## Seminar Objectives

**Inspection AND Feature Tour Testing**

### Topics

- **Inspection**
- **Tour testing (Feature)**

## Assignment 1 - 10-minutes - Discussion

### Topics

- Similarities/Differences between code inspection and walkthroughs.
- Inspection/Walkthroughs team and duties.
- Types of errors to be discovered. Name at least three and give an example for each type.
- Pair programming

## Assignment 2 – 40 minutes – Inspection

Inspect the documents (problem statement, design, source code) for the received problem.

Inspection refers to the analysis and the highlighting of the current state of the documents into a report.

Inspection may cause modification of the analyzed documents, like:

- Clarification of the statement problem;
- Modification of the design and/or the source code.
- Use the same available documents from the Laboratory 1 assignments.
- For the identification of the ambiguities/defects the following *check-lists* will be used:
  - a. **Statement problem:** Lab01_RequirementsPhaseDefectsChecklist.pdf;
  - b. **Design:** Lab01_DesignPhaseDefectsChecklist.pdf**;**
  - c. **Source code**: Lab01_ProgramCodingPhaseDefectsChecklist.pdf.
- For the inspected documents/artifacts a report will be realized (Lab01_Review Form.xls).

## Assignment 3 – 40 minutes – Feature Tour Testing

Tour testing

- **Lecture 1: Function Testing, Tours, & A Taxonomy of Techniques**
- http://www.testingeducation.org/BBST/testdesign/
- Video part A (9 mins, 14 secs) - starting 4:45 to 9:15 minutes.
- Slides - Slide set for all Test Design course lectures – slides 9->21 (->41)
- Application to apply Feature Tour Testing – Socrative (https://www.socrative.com/)
  - Conduct a Feature tour testing by "Playing" with the application and create a document that contains a List of feature that you have discovered.

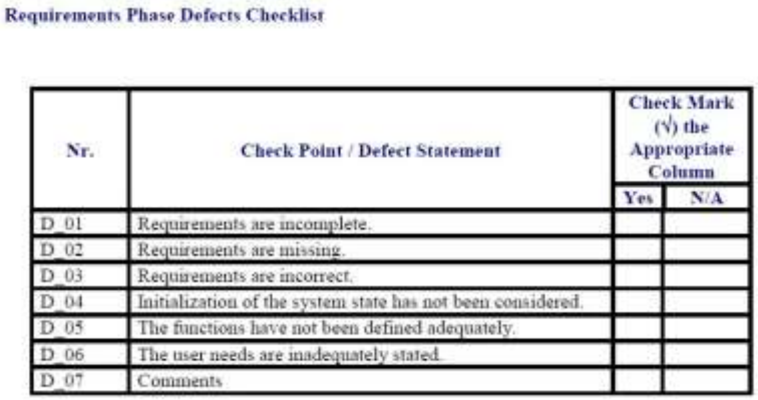
## Assignment 4 – 5-10 minutes – Quiz

**Problem statement.** Write a program that reads natural numbers n1, n2, …, nk and prints the longest sequence ns, ns+1, …, nd, with 1≤s≤d≤k, that contains only prime numbers.

**Problem design.** The program must have: a subalgorithm that reads the given numbers, a function that verifies if a natural number is prime;a subalgorithm that computea the indexies s and d, 1≤s≤d≤k, with the property that ns, ns+1, …, nd are prime numbers; a subalgorithm that prints the numbers ns, ns+1, …, nd.

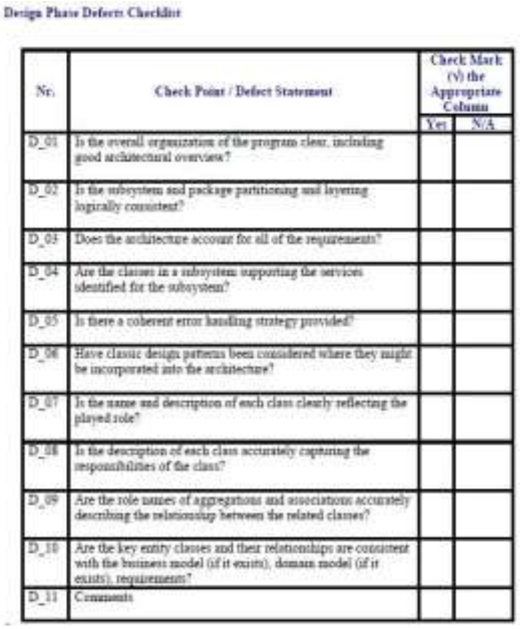
**Source code.**

```
public class LongestPrimeSequence {
    private ArrayList l;
    private int start, length;
    public LongestPrimeSequence(){
        System.out.println("Long. Seq. empty ...");
    }
    public void setSequence(ArrayList l){
        this.l=l;
    }
    public LongestPrimeSequence(ArrayList newL){
        this.l = newL;
        this.start=-1;
        this.length=0;
    }
    public int getStart(){return this.start;}
    public int getLength(){return this.length;}
  public boolean isPrime(int n) throws ValueException{
    boolean b = true;
    if(n<0){
        throw new ValueException("data not valid");
    }
    if(n<2){
        b=false;
    }
    else{
        int i=2;
        while (i< (n/2)){
            if ((n % i) == 0){
                    b=false;
            }
            else
                b=true;
            i++;
        }
    }
    return b;
 }
public void SolveLongestSequence() throws ValueException{
    int posI=-1, lengthI=0, i=0;
    int posF=-1, lengthF=0;
    while(i<this.l.size()){
        if(isPrime((int)this.l.get(i))==true){
            if(posI==-1){
                posI=i;
                lengthI=1;
            }
            else
                lengthI++;
        }
        else{
            if(lengthI>lengthF){
                lengthF=lengthI;
                posF = posI;
            }
        }
        i++;
    }
    this.start =posF;
    this.length=lengthF;
  }
}
```



