

Miglioramenti Sul Codice

1. Separazione delle varie operazioni, ciascuna in un suo metodo. Ciò permette di verificare la correttezza delle operazioni e di riusarle
2. Costruzione di astrazioni, via via più utili, ovvero metodi di livello più basso e metodi di livello più alto, che richiamano i precedenti, lo stesso si fa per le classi
3. Separazione delle operazioni che aggiornano lo stato (command) da quelle che restituiscono valori (query), in metodi diversi. Query sono operazioni che ottengono informazioni (accedendo a dati o facendo calcoli sui dati). Command sono operazioni che effettuano cambiamenti
4. Lo stato dell'oggetto diventa osservabile, attraverso metodi opportuni, in modo da poter fare i test

1

Prof. Tramontana - Marzo 2019

```
// continua
public void testLeggiFile() {
    try {
        pgm.leggiFile("csvfiles", "Importi.csv");
        System.out.println("OK test leggi file");
    } catch (IOException e) {
        System.out.println("FAILED test leggi file");
    }
}

public static void main(String[] args) {
    TestPagamenti tl = new TestPagamenti();
    tl.testLeggiFile();
    tl.testSommaValori();
    tl.testMaxValore();
}
```

```
Output dell'esecuzione di TestPagamenti quando il file Importi.csv
è presente nella cartella csv (dentro la cartella con gli eseguibili)
OK test leggi file
OK test somma val
OK test massimo val

Esecuzione quando il file non viene trovato
FAILED test leggi file
OK test somma val
OK test massimo val
```

3

Prof. Tramontana - Marzo 2019

```
public class TestPagamenti { // per classe Pagamenti vers 0.9
    private Pagamenti pgm = new Pagamenti();

    private void initLista() {
        pgm.svuota();
        pgm.inserisci("321.01");
        pgm.inserisci("531.7");
        pgm.inserisci("1234.5");
    }

    public void testSommaValori() {
        initLista();
        pgm.calcolaSomma();
        if (pgm.getSomma() == 2087.21f)
            System.out.println("OK test somma val");
        else System.out.println("FAILED test somma val");
    }

    public void testMaxValore() {
        initLista();
        pgm.calcolaMassimo();
        if (Math.abs(pgm.getMassimo() - 1234.5f) < 0.01)
            System.out.println("OK test massimo val");
        else System.out.println("FAILED test massimo val");
    }
}
// continua ...
```

2

Prof. Tramontana - Marzo 2019

```
public class Pagamenti { // Pagamenti vers 1.2
    private List<Float> importi = new ArrayList<>();
    public void leggiFile(String c, String n) throws IOException {
        LineNumberReader f = new LineNumberReader(new FileReader(new File(c, n)));
        String riga;
        while (true) {
            riga = f.readLine();
            if (riga == null) break;
            inserisci(Float.parseFloat(riga));
        }
        f.close();
    }
    public void inserisci(float x) {
        if (!importi.contains(x)) importi.add(x);
    }
    public float calcolaSomma() {
        float risultato = 0;
        for (float v : importi) risultato += v;
        return risultato;
    }
    public float calcolaMassimo() {
        float risultato = 0;
        for (float v : importi)
            if (risultato < v) risultato = v;
        return risultato;
    }
    public void svuota() {
        importi = new ArrayList<>();
    }
}
```

4

Prof. Tramontana - Marzo 2019

Pagamenti
- importi: List<Float>
+ leggiFile(c: String, n: String)
+ inserisci(x: float)
+ calcolaSomma(): float
+ calcolaMassimo(): float
+ svuota()

```

public class TestPagamenti { // per classe Pagamenti vers 1.2
    private Pagamenti pgm = new Pagamenti();
    private void initLista() {
        pgm.svuota();
        pgm.inserisci(321.01f);
        pgm.inserisci(531.7f);
        pgm.inserisci(1234.5f);
    }
    public void testSommaValori() {
        initLista();
        if (pgm.calcolaSomma() == 2087.21f) System.out.println("OK test somma val");
        else System.out.println("FAILED test somma val");
    }

    public void testListaVuota() {
        pgm.svuota();
        if (pgm.calcolaSomma() == 0) System.out.println("OK test somma val empty");
        else System.out.println("FAILED test somma val empty");
        if (pgm.calcolaMassimo() == 0) System.out.println("OK test massimo val empty");
        else System.out.println("FAILED test massimo val empty");
    }
}

```

5

Prof. Tramontana - Marzo 2019

Ulteriori Risorse

- Per i tool di sviluppo e il runtime Java: www.java.com
- Per compilare una classe Java da shell dei comandi
 - javac NomeClasse.java, es. javac Pagamenti.java
- Se la compilazione va a buon fine non viene dato nessun messaggio e viene generato un file NomeClasse.class
- Per eseguire un programma Java
 - java NomeClasse, es. java Pagamenti
- La classe che si dà come argomento del comando java deve avere il metodo main (public static void main(String[] args) { ... }), che è il metodo da cui comincia l'esecuzione del codice
- La classe Java specifica con import dove trovare il tipo che fa parte della libreria Java ed è usato all'interno della classe (per es. List, File), es.


```
import java.io.File, inserito all'inizio del file NomeClasse.java
```

7

Prof. Tramontana - Marzo 2019

Punti Importanti Da Ricordare

- Correttezza: è stato possibile eseguire test che verificano il codice. Soddisfare (e verificare) i requisiti permette di ottenere la qualità del sistema
 - I test documentano le condizioni sotto le quali il codice funziona e protegge il codice da modifiche indesiderate. I test devono essere indipendenti fra loro ed auto valutarsi
 - Test Driven Development (TDD): scrivere prima il test e poi il codice che risolve un requisito. Lo sviluppatore sceglie nomi di classi e metodi (progettazione)
- Responsabilità: i compiti sono suddivisi su vari metodi (e quindi su più classi), questo permette di ottenere coesione del codice
 - Principio di Singola Responsabilità. La singola responsabilità è cruciale per la comprensione, il riuso, l'ereditarietà (OOP)
- Astrazioni: lo sviluppatore OOP costruisce astrazioni. Il nome delle astrazioni è estremamente importante: il nome che si dà a classi e metodi ne descrive l'obiettivo

6

Prof. Tramontana - Marzo 2019

Conclusioni

- Key points
 - Test driven development
 - Single Responsibility Principle
- Esempi di domande d'esame
 - Implementare un test per un metodo che prende in ingresso un intero
 - Dire come si compila una classe in Java
 - Implementare una classe Java che può essere data alla JVM per essere eseguita

8

Prof. Tramontana - Marzo 2019