



ESERCIZI / DOMANDE

Fondamenti IS – ‘simil’ esame

Ingegneria del software 2023-2024

AULAWEB – ESAME IS

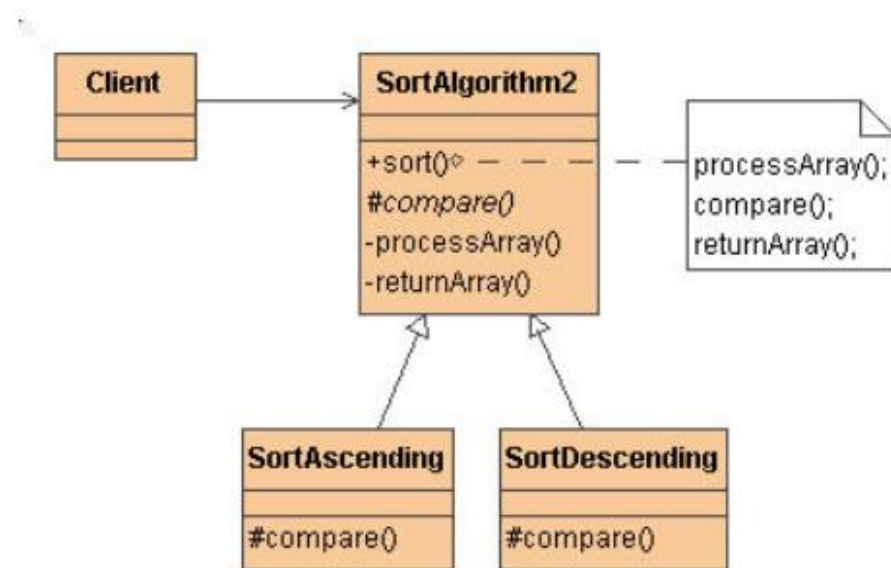
- La **prova scritta** sarà articolata **in due parti**:
 - **Prima parte**: costituita da 10 domande a risposta multipla, che costituisce **sbarramento** per la partecipazione alla parte successiva. Le domande saranno sia di tipo teorico che pratico e verteranno sugli argomenti visti a lezione
 - Almeno 7 su 10 corrette
 - **Seconda parte**: costituita da una **domanda teorica aperta** e da un **esercizio** che dovranno necessariamente essere svolti, ognuno, su un'unica facciata di foglio A4. Domanda e esercizio verteranno sugli argomenti visti a lezione



QUIZ - DOMANDE A RISPOSTA MULTIPLA

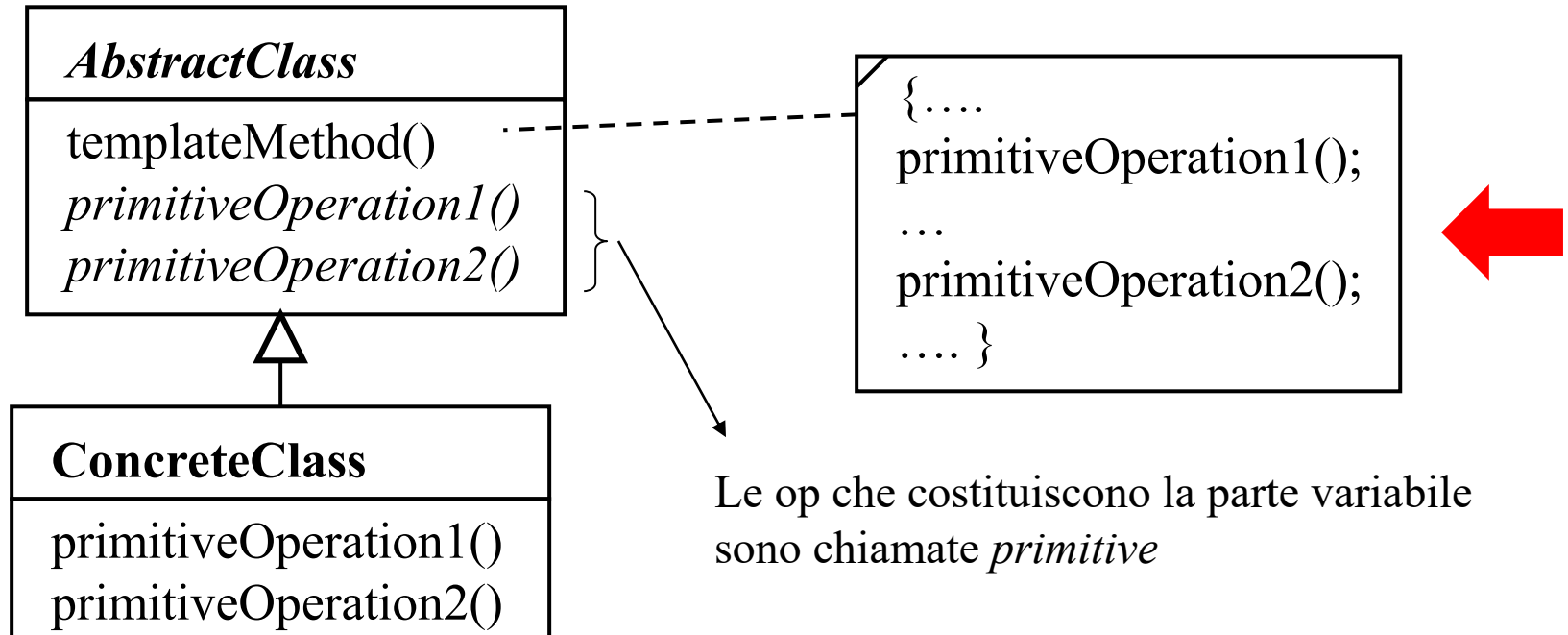
DOMANDA 1

Quale tra i seguenti **design pattern** ha ispirato il diagramma sottostante?



- a) Composite
- b) Template Method
- c) Adapter
- d) State

TEMPLATE METHOD



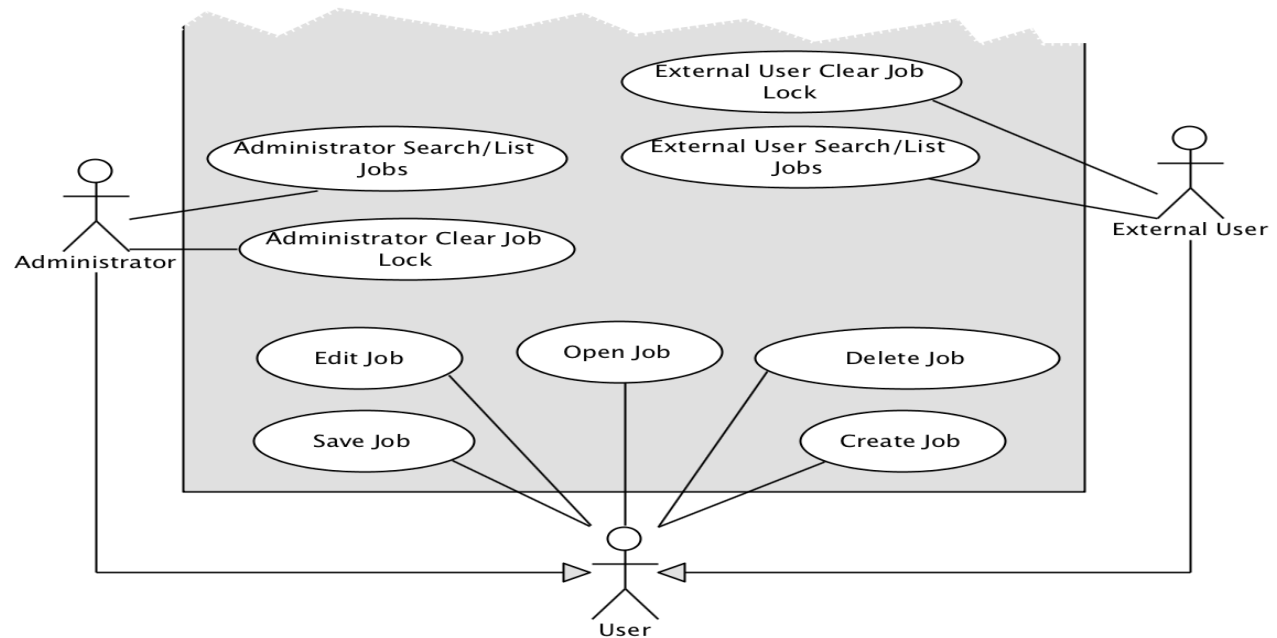
Hollywood principle (inversione di controllo)

