

## Conținut

1. Inspectarea cerințelor (Requirements Inspection).....	1
2. Inspectarea arhitecturii (Architectural Inspection) .....	1
3. Inspectarea codului sursă (Source Code Inspection).....	2

### 1. Inspectarea cerințelor (Requirements Inspection)

1. Se parcurge documentul cu cerințe (enunțul problemei), e.g., [RequirementsBiblioteca\\_v1.0.pdf](#).
2. Pentru fiecare element **R01..R07** din documentul [Lab01\\_RequirementsPhaseDefects.pdf](#) se caută în enunțul problemei dacă elementul **Rxx** este îndeplinit sau nu;
3. În fișierul [Lab01\\_ReviewReport.xlsx](#), în sheet-ul corespunzător inspectării cerințelor, se completează:
  - pentru fiecare element **Rxx** cel puțin câte o observație proprie (e.g., **Rxx** apare în enunțul problemei dar nu este adecvat, **Rxx** nu apare în enunțul problemei deși este potrivit să fie inclus în enunțul problemei);
  - timpul necesar pentru realizarea inspectării cerințelor;
4. Se modifică enunțul inițial, astfel încât să se reflecte observațiile incluse în raportul de inspectare a cerințelor, i.e., [RequirementsBiblioteca\\_v2.0.pdf](#).

### 2. Inspectarea arhitecturii (Architectural Inspection)

1. Se identifică arhitectura aplicației (i.e., clasele, pachetele, dependențele logice între clasele aplicației, etc.), e.g., proiectul [Biblioteca](#);
2. Se verifică corespondența logică dintre elementele de arhitectură (i.e., clase, relații între clase, attribute, metode, pachete, etc.) și diagrama de clase supusă inspectării, i.e., [DiagramBiblioteca\\_v1.0.pdf](#);
3. Pentru fiecare element **A01..A10** din documentul [Lab01\\_ArchitecturalDesignPhaseDefectsChecklist.pdf](#) se caută în arhitectura aplicației și în diagrama de clase dacă elementul **Axx** apare în arhitectură și în diagrama de clase sau nu;
4. În fișierul [Lab01\\_ReviewReport.xlsx](#), în sheet-ul corespunzător inspectării arhitecturii, se completează:
  - pentru fiecare element **Axx** cel puțin câte o observație proprie (e.g., **Axx** apare în arhitectura aplicației și/sau în diagrama de clase dar nu este adecvat, **Axx** nu apare în arhitectura aplicației și/sau diagrama de clase deși este potrivit să fie inclus);
  - timpul necesar pentru realizarea inspectării arhitecturii;
5. Se corectează arhitectura aplicației și/sau diagrama de clase în conformitate cu observațiile incluse în raportul de inspectare a arhitecturii aplicației, i.e., [DiagramBiblioteca\\_v2.0.pdf](#).
6. Dacă este necesar, modificările la nivelul arhitecturii se propagă la nivelul documentului cu cerințele aplicației, i.e., [RequirementsBiblioteca\\_v2.0.pdf](#).

### 3. Inspectarea codului sursă (Source Code Inspection)

1. Se identifică elementele de implementare prezente în codul sursă (i.e., obiecte și/sau variabile și tipul acestora, metodele și semnăturile acestora, algoritmi implementați și complexitatea acestora, instrucțiuni de programare (e.g., **for**, **if**, **while**, **repeat**, **switch**, etc.)), e.g., proiectul [Biblioteca](#);
2. Pentru fiecare element **C01..C12** din documentul [Lab01\\_ProgramCodingPhaseDefectsChecklist.pdf](#) se stabilește dacă elementul **Cxx** apare în codul sursă al aplicației sau nu;
3. În fișierul [Lab01\\_ReviewReport.xlsx](#), în sheet-ul corespunzător inspectării codului sursă, se completează:
  - pentru fiecare element **Cxx** cel puțin câte o observație proprie (e.g., **Cxx** apare în codul sursă al aplicației dar nu este adecvat, **Cxx** nu apare în codul sursă al aplicației deși este potrivit să fie inclus);
  - timpul necesar pentru realizarea inspectării codului sursă;
4. Se corectează **codul sursă** în conformitate cu observațiile incluse în raportul de inspectare a codului sursă.
5. Dacă este necesar, modificările de la nivelul codului sursă se propagă la nivelul arhitecturii, actualizând diagrama de clase, i.e., [DiagramBiblioteca\\_v2.0.pdf](#). În continuare, dacă este necesar, actualizările de la nivelul arhitecturii se propagă la nivelul documentului cu cerințele aplicației, i.e., [RequirementsBiblioteca\\_v2.0.pdf](#).