**MSSV: DE180292**

**Name: Nguyễn Kim Đăng Khoa**

**Class: SE18D12**

**Lab2\_Part1**

**Question 1:**

1. *Key Characteristics of Each Test Level*

*Software testing is typically performed at* ***four main levels****:*

**🔹 1. Component Testing (also called Unit Testing)**

* **Test Objectives**:
  + Verify the correctness of individual components (e.g., functions, classes, modules).
  + Detect logic errors, syntax errors, incorrect handling of inputs/outputs.
* **Test Basis**:
  + Detailed design
  + Source code
  + Technical specifications

**🔹 2. Integration Testing**

* **Test Objectives**:
  + Test interactions between components or subsystems.
  + Detect interface mismatches, communication errors, or incorrect data passing.
* **Test Basis**:
  + Architecture design
  + Interface specifications
  + System design

**🔹 3. System Testing**

* **Test Objectives**:
  + Verify the complete, integrated system against defined requirements.
  + Check both functional and non-functional behaviors.
* **Test Basis**:
  + Software requirements specification (SRS)
  + Use cases
  + Functional specifications

**🔹 4. Acceptance Testing**

* **Test Objectives**:
  + Validate the system from the business/user perspective.
  + Confirm the system meets acceptance criteria and is ready for deployment.
* **Test Basis**:
  + Business requirements
  + Acceptance criteria
  + User scenarios

1. *Common Defects or Failures at Each Level + 3 Examples*

**🔹 1. Component Testing – Common Defects**

* Logical errors in code
* Uninitialized variables
* Incorrect exception handling

**Examples**:

1. A calculateTax() function returns incorrect results due to faulty logic.
2. A variable is used before being initialized, causing a runtime error.
3. Missing a null check leads to an application crash.

**🔹 2. Integration Testing – Common Defects**

* Interface mismatch between modules
* Incorrect data transfer between services
* Misordered method calls

**Examples**:

1. A login module sends malformed JSON to the user service.
2. The app does not properly receive a response from a payment gateway.
3. A function calls getUser() before authenticate(), causing unexpected behavior.

**🔹 3. System Testing – Common Defects**

* Functionality not aligned with requirements
* Poor performance
* System crashes with high data volume

**Examples**:

1. The "Book Tour" button does not follow the correct booking process.
2. The page loads very slowly when displaying over 1000 records.
3. The system crashes when a user uploads a large file.

**🔹 4. Acceptance Testing – Common Defects**

* Missing or incorrect business logic
* Misaligned with user expectations
* Incomplete or non-user-friendly features

**Examples**:

1. Customers cannot use a discount voucher during checkout, despite business rules.
2. The tour cancellation process does not follow the refund policy.
3. The UI is not intuitive for non-technical users.

**Question 2:**

|  |  |  |
| --- | --- | --- |
| **Error** | **Description** | **Fix** |
| 1 | Convention Error. Incorrect variable naming convention,  Java uses camelCase for variable names. | private String claimId; |
| 2 | Logic Error. String comparison using == (logic error),  == checks reference equality, not string content | if("Pending".equals(claimStatus)) |
| 3 | Convention Error. Method naming convention violation, methods name in java should start with a lowercase letter | boolean processClaim(String statusUpdate) |
| 4 | Standard Error. Lack of Access Modifiers on Fields,  fields should be private to enforce encapsulation | private String claimId;  private double amount;  private String claimStatus; |
| 5 | Logic Error. Potential NullPointerException claimStatus could be null, causing a crash | if("Approved".equals(claimStatus)) |
| 6 | Standard Error. Missing this for Clarity in Constructor, ambiguous assignment; this.claimId improves readability | this.claimId = id;  this.amount=claimAmount; |