Non chiamate, richiameremo noi ...

Normalmente sono le sottoclassi a chiamare i metodi delle superclassi. Con questo pattern è `templateMethod()` a chiamare i metodi specifici ridefiniti nelle sottoclassi

DOMANDA 2

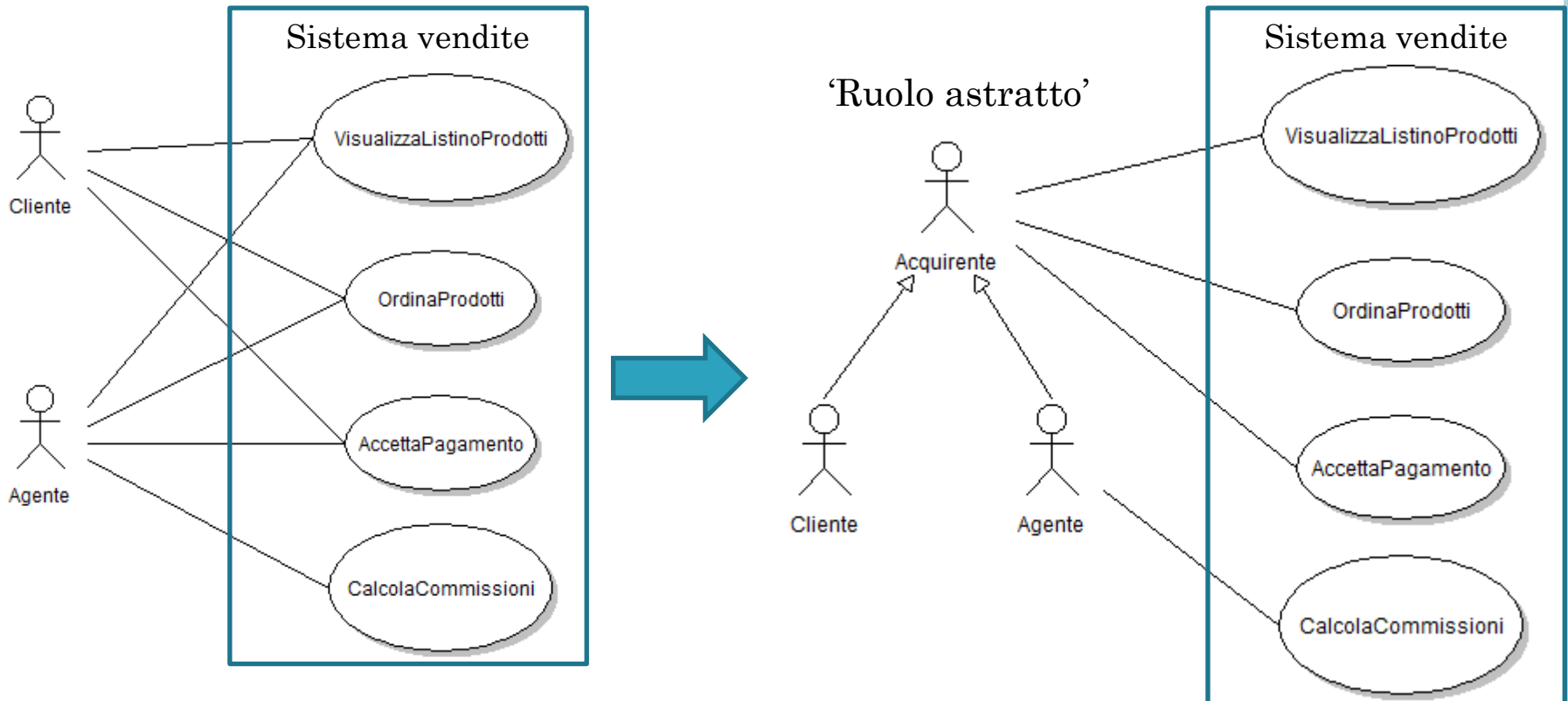
- Considerando lo Use case diagram rappresentato sotto quale tra gli attori può invocare la funzionalità “Open Job”?



- a) User
- b) User, Administrator, External User**
- c) Job, Administrator, External User
- d) Administrator, External User

GERARCHIE DI ATTORI

È possibile definire **gerarchie di attori**



DOMANDA 3

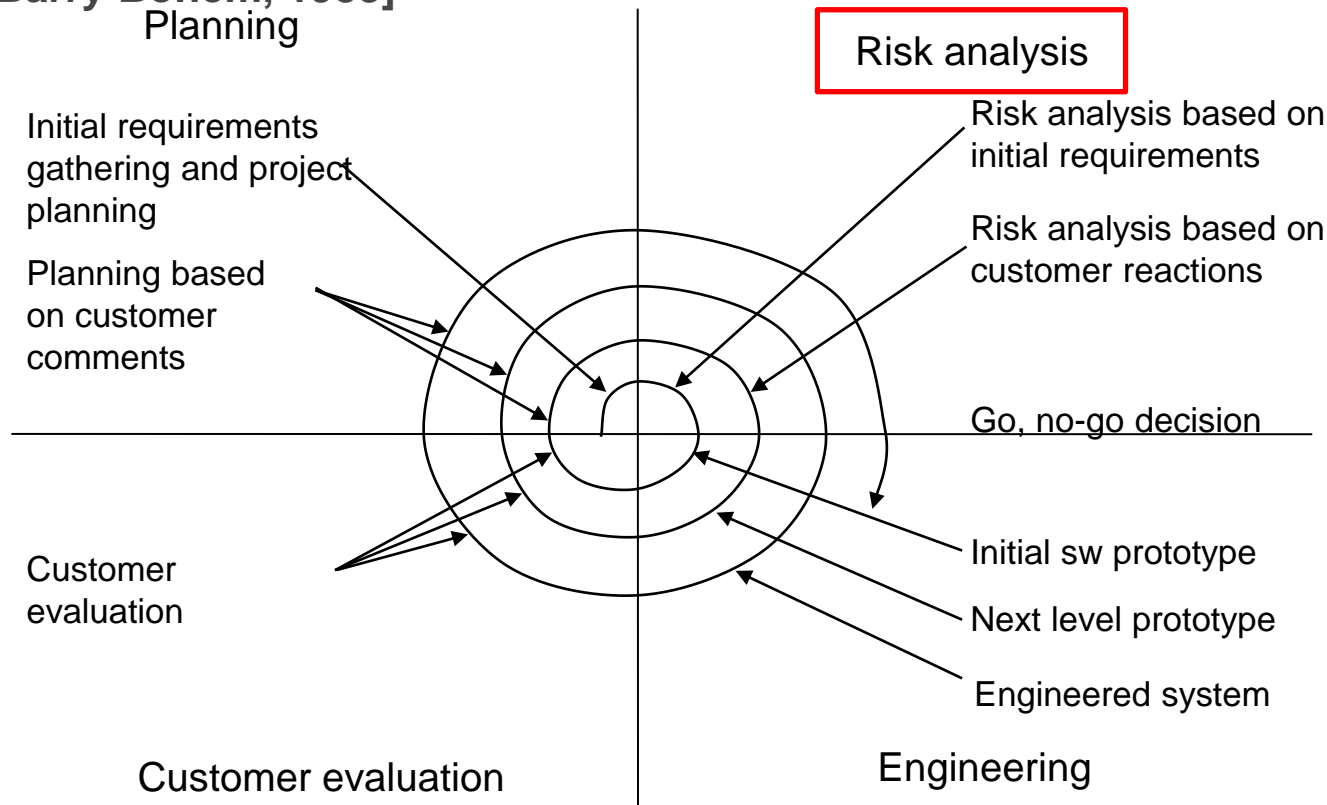
- Qual è stato il primo modello di processo di sviluppo a considerare esplicitamente il concetto di **rischio**?
- a) Modello a cascata
- b) V model
- c) Modello a spirale
- d) RUP

