SAT: Refactoring

In a team of 3-4 students (aim for 3 if we have 1-2 students left they will join existing teams) talk about these questions and write your collective answers in the report. You may refer to Reading2.

Testing Principles

1. What are the differences between systematic and effective testing and gut-feeling testing? (the example of John vs Eleanor). Gut-feeling tests: what characterizes John's behavior? What makes Eleanor's behavior more systematic? Our goal is to come up with one sentence that explains what gut-feeling testing is, and another one explaining what systematic testing is.

2. **This part should not be included in your report**
Recall the testing principles mentioned in the book:

- Exhaustive testing is impossible
- Knowing when to stop
- Variability is important (the pesticide paradox)
- Bugs happen in some places more than others (defect clusters)
- Testing will never be perfect
- Context is king
- Verification is not validation

Each of you is responsible for explaining 2 principles to the others. When a team mate explains a principle don't rely on your knowledge, act like you hear this information for the first time and ask questions for clarifications, examples, etc, whatever you would need to learn this information as if you hear it for the first time.

3. Compare the testing principles:

- Exhaustive testing is impossible vs testing shows the presence of bugs but not their absence
- Absence-of-errors fallacy vs pesticide paradox
- Verification vs validation

The testing pyramid

4. Can you think of reasons to do manual testing?

5. Deciding the test level isn't that straightforward. How would you do it? What would you take into account to decide which test level to go for?

6. Personal taste plays a role. Are you more of a 'unit testing type of person' or an 'integration testing type of person' ?

7. Consider the following program and provide your testcases with explanations for why each testcase is needed:

```
import java.util.*;
public class Triangle {
 public static void main(String[] args) {
  Scanner in = new Scanner(System.in);
  System.out.print("Input side1: ");
  int s1 = in .nextInt();
  System.out.print("Input side2: ");
  int s2 = in .nextInt();
  System.out.print("Input side3: ");
  int s3 = in .nextInt();

  if (!isValidTriangle(s1, s2, s3))
    System.out.println("These sides can't form a triangle");
  else if (isEquilateral(s1, s2, s3))
    System.out.println("These sides can form an equilateral triangle");
  else if (isIsosceles(s1, s2, s3))
    System.out.println("These sides can form an isosceles triangle");
  else
    System.out.println("These sides can form a scalene triangle");
 }
 public static boolean isValidTriangle(int a, int b, int c) {
  return (a + b > c && b + c > a && c + a > b);
 }
 public static boolean isEquilateral(int a, int b, int c) {
  return (a == b && b == c);
 }
 public static boolean isIsosceles(int a, int b, int c) {
  return (a == b || b == c || a == c);
 }
}

```