7.2.10.1 Inizializzazione

Variabili e oggetti, prima di essere utilizzati, devono essere correttamente inizializzati. Per evitare l'allocazione di oggetti non inizializzati:

- rendere tutte le variabili private e, se necessario, fornire l'accesso ad esse dall'esterno esclusivamente attraverso i metodi get() e set();
- aggiungere in ogni oggetto una variabile booleana privata (es: isInizialized) e fare in modo che ogni costruttore, come ultima operazione, la inizializzi a "true";
- in ogni metodo che non sia un costruttore, verificare che la variabile di inizializzazione della classe sia impostata a true prima di eseguire qualsiasi operazione.

Esempio:

```
public class MyClass {
    private boolean isInizialized;
    private String nome;
    public MyClass(String nome){
    this.nome = nome;
    this.isInizialized = true;
    }
    public String getNome(){
    return (isInizialized == true ? this.nome : null);
    }
}
```

Se la classe ha costruttori statici è necessario seguire la stessa procedura ma a livello di classe:

- rendere tutte le variabili statiche private e, se necessario fornirne l'accesso dall'esterno della classe stessa: questo deve sempre essere consentito esclusivamente attraverso i metodi get() e set();
- aggiungere alla classe una variabile booleana privata statica (es: isClassInizialized) e fare in modo che ogni costruttore statico, come ultima operazione, la inizializzi a "true";
- prima di eseguire qualsiasi operazione, in ogni metodo statico ed ogni costruttore si deve verificare che la variabile "isClassInitialized" sia impostata a "true";
- Gestione delle allocazioni / deallocazioni di memoria dinamica;

Prima di uscire da una classe occorre ricordarsi sempre di azzerare il contenuto delle variabili. Si supponga, nel seguente esempio, che la variabile k contenesse la chiave per decriptare un messaggio cifrato:

Esempio :

Forma non corretta

```
public class Decodificatore {
   private byte[] k;
}
```

Forma corretta

```
public class Erase {
   private byte[] k;
   public void clear() {
   for(int i = 0; i < k.length; i++)
          k [i] = (byte) 0x00;
   }
}
```

Limitare l'accesso alle classi, ai metodi e alle variabili.

Ogni classe, metodo e variabile dovrebbero essere definiti come private o protected. Potrebbero essere dichiarati"public" in casi del tutto eccezionali, motivati e documentati. Ogni variabile privata deve essere accessibile dall'esterno unicamente attraverso metodi set() e get() per mantenere l'oggetto al sicuro.

Esempio:

```
public class Studente {
        private int eta;
        public int getEta(){
```