Modello a Spirale

[Barry Bohem, 1988]

Planning

Quattro fasi ben distinte



- **Planning:** determinazione di obiettivi, alternative, vincoli
- **Risk analysis:** analisi delle alternative e identificazione/risoluzione dei rischi
- **Engineering:** sviluppo del prodotto di successivo livello
- **Customer evaluation:** valutazione dei risultati dell'engineering dal punto di vista del cliente

MODELLO A SPIRALE

Vantaggi

- Adatto allo sviluppo di sistemi complessi
- **Primo modello che considera il rischio**
 - Modello guidato dal rischio



Svantaggi


- Non è una panacea ...
- Necessita di competenze di alto livello per la stima dei rischi
- Richiede un'opportuna personalizzazione
- Se un rischio rilevante non viene scoperto o tenuto a bada siamo da capo ...



DOMANDA 4

- Una componente (o modulo) che modifica dati interni ad un'altra componente (o modulo) è un esempio di:
 - a) Accoppiamento forte da evitare
 - b) Accoppiamento debole
 - c) Bassa coesione tra le due componenti
 - d) Accoppiamento forte accettabile

ACCOPPIAMENTO «CATTIVO»

- **Content:** un modulo riesce a
 - **modificare dati interni ad un altro modulo** 
 - “saltare” all’interno di un altro modulo oppure
 - alterare uno statement in un altro modulo
- Va evitato perchè complica **enormemente** la comprensione e la modifica
- In un sistema OO si riduce *incapsulando* tutte le variabili d’istanza (inoltre non esiste GOTO)
 - dichiarandole private
 - fornendo i metodi di get e set
- Nei sistemi Legacy (es. Fortran, Cobol, C, ...) le cose sono più complicate: GOTO e puntatori

DOMANDA 5

- Nel design by contract un **invariante di classe** è una condizione che ogni oggetto della classe deve soddisfare:
 - a) Sempre, in ogni possibile momento
 - b) In ogni momento in cui è possibile eseguire un metodo sull'oggetto
 - c) Durante l'esecuzione di un metodo pubblico, se è soddisfatta la preconditione del metodo
 - d) Durante l'esecuzione di un metodo privato, se è soddisfatta la preconditione del metodo

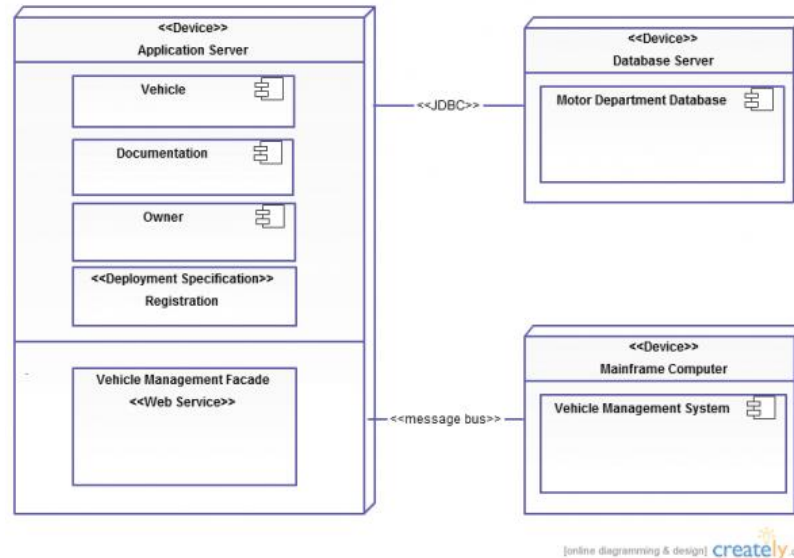
ELEMENTI DI UN CONTRATTI (CLASSI OO)

- **Pre-condizione.** Espressione a valori booleani rappresentante **le aspettative** sullo «stato del mondo» prima che venga eseguita un'operazione
 - Es. radice quadrata: `int sqrt(int x)`
pre: $x \geq 0$
- **Post-condizione.** Espressione a valori booleani riguardante lo «stato del mondo» dopo l'esecuzione di un'operazione.
 - Es. radice quadrata: `int sqrt(int x)`
pre: $x \geq 0$
post: $x = \text{risultato} * \text{risultato}$
- **Invariante di classe.** Condizione che ogni oggetto della classe deve soddisfare quando è «**in equilibrio**»
 - **In equilibrio:** non «in mezzo» a una transizione in ogni momento in cui è possibile eseguire un'operazione



