1. **What is Verification and Validation?\*\*\***

**-**

**Verification** is checking whether the software is built correctly according to design and plan. It answers: Are we building the product right?

**Validation** is checking whether the software meets the real needs of the customer. It answers: Are we building the right product?  
>>Example: Verification checks if the login form is coded as planned. Validation checks if the login form is useful to the user.

1. **What are the goals of software testing?\*\*\***

**-**

First, to make sure the software works as expected and meets all its requirements.

Second, to find bugs or errors where the software does not behave correctly.

>>Example: A test shows that clicking 'submit' saves data (goal 1), or crashes the system (goal 2).

1. **What is a test case?\*\*\***

**-**

A test case is a detailed check that includes input, expected output, and actual result.

It helps test one part of the software to see if it works correctly.  
>>Example: Input: username + password, Expected: dashboard appears.

1. **What is a software testing strategy?\*\*\***

**-**

It is the overall plan of how testing will be done — starting from small parts to the full system.

It often follows a spiral model: starting with unit tests and ending with system tests.

>>Like checking ingredients, cooking steps, and the full dish.

1. **What is Unit Testing?\*\*\***

**-**

Unit testing checks the smallest part (module) of software individually.

It tests logic, data, limits, and error handling inside that module.

>>Example: Testing only the "add to cart" function in an e-commerce app.

1. **What is Integration Testing?\*\*\***

**-**

Integration testing checks if two or more modules work properly together.

It finds problems in how different parts are connected.

>>Example: After testing login and profile separately, test if login leads to profile correctly.

1. **What are the types of Integration Testing?\*\*\***

**-**

**Non-incremental (Big Bang)**: All modules are combined and tested together at once. It’s fast but hard to find bugs.

**Incremental**: Modules are tested step-by-step, making bugs easier to catch and fix.  
>>Like testing a machine one part at a time vs. starting everything at once.

1. **What are the approaches in Incremental Integration?\*\***

**-**

**Top-down**: Start testing from the top module and move down.

**Bottom-up**: Begin from the lower modules and go up.

**Sandwich**: Mix of both top-down and bottom-up.  
>>Example: Like building a sandwich from both sides at once.

1. **What is Alpha and Beta Testing?\*\*\***

**-**

**Alpha Testing**: Done at the developer’s site by selected users. Developers watch and fix bugs.

**Beta Testing**: Done at the real user’s place without developers. Bugs are reported back.  
>>Alpha = Controlled lab test; Beta = Real-world trial.

1. **What is White Box Testing?\*\*\***

**-**

White box testing checks the internal code and logic of the software.

It needs knowledge of how the program works inside.  
>>Example: A developer checks if all code paths in a function are used properly.

1. **What is Black Box Testing?\*\*\***

**-**

Black box testing checks software based on input and output, without looking at the code.

It is focused on functionality only.  
>>Example: User clicks "search" and checks if correct results appear, without knowing the logic.

1. **What is System Testing?\*\*\***

**-**

System testing checks the full system, including all modules and interactions.

It ensures the complete software works properly.  
>>Like testing the full online shopping process from sign-up to payment.

1. **What are the types of System Testing?\*\*\***

**-**

**Recovery Testing** – Check if the system can recover from errors or crashes.  
>>Test if app restarts safely after a crash.

**Security Testing** – Test if the system is protected from unauthorized access.  
>>Test if login blocks wrong passwords or hackers.

**Stress Testing** – Run the system under heavy load to test stability.  
>>Send thousands of users or requests to check if it crashes.

**Performance Testing** – Measure speed, response time, and efficiency.  
>>Check if the system loads in under 2 seconds under normal use.

1. **What is Debugging?\*\*\***

**-**

Debugging is the process of finding and fixing the cause of software errors found during testing.

It starts when actual output is different from expected, and the reason is unknown.  
>>Example: Test shows wrong result, so developer checks code to fix it.

1. **Why is debugging difficult?\*\*\***

**-**

The error and its cause may be in different places.

Fixing one issue may create another.

Bugs can disappear or be caused by small mistakes or timing.  
>>Like fixing one water pipe, only to find another leak elsewhere.

1. **What are the main debugging strategies?**

**-**

**Brute Force** – Print everything and search for errors manually. Simple but slow.

**Backtracking** – Go backward from the error to find the root cause.

**Cause Elimination** – Make a list of possible causes and remove them one by one by testing.  
👉 Like a doctor diagnosing illness by testing symptoms one by one.







