Unite & Integration Testing

1. What is Unit Testing?
2. What is Integration Testing?
3. What is All-at-Once Integration? What are its strengths and weaknesses?
4. What is Top-Down Integration? What are its strengths and weaknesses?
5. What is Bottom-Up Integration? What are its strengths and weaknesses?
6. What is Sandwich Integration? What are its strengths and weaknesses?
7. How do the integration strategies relate to Continuous Integration (CI)?
8. A web application features a "Discount Code" input field. The requirements for this field are as follows:
   Valid Discount Codes:
   - Must be between 5 and 10 characters long (inclusive).
   - Must contain only uppercase letters (A-Z) and digits (0-9).
   - Must start with at least one uppercase letter.

   Invalid Discount Codes:
   - Any code that does not meet the "Valid Discount Codes" criteria.
   - Specifically, codes that are too short, too long, contain lowercase letters, contain special characters, or do not start with an uppercase letter.

   Based on the requirements, the following equivalence classes can be identified:
   Valid Equivalence Classes:
   - EC1 (Valid Length & Characters): Codes 5-10 characters long, uppercase letters and digits only, starting with an uppercase letter.

   Invalid Equivalence Classes:
   - EC2 (Too Short): Codes less than 5 characters long.
   - EC3 (Too Long): Codes more than 10 characters long.
   - EC4 (Contains Lowercase): Codes containing at least one lowercase letter.
   - EC5 (Contains Special Characters): Codes containing at least one special character (e.g., !, @, #).
   - EC6 (Starts with Non-Uppercase): Codes starting with a digit or special character.

   Design a set of black-box test cases using Equivalence Class Partitioning for the "Discount Code" input field.
   For each test case:
   a) Specify the equivalence class it represents
   b) Test data
   c) Expected outcome

1. Unit testing checks one small part of a program, the "unit" by itself. Like a single function or method to make sure each piece works independently
2. Integration tests multiple units together, "integrating" them to find problems with how the modules interact
3. All-at-once integration means combining every module and testing the whole system. It is a simple test with no specific debugging blocks needed, but can be hard to debug and figure out an issue and you can only do it once everything is built
4. Top-Down integration starts at the top level modules and moves downwards. This is good because you can test early, but requires blocks for lower level modules and the lower modules are tested late
5. Bottom-Up integration starts by testing low level modules first. This is good because the low level modules are tested early and no blocks are needed. However, the high-level modules aren't tested until late.
6. Sandwich integration combines top down and bottom up integration at the same time. This is faster for testing and tests both modules early, however, it is complex to manage and may require blocking
7. Continuous Integration runs automated builds and tests every time code changes. Unit tests are run on every commit and integration tests are run every time a module interaction is

changed.
8. .
1. Test Case 1
1. EC1 - Valid
2. Test data: A123B
3. Expected outcome: Accepted
2. Test Case 2
1. EC2 - Too Short
2. Test Data: H2Q
3. Expected Outcome: Rejected
3. Test Case 3
1. EC3 - Too Long
2. Test data: A12345678901
3. Expected Outcome: Rejected
4. Test Case 4
1. EC4 - Contains lowercase letters
2. HeLlo123
3. Expected outcome: Rejected
5. Test Case 5
1. EC5 - Contains Special Characters
2. Test data: ABC#1
3. Expected outcome: Rejected
6. Test Case 6
1. EC6 - Starts with digit
2. Test data: 1ABCD
3. Expected outcome: Rejected