DOMANDA 6

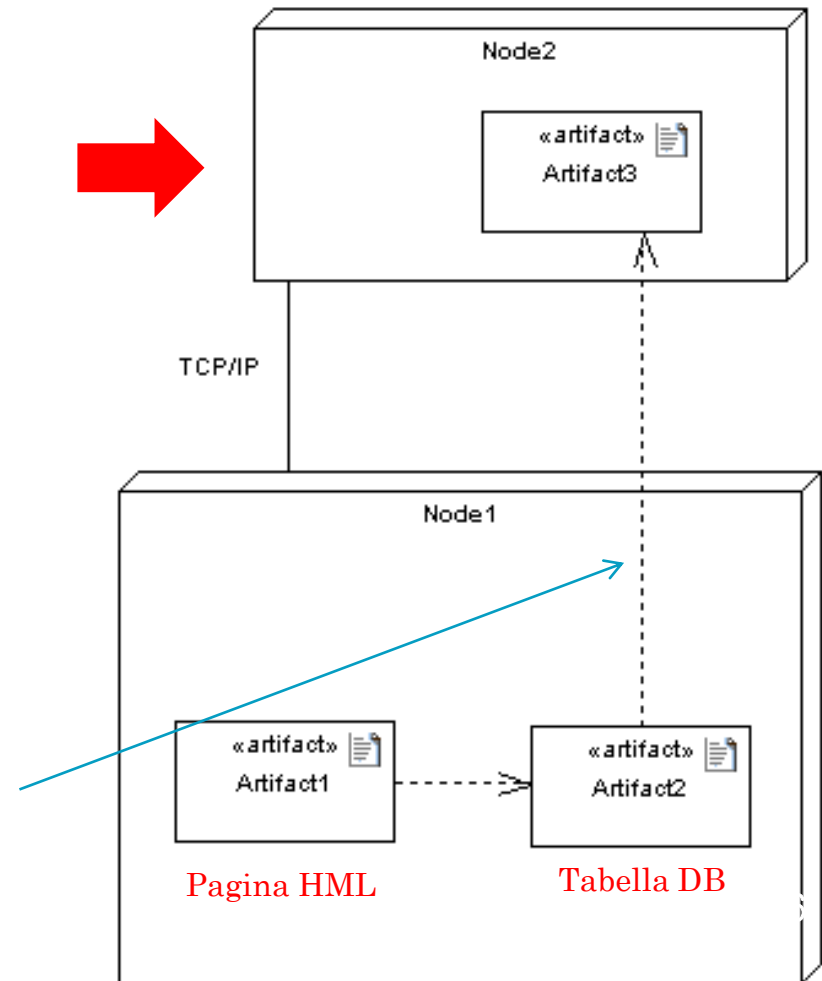
- Dato il seguente diagramma quale tra le seguenti affermazione è vera?



- La figura rappresenta un **deployment diagram** UML corretto
- La figura rappresenta un **component diagram** UML corretto
- La figura rappresenta un **deployment diagram** UML non corretto. All'interno dei nodi occorre posizionare gli artefatti e non i componenti
- La figura rappresenta un **component diagram** UML non corretto. All'interno dei nodi occorre posizionare gli artefatti e non i componenti

ARTEFATTI

- Sono **entità concrete** del mondo reale
- Per esempio:
 - file di codice sorgente, file eseguibili, script, tabelle in un database, pagine HTML ...
- Possono essere dislocati sui nodi
- Esiste la relazione di dipendenza tra manufatti

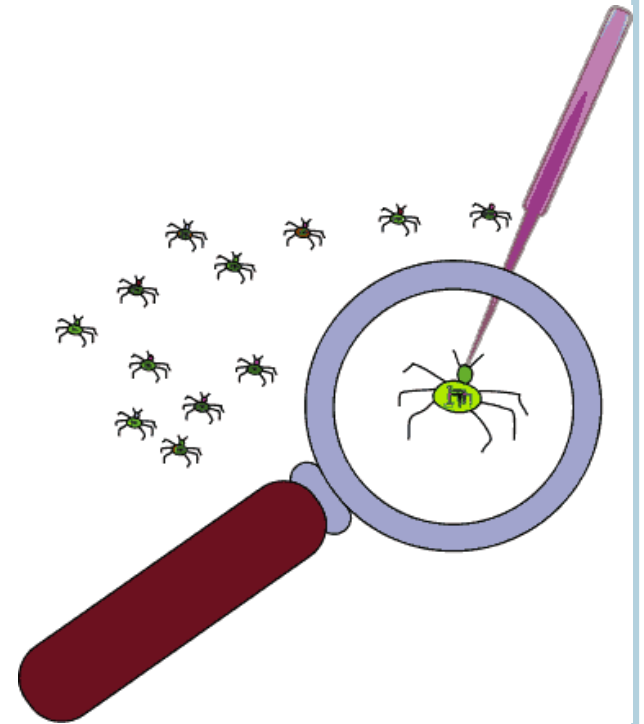


DOMANDA 7

- Quale delle seguenti affermazioni **identifica meglio** le differenze tra **software testing** e fase di **debugging**?
- a) Il testing rivela i failure. Il debugging invece è costituito da due fasi: (i) localizzazione del file/componente che contiene il fault e (ii) rimozione del fault
- b) Il testing dinamico previene gli errori degli sviluppatori. Il debugging invece trova, analizza e rimuove le cause dei failure nel software
- c) Il testing rimuove i fault. Il debugging identifica le cause dei failure
- d) Testing e debugging sono due fasi correlate e molto importanti. Il debugging segue sempre la fase di testing

DEBUGGING

- Software testing rivela i failure ←
 - ovvero un comportamento anomalo del SUT
- **Debugging**: il processo usato per trovare un bug/fault e rimuoverlo
- Due fasi: ←
 - fault localization/location
 - es. trovare il file che contiene il fault
 - rimozione del fault





DOMANDA 8

- Quale è la differenza tra un **framework** e una **libreria software tradizionale**? (almeno come visto a lezione)
- a) I framework sono meno specializzati delle librerie, cioè non sono relativi ad un dominio applicativo specifico
- b) Il primo è più astratto e va implementato a partire da un modello e quindi risulta essere usabile in contesti differenti, la seconda invece è più concreta
- c) Una libreria viene “chiamata” dal programma che la utilizza mentre è il framework a “chiamare” le componenti custom relative allo specifico problema
- d) Un framework viene “chiamato” dal programma che lo utilizza mentre è la libreria software a “chiamare” le componenti custom relative allo specifico problema

FRAMEWORK

◦ Framework vs libreria software:

- **libreria:** il programmatore di un'applicazione chiama le operazioni 
- **framework:** il programmatore di un'applicazione
 - scrive le op che vengono chiamate dalle op del framework 
 - “riempie i buchi”

◦ Design pattern vs framework

- più astratti
- più piccoli (come elementi architettureali)
- meno specializzati, non relativi a dominio applicativo specifico

DOMANDA 9

- Di che tipo è il seguente requisito software: **“La visualizzazione della busta paga deve avvenire entro 10 secondi dalla sua richiesta”?**
- a) Funzionale
- b) Tecnologico
- c) Non funzionale
- d) Desiderabile

REQUISITI FUNZIONALI VS. NON FUNZIONALI

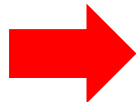
Requisiti funzionali

descrivono le funzionalità ed i servizi forniti dal sistema

- indipendenti dall'implementazione di una soluzione (no come ma cosa)

Es (sistema bancario):

- *il sistema dovrà permettere la consultazione del saldo*
- *il sistema dovrà convertire il saldo (espresso in euro) in dollari a richiesta del cliente*



Requisiti non-funzionali

non sono collegati direttamente con le funzionalità implementate dal sistema, ma piuttosto alle modalità operative, di gestione, ...

