in cartella 1
====> ESTRATTO: 7
Il numero estratto é 7
====> ESTRATTO: 10
Il numero estratto é 10
====> ESTRATTO: 2
Il numero estratto é 2
====> ESTRATTO: 4
Il numero estratto é 4
Pippo ha il numero 4 in cartella 1
Pippo ha vinto con cartella 1
1 4 5
```

Tombola - esercizio

Esercizio:

Modificare il codice aggiungendo un numero
arbitrario di giocatori, ciascuno con un numero
arbitrario di cartelle

Tombola - esercizio

Esercizio:

Aggiungere i commenti alla Tombola