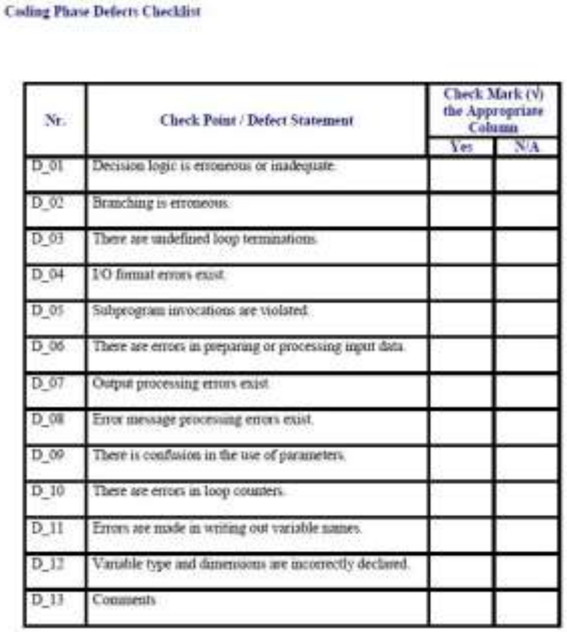
Requirements Phase Defects Checklist

| Nr. | Check Point / Defect Statement | Check Mark (√) the Appropriate Column | |
|---|---|---|---|
| | | Yes | N/A |
| D_01 | Requirements are incomplete. | | |
| D_02 | Requirements are missing. | | |
| D_03 | Requirements are incorrect. | | |
| D_04 | Initialization of the system state has not been considered. | | |
| D_05 | The functions have not been defined adequately. | | |
| D_06 | The user needs are inadequately stated. | | |
| D_07 | Comments | | |



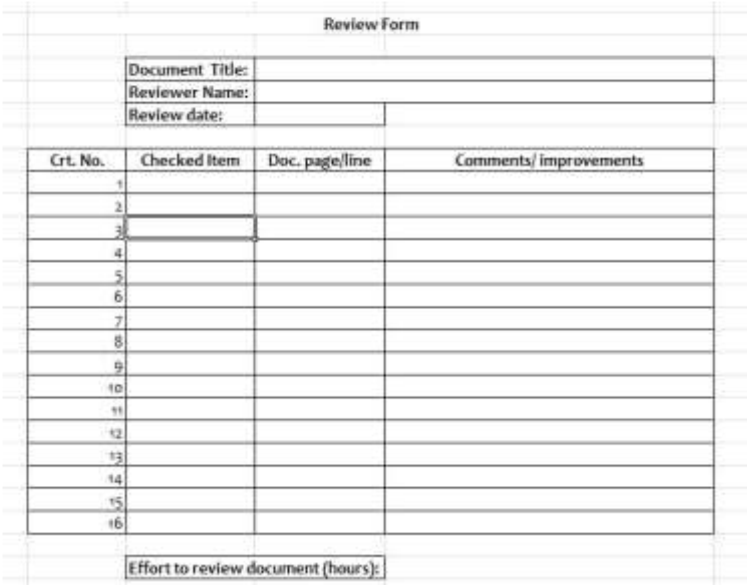
Design Phase Defects Checklist

| Nr. | Check Point / Defect Statement | Check Mark (√) the Appropriate Column | |
|---|---|---|---|
| | | Yes | N/A |
| D_01 | Is the overall organization of the program clear, including good architectural overview? | | |
| D_02 | Is the subsystem and package partitioning and layering logically consistent? | | |
| D_03 | Does the architecture account for all of the requirements? | | |
| D_04 | Are the classes in a subsystem supporting the services identified for the subsystem? | | |
| D_05 | Is there a coherent error handling strategy provided? | | |
| D_06 | Have classic design patterns been considered where they might be incorporated into the architecture? | | |
| D_07 | Is the name and description of each class clearly reflecting the played role? | | |
| D_08 | Is the description of each class accurately capturing the responsibilities of the class? | | |
| D_09 | Are the role names of aggregations and associations accurately describing the relationship between the related classes? | | |
| D_10 | Are the key entity classes and their relationships are consistent with the business model (if it exists), domain model (if it exists), requirements? | | |
| D_11 | Comments | | |



Coding Phase Defects Checklist

| Nr. | Check Point / Defect Statement | Check Mark (√) the Appropriate Column | |
|---|---|---|---|
| | | Yes | N/A |
| D_01 | Decision logic is erroneous or inadequate. | | |
| D_02 | Branching is erroneous. | | |
| D_03 | There are undefined loop terminations. | | |
| D_04 | I/O format errors exist. | | |
| D_05 | Subprogram invocations are violated. | | |
| D_06 | There are errors in preparing or processing input data. | | |
| D_07 | Output processing errors exist. | | |
| D_08 | Error message processing errors exist. | | |
| D_09 | There is confusion in the use of parameters. | | |
| D_10 | There are errors in loop counters. | | |
| D_11 | Errors are made in writing out variable names. | | |
| D_12 | Variable type and dimensions are incorrectly declared. | | |
| D_13 | Comments | | |



Review Form

| | |
|---|---|
| Document Title: | |
| Reviewer Name: | |
| Review date: | |

| Crt. No. | Checked Item | Doc. page/line | Comments/ improvements |
|---|---|---|---|
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| 6 | | | |
| 7 | | | |
| 8 | | | |
| 9 | | | |
| 10 | | | |
| 11 | | | |
| 12 | | | |
| 13 | | | |
| 14 | | | |
| 15 | | | |
| 16 | | | |

Effort to review document (hours):