- definiscono vincoli sul sistema e sullo sviluppo del sistema
- in generale riguardano la scelta di linguaggi, piattaforme, strumenti (tools) e tecniche d'implementazione

Es:

- *i computer devono essere dei PC IBM*
- *la risposta ad un'interrogazione deve avvenire entro tre secondi*
- *i documenti devono essere registrati in formato PDF*

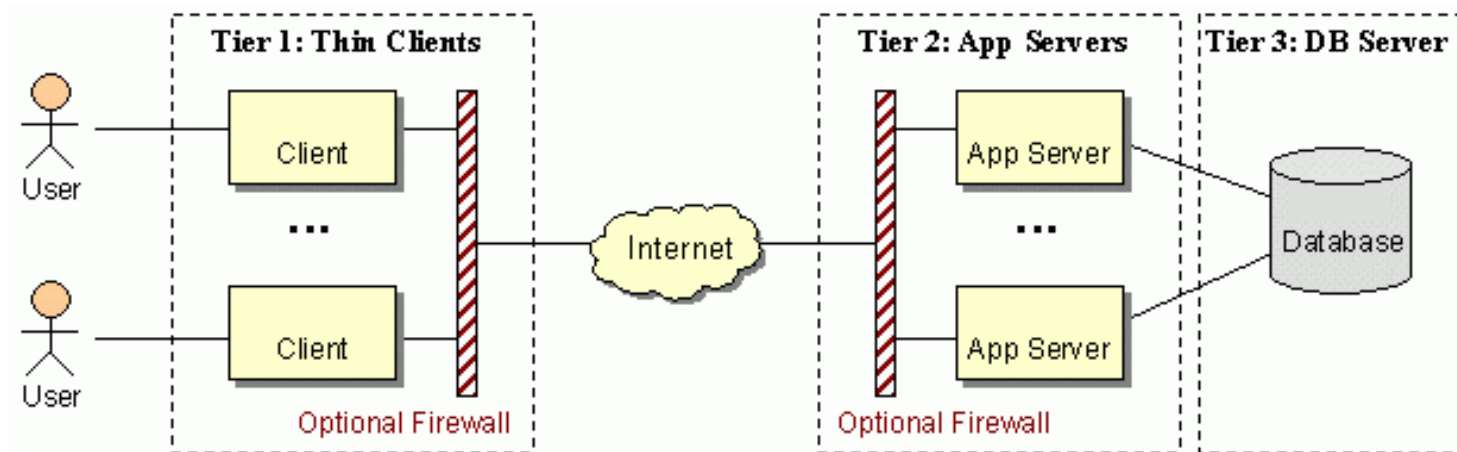
DOMANDA 10

- Si supponga di dover implementare un applicazione Web integrata per la compilazione, gestione e pagamento dei modelli 730/Unico a supporto dell'ordine dei dottori commercialisti.
- L'architettura scelta è **client-server** di tipo **three-tier**. Dove (con molta probabilità) dovranno essere posizionate le componenti eseguibili che realizzano le funzionalità *Calculate taxes* e *Authorize payments*?
 - a) Nello strato client
 - b) Nello strato DB server
 - c) Nello strato application server
 - d) Nello strato server Web

CLIENT-SERVER A TRE LIVELLI

◦ Evoluzione del Two-Tier

- sempre client-server
- sul client resta solo l'interfaccia utente
- la logica del sistema è uno strato separato che gestisce multi-utenti
- gli strati di logica e gestione DB possono essere distribuiti su più server





DOMANDE A RISPOSTA APERTA

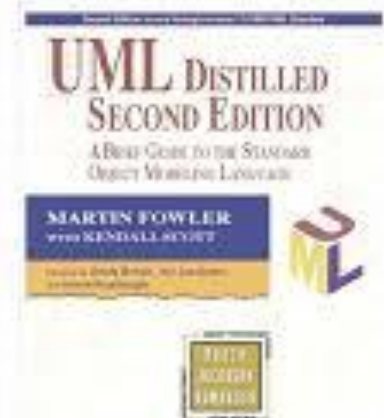
DOMANDA 1

- Descrivere i tre possibili ‘modi di utilizzo’ di UML, in particolare mettendo in risalto le differenze tra loro. Tra questi ‘modi di utilizzo’ quale pensi essere il più indicato per lo sviluppo di **applicazioni Web di commercio elettronico** e perchè?

Approaches to UML

- ◆ Sketch
- ◆ Blueprint
- ◆ Programming

Ref: Martin Fowler,
"UML Distilled"



POSSIBILE RISPOSTA

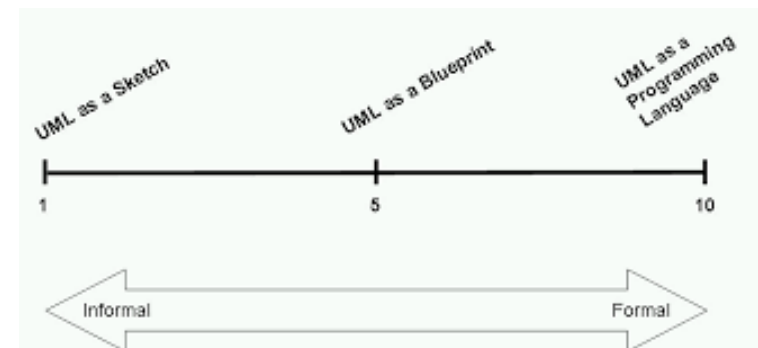
○ Tre 'utilizzi' di UML

- Sketch o Abbozzo
 - Disegno alla lavagna
 - I modelli non si conservano
- Blueprint o dettagliato
 - UML Modeler (e.g., Visual Paradigm)
- Linguaggio di programmazione
 - Concetto di Profilo / MDD (es. WebRatio)

Descrizione solo
abbozzata

○ Sviluppo di app commercio elettronico

- Due opzioni:
 - Sketch o Abbozzo
 - Linguaggio di programmazione
 - Tool WebRatio



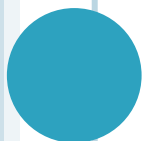
DOMANDA 2

- Spiegare la differenza tra **processo di sviluppo tradizionale**, nel quale la codifica del codice è eseguita dallo sviluppatore in un linguaggio di programmazione (es. Java), e sviluppo denominato **MDD (Model Driven Development)**.
- Nominare e descrivere brevemente un tool MDD tra quelli visti a lezione.

POSSIBILE RISPOSTA

- MDD è un approccio allo sviluppo del SW alternativo alla programmazione tradizionale (**code centric programming**) che si basa sullo sviluppo di un modello e sull'esistenza di un tool capace di eseguirlo o generare codice a partire da questo. Il modello di un sistema software è sviluppato utilizzando una notazione (di solito grafica ma non solo) di design come ad esempio UML (arricchito con un profilo).
- Un esempio di tool MDD è WebRatio che prende in input un modello IFML e lo esegue. Il risultato è una Web (o Mobile) app funzionante.





