**1. Task**

**a) What each one is:**  
Unit testing is testing the smallest parts of the code, like individual methods, in isolation. It's usually written and maintained by developers. Component Integration Testing (CIT) checks if multiple components or modules work properly together.

**b) When you would use each one:**  
Unit testing is done early in development, right after writing a method. CIT is done after combining modules to make sure they integrate and function correctly together.

**c) The advantages of each type:**  
Unit tests are fast, easy to debug, and help catch issues early. CIT gives better real-world coverage by testing how components interact with each other.

**d) The disadvantages of each type:**  
Unit tests don’t catch integration issues since they only test isolated parts. CIT tests are slower, harder to maintain, and debugging can be tricky when something breaks.

2.Task:

**a) Sanity Testing**

* **What:** Quick check to ensure that specific changes or bug fixes work as expected.
* **When:** After receiving a build with small changes or bug fixes.
* **Advantages:** Quick and ensures basic functionality.
* **Disadvantages:** Limited in scope, doesn’t cover the entire system.

**b) Smoke Testing**

* **What:** Basic check of major functionalities to ensure the build is stable enough for further testing.
* **When:** Before starting more detailed testing on a new build.
* **Advantages:** Quick, helps identify major issues early.
* **Disadvantages:** Limited coverage, only verifies critical functions.

**c) Exploratory Testing**

* **What:** Testers explore the application without predefined test cases, using their intuition.
* **When:** When there are no clear requirements or documentation, or during late stages.
* **Advantages:** Flexible, uncovers unexpected issues.
* **Disadvantages:** Hard to track coverage, inconsistent results.

**d) Risk-Based Testing**

* **What:** Focuses on testing the most critical, high-risk parts of the application.
* **When:** When resources or time are limited, or when certain features are higher risk.
* **Advantages:** Prioritizes high-impact areas, efficient use of resources.
* **Disadvantages:** May overlook lower-risk areas, requires deep knowledge of the app.
  1. **API Testing**
* **What it is:** Verifies that APIs function correctly, focusing on data exchange, security, and performance.
* **When to use:** When an app relies on APIs for communication.
* **Advantages:** Ensures integration works, can be automated.
* **Disadvantages:** Requires technical knowledge, may miss UI-related issues.

Task 3

### ****1. Black Box Testing****

* **What:** Tests what the system does, without knowing the code.
* **When to use:** Functional, system, or user acceptance testing.
* **Advantages:**
  + Easy to do, no coding knowledge needed.
  + Focuses on user experience.
* **Disadvantages:**
  + Can miss internal code issues.
  + Limited test coverage.

### ****2. White Box Testing****

* **What:** Tests the internal code and logic.
* **When to use:** Unit or integration testing.
* **Advantages:**
  + Detects hidden code bugs.
  + Thorough and detailed.
* **Disadvantages:**
  + Requires programming knowledge.
  + Time-consuming.

## **b) 4 Testing Types – Black or White Box**

| **Testing Type** | **Type** | **Why?** |
| --- | --- | --- |
| Smoke Testing | Black Box | Tests basic functions from the outside, no code needed. |
| Sanity Testing | Black Box | Quick checks after changes, user-level testing. |
| API Testing | White Box\* | Often needs knowledge of data formats and responses. |
| Unit Testing | White Box | Written by developers, tests code directly. |

\*Note: API Testing can be black box in some cases, but often leans toward white box due to technical detail.