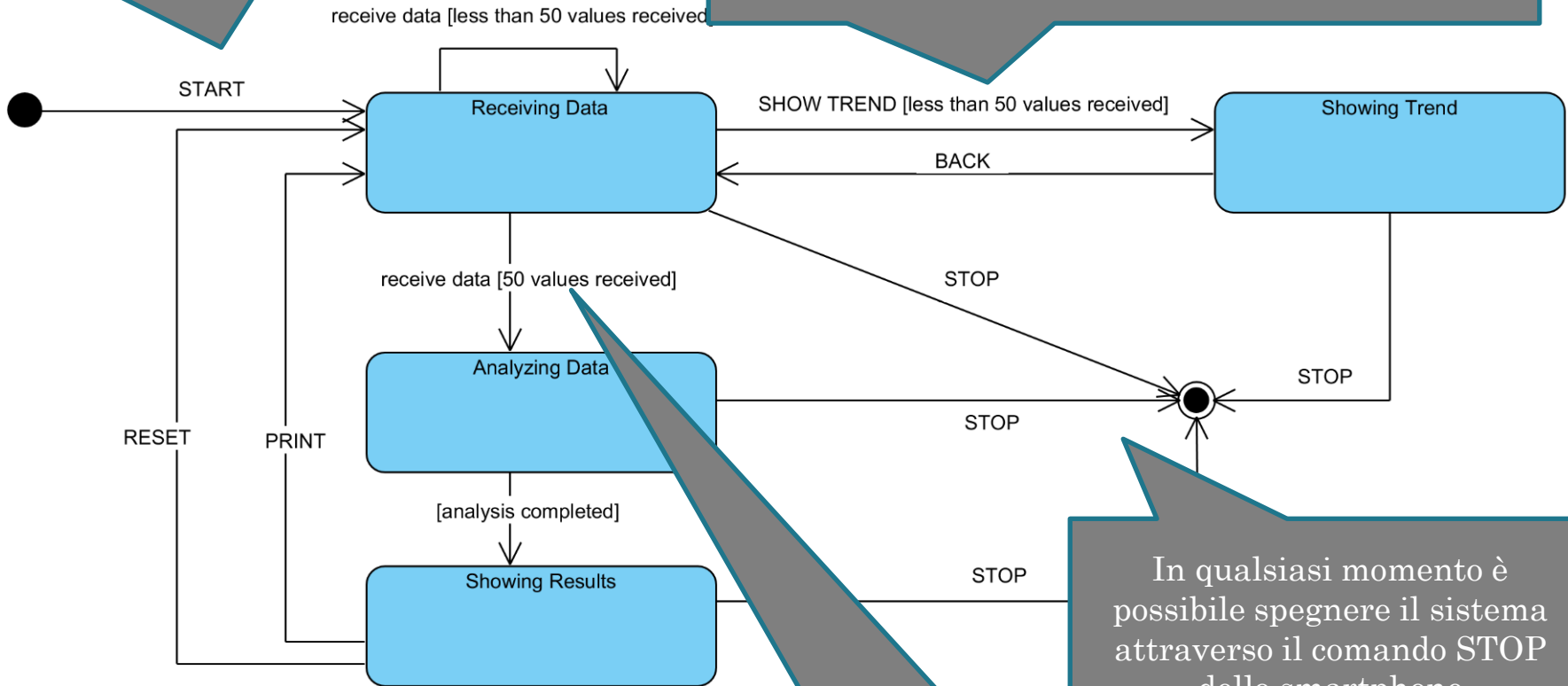
ESERCIZI

ESERCIZIO 1 (STATE MACHINE)

- Modellare, **attraverso un diagramma UML** che trovate adatto, un sistema composto da uno **smartphone** e da un **sensore** così come descritto:
- Il sistema viene avviato attraverso il comando **START** dello smartphone e da quel momento inizia a ricevere dati dal sensore (un valore ogni X secondi).
- Nel momento in cui vengono ricevuti 50 valori, il sistema procede alla loro analisi. Una volta completata l'analisi, il sistema mostra i risultati a video, che successivamente possono essere stampati (e resettati) via comando **PRINT** o solo resettati via comando **RESET** dello smartphone, ripristinando in entrambi i casi il sistema allo stato di ricezione dati dal sensore.
- È anche possibile, ma solo durante la ricezione di dati dal sensore e quando questi sono meno di 50, visualizzare il trend corrente dei dati ricevuti, selezionando il comando **SHOW TREND** dello smartphone. Ciò provoca momentaneamente l'interruzione della ricezione dei dati dal sensore, che è possibile ripristinare attraverso il comando **BACK** dello smartphone.
- In qualsiasi momento è possibile spegnere il sistema attraverso il comando **STOP** dello smartphone.

Il sistema viene avviato attraverso il comando START dello smartphone e da quel momento inizia a ricevere dati dal sensore

È anche possibile, ma solo durante la ricezione di dati dal sensore e quando questi sono meno di 50, visualizzare il trend corrente dei dati ricevuti, selezionando il comando SHOW TREND dello smartphone. Ciò provoca momentaneamente l'interruzione della ricezione dei dati



Una volta completata l'analisi, il sistema mostra i risultati a video, che successivamente possono essere stampati (PRINT) e resettati (RESET)

Nel momento in cui vengono ricevuti 50 valori, il sistema procede alla loro analisi.

In qualsiasi momento è possibile spegnere il sistema attraverso il comando STOP dello smartphone

ESERCIZIO 2 (SEQUENCE DIAGRAM)

- Date le seguenti tre classi **Farmacia**, **Medicina** e **TipoMedicina**

```
public class Farmacia {
    private List<Medicina> medicine = new ArrayList();

    public void inserisciMed(int id, String name, TipoMed tipo, int prezzo) {
        Medicina m = new Medicina();
        m.setId(id);
        m.setName(name);
        m.setPrezzo(prezzo);
        m.setTipoMedicina(tipo);
        add(m);
    }

    private void add(Medicina m) {
        if (!InFarmacia(m))
            medicine.add(m);
    }

    private boolean InFarmacia(Medicina m) {
        for (Iterator iterator = medicine.iterator(); iterator.hasNext();) {
            Medicina ml = (Medicina) iterator.next();
            if (ml.id == m.id)
                return true;
        }
        return false;
    }

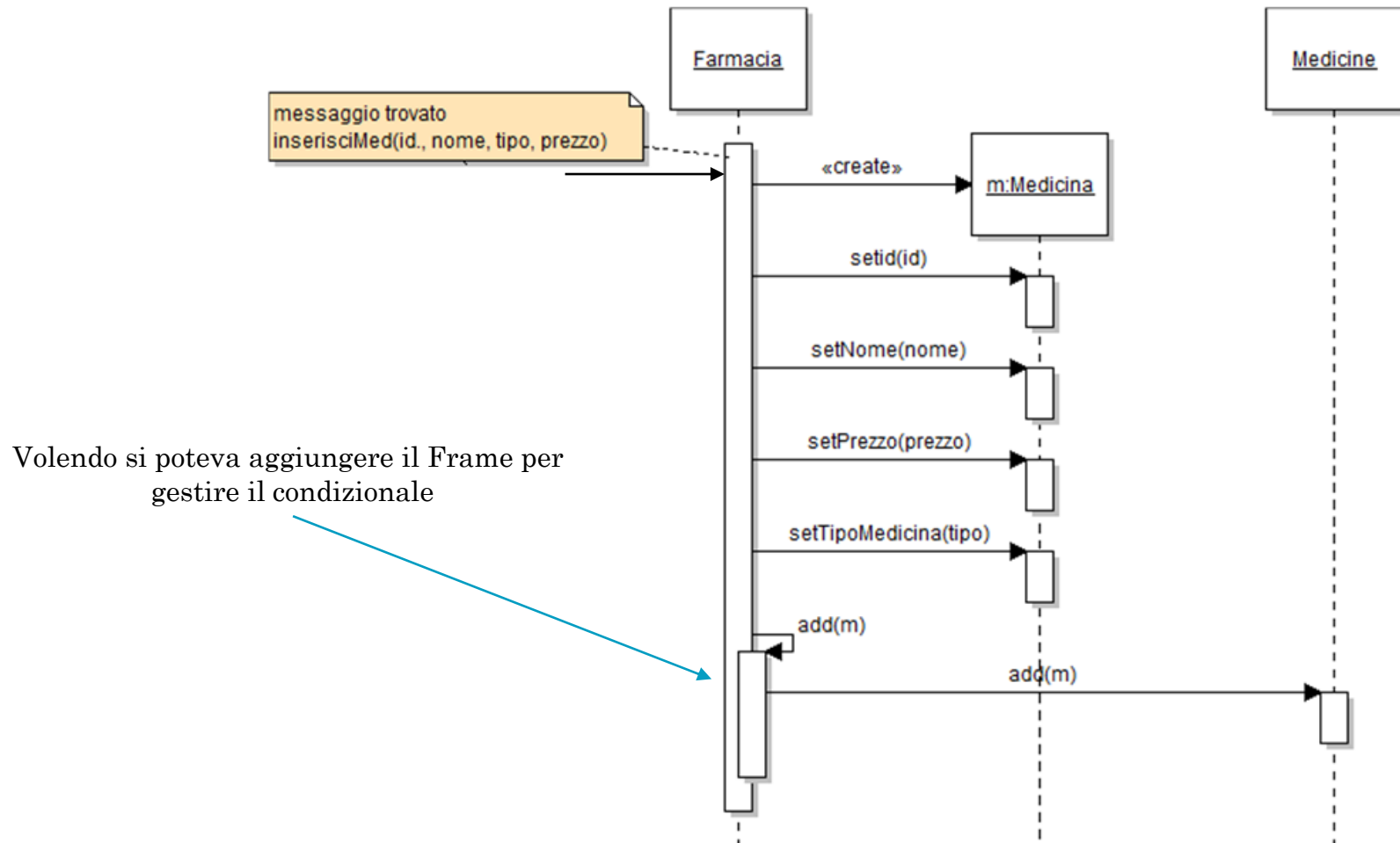
    public String contaMedicinePerTipo() {
        int a=0, b=0, c=0;
        for (Iterator iterator = medicine.iterator(); iterator.hasNext();) {
            Medicina ml = (Medicina) iterator.next();
            switch (ml.tipo) {
                case A:
                    a++;
                    break;
                case B:
                    b++;
                    break;
                case C:
                    c++;
            }
        }
        return a + " " + b + " " + c;
    }
}
```

Domanda: ricavare il sequence diagram relativo all'operazione **inserisciMed()**. Sostanzialmente si tratta di rappresentare tutte le invocazioni dell'operazione **inserisciMed()** mediante un sequence diagram

```
public class Medicina {
    public int id;
    private String name;
    private int prezzo;
    public TipoMedicina tipo;

    // usuali metodi set ....
}

public enum TipoMedicina {
    A, B, C
}
```



```

public void inserisciMed(int id, String name, TipoMed tipo, int prezzo) {
    Medicina m = new Medicina();
    m.setId(id);
    m.setName(name);
    m.setPrezzo(prezzo);
    m.setTipoMedicina(tipo);
    add(m);
}

```

ESERCIZIO 3 (CLASS DIAGRAM)

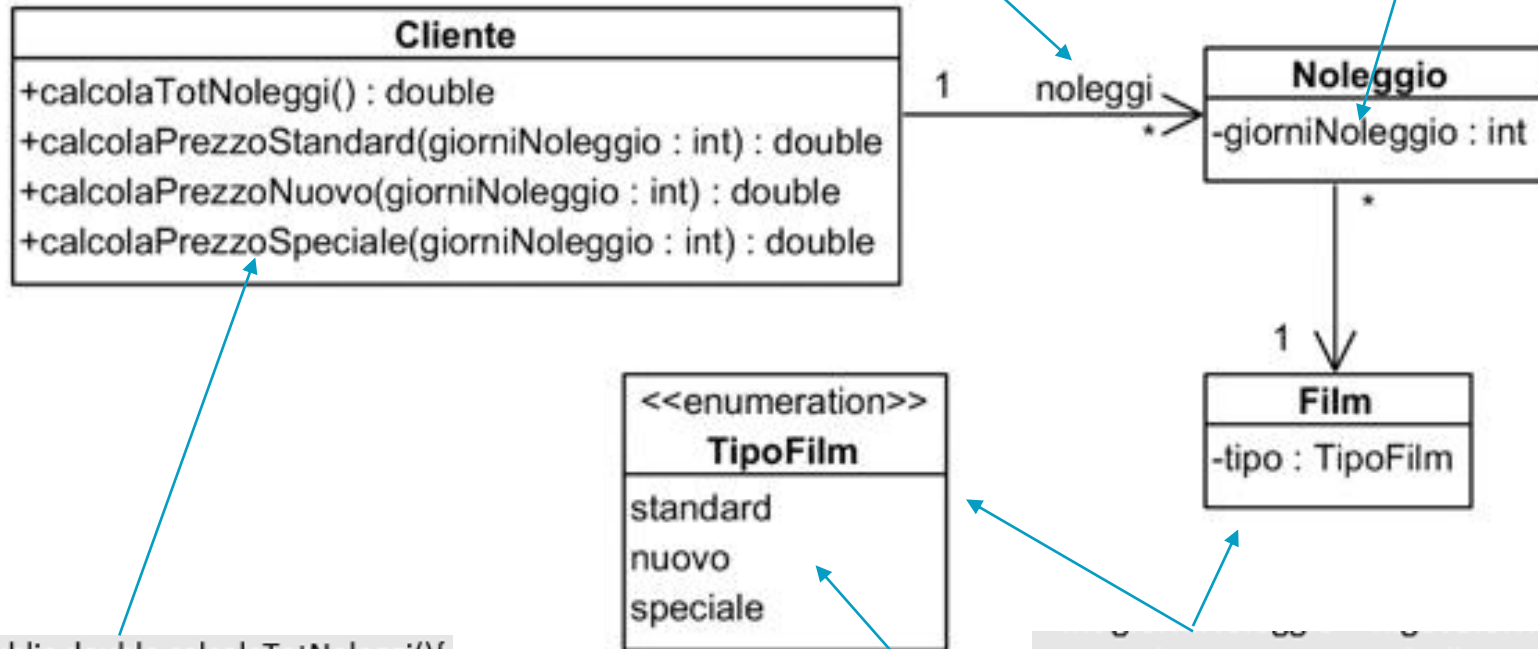
- Dato il codice seguente, **inferire** il class diagram UML corrispondente alla classe **Ciente** e classi associate, indicando con precisione le cardinalità delle associazioni, gli attributi (nome e tipo) e le operazioni (parametri e tipo di ritorno)

```
public class Cliente{
    private List<Noleggior> noleggi;
    public Cliente(List<Noleggior> noleggi) {...}
    public double calcolaTotNoleggi(){
        double tot = 0;
        for(Noleggior n: noleggi){
            int giorniNoleggior = n.getGiorniNoleggior();
            TipoFilm tipo = n.getFilm().getTipoFilm();
            if (tipo == TipoFilm.standard)
                tot += calcolaPrezzoStandard(giorniNoleggior);
            else if (tipo == TipoFilm.nuovo)
                tot += calcolaPrezzoNuovo(giorniNoleggior);
            else
                tot += calcolaPrezzoSpeciale(giorniNoleggior);
        }
        return tot;
    }
    public double calcolaPrezzoStandard(int giorniNoleggior) {...}
    public double calcolaPrezzoNuovo(int giorniNoleggior) {...}
    public double calcolaPrezzoSpeciale(int giorniNoleggior) {...}
}
```

```
int giorniNoleggio = n.getGiorniNoleggio();
```

```
private List<Noleggio> noleggi;
```

Possibile Soluzione:



```
public double calcolaTotNoleggi(){
public double calcolaPrezzoStandard(int giorniNoleggio) {...}
public double calcolaPrezzoNuovo(int giorniNoleggio) {...}
public double calcolaPrezzoSpeciale(int giorniNoleggio) {...}
```

```
TipoFilm tipo = n.getFilm().getTipoFilm();
```

```
if (tipo == TipoFilm.standard)
    tot += calcolaPrezzoStandard(giorniNoleggio);
else if (tipo == TipoFilm.nuovo)
    tot += calcolaPrezzoNuovo(giorniNoleggio);
else if (tipo == TipoFilm.speciale)
    tot += calcolaPrezzoSpeciale(giorniNoleggio);
```

ESERCIZIO 4 (SOFTWARE TESTING)

- Dato il seguente Pseudo-codice
 - Disegnare il CFG
 - Progettare una test suite che abbia **branch coverage 100%**

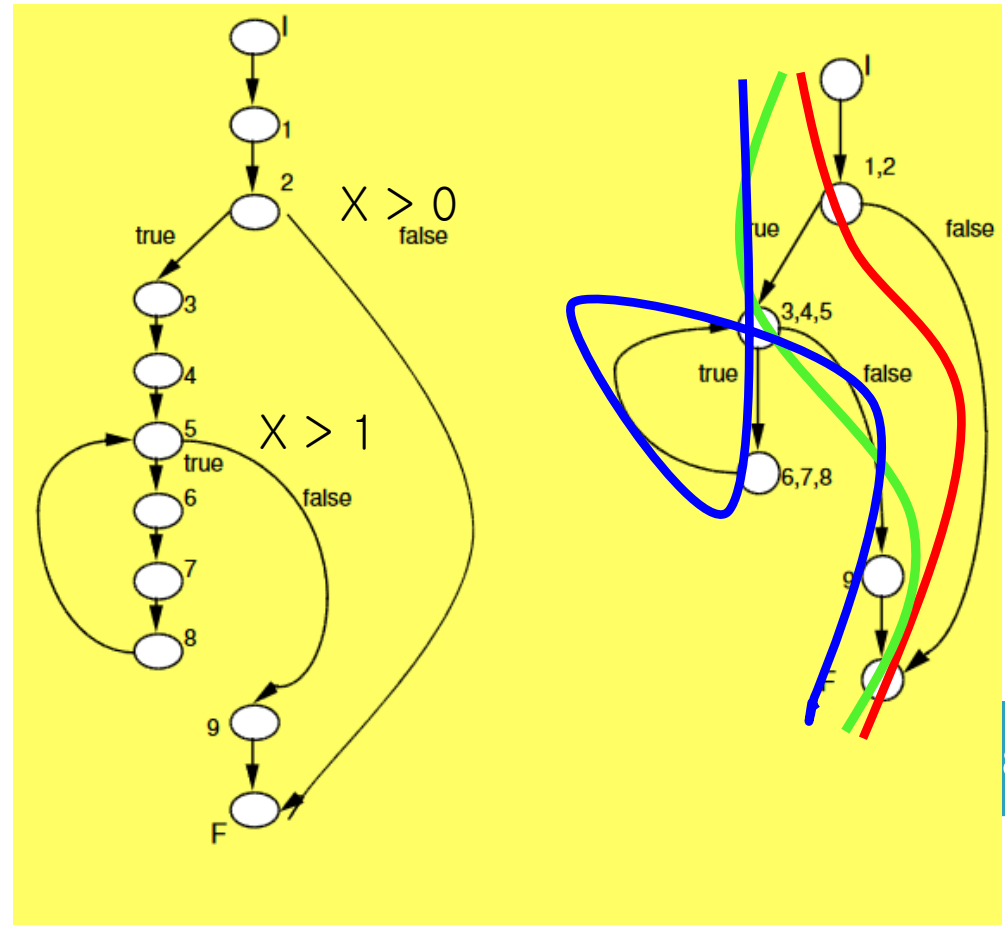
```
procedure Quadrato;  
  var x, y, n: integer;  
  begin  
    1.  read(x);  
    2.  if x > 0  
        then begin  
          3.    n := 1;  
          4.    y := 1;  
          5.    while x > 1 do  
              begin  
                6.    n := n + 2;  
                7.    y := y + n;  
                8.    x := x - 1;  
              end;  
          9.    write(y);  
        end;  
  end;
```

POSSIBILE SOLUZIONE ESERCIZIO 4

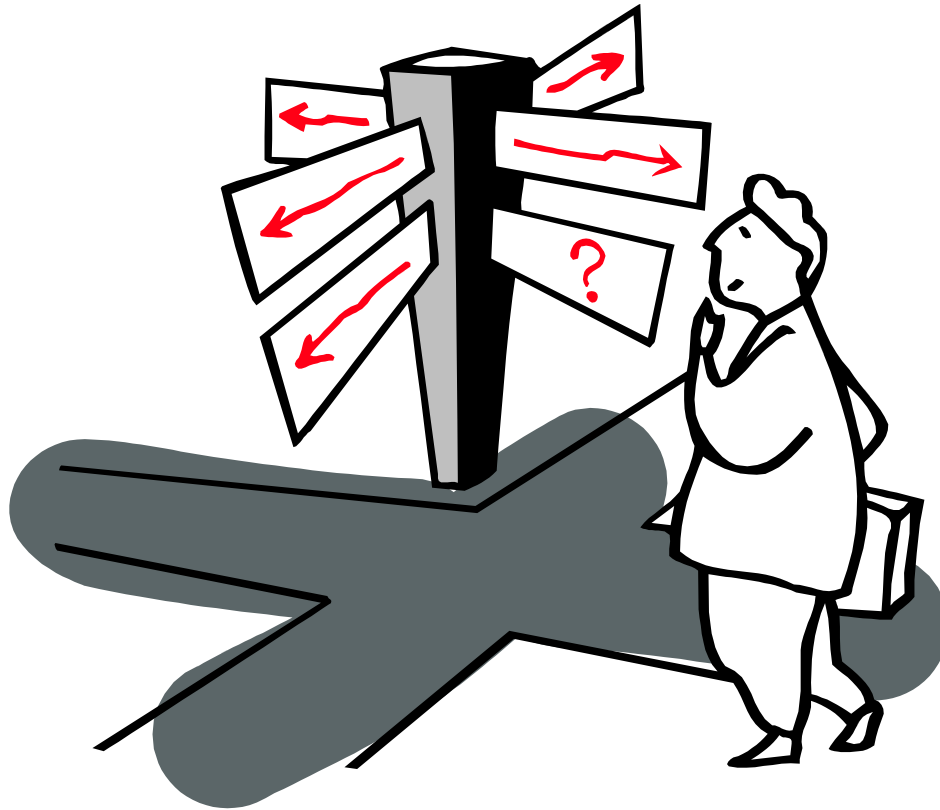
Test Suite

- TC1 (x = -1, No valore stampato)
- TC2 (x = 1, y = 1)
- TC3 (x = 2, y = 4)

TC2 è necessario?



THE END ...



Domande?