Test KEN1520 Exam 2025

Question 1 − FILL_ARCHITECTURE_1 − 201897.1.0

You are leading the development of a **mobile banking app**. The app needs to support **millions of users**, offer **real-time updates**, and be **independently deployable** across several internal teams working in parallel. The app should also be resilient and scalable.

The most appropriate architecture for this system is likely a [Microservices architecture]
This architecture supports scalability by enabling [independent deployment of components]
One key challenge of this approach is [increased complexity in service coordination]

[List]
Microservices architecture
Layered architecture
Monolithic architecture
Client-server architecture

[List]
independent deployment of components
code reuse through inheritance
unified build pipelines
global exception handling
shared memory access across components

[List]
increased complexity in service coordination
tight coupling between components
lack of testability
limited reusability
insufficient logging mechanisms

Question 2 − FILL_Testing_1_AS − 202969.5.0

Fill the blanks.

[Testing] is the process of verifying whether software meets its expectations.
Within this process, there are mainly two types: structural and [behavioural].
In [unit testing], the software's different parts are tested separately.
In [integration testing], the software is tested by combining different units and verifying the interactions between components.

[List]
Testing
Static Checking
Specifications
Coupling

[List]
behavioural
deterministic
strong
declarative
unit

[List]
unit testing
integration testing
structural testing
partiioning testing

[List]
integration testing
unit testing
regression testing
partitioning testing

Question 3 – KPR_Checking_1_AS – 202999.1.0

Match each term with the most appropriate definition.

Checking	The Process of analysing source code to identify errors and potential issues.
Static Checking	The checking is performed before the code runs, e.g., type checking in Java
Dynamic Checking	The checking is performed at runtime, e.g., illegal operations such as division by zero.
Style checking	Checking for code style guidelines.

Question 4 – KPR_Debugging_1_AS – 203029.1.1

For each statatement about Debugging, mark if they are True or Not True.

	True	Not true
In debugging, the first important step is trying making bugs impossible.	X	
Static and Dynamic checking are important process in preventing bugs, as they can be caught early.	X	
When Debugging, reproducing the bug is not important.		X
Debugging is the process of identifying, analysing, and fixing bugs or defects in a software program.	X	
Debugging can only be done using IDEs' debuggers.		X

Question 5 – KPR_UML_YCS – 202261.2.1

Mark each of the following statements as **True** or **False**.

	True	Not true
A use case diagram shows the dynamic behavior of a system, including how objects interact with one another over time.		X
In UML, an association represents a structural relationship between classes.	X	
A class diagram can include methods, attributes, and relationships such as inheritance and aggregation.	X	
UML activity diagrams are used to model database schemas and table relationships.		X

Question 6 – MATCH_ARCH_1 – 201893.1.0

Match each **architecture style** with its most appropriate **description**.

Layered Architecture	Organizes system into horizontal layers (e.g., UI, logic, data), each depending on the one below.
Microservices	Divides functionality into independently deployable units, each handling a distinct feature.
Monolithic Architecture	The entire application is deployed as a single unit, with tightly coupled components.
Client-Server Architecture	Separates client and backend logic, typically over a network.
Event-Driven Architecture	Organizes code into independent components that react to external inputs or triggers.

Question 7 – MATCH_GIT_1_AS – 202966.1.0

Match the terms/commands to what they mean/do.

git init	Git command required to start a local git repository in a directory.
git commit	Git command to create a "snapshot" of the current state of the files in the staging area.
git merge	This git command automatically creates an additional commit pointing to two branches: the branch you run this command and the branch which name you pass to the command, combining the content of both branches.
Working directory	A term referring the computer directory where all your project files are located.
Staging area	All files added in this location will be part of the next created commit.

Question 8 – MATCH_OOP_YCS – 202175.1.1

Match the following Object Oriented Programming concepts with their definitions:

Encapsulation	Allowing only certain parts of the code to be visible
Polymorphism	Allowing different implementations of the same method
Inheritance	Reusing common behavior in new classes
Abstraction	Hiding internal implementation details

Question 9 – MC_CLEANCODE_1 – 201888.1.0

Which of the following best exemplifies clean code?

- **A** Code that is highly optimized for speed but difficult to read
- **B** Code with no comments to avoid distractions
- **C** Code that clearly communicates its intent and is easy to modify
- **D** Code with the most compact syntax possible
- **E** Code that uses global variables to reduce parameter passing

Question 10 – MC_TDD_1_AS – 203021.2.0

Which of the following best describes the process of Test-Driven Development (TDD)?

- **A** Write the full implementation first, then write tests to verify correctness.
- **B** Write tests only after the software has been deployed.
- **C** Write a specification for the method, write tests according to the specification, then write the code.
- **D** Skip testing if the implementation seems simple or obvious.

Question 11 – OP_AGILE_YCS – 202241.1.1

Imagine you're part of a newly formed interplanetary Agile software development team. Your team is tasked to develop a universal knowledge-sharing platform for all species in the Planets United

You're preparing for your first Agile sprint, but there's a twist: standard practices don't work well across species since your team consists of different species from all over the galaxy with different ways of thinking, communicating, and collaborating.

Your mission:
1) Design a new, inclusive Agile process or framework that your team can realistically and enjoyably follow.
2) Describe the process or framework by explaining how it would work in practice.
3) Explain how your process adapts core Agile values and principles to suit a galactic team
4) Explain how you would onboard new members into your Agile process.

Grading instruction

Creativity (Number of points: 1)

Is the idea imaginative, unique, or fun in a galactic context?

Clarity (Number of points: 2)

Is the process or framework clearly described?

Practicality (Number of points: 2)

Can it realistically be applied in a real-world Agile context?

Justification (Number of points: 3)

Does it clearly map elements of the process to core Agile values?

Onboarding (Number of points: 2)

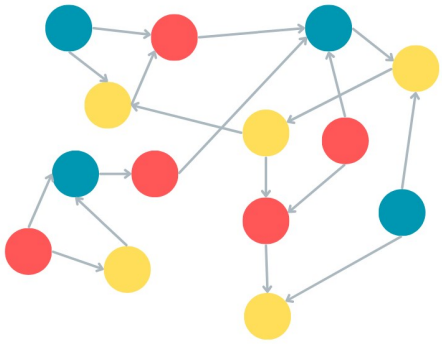
Is there a clear and reasonable method for teaching it to team members?

Question 12 – OPEN_Coupling_1_AS – 203000.2.0

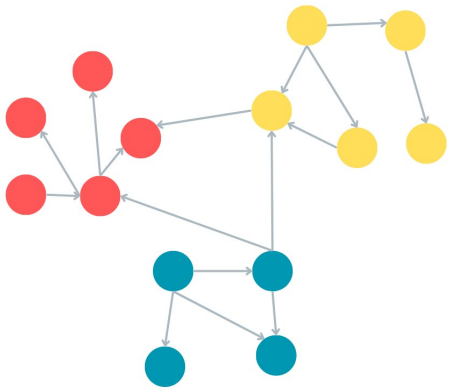
Based on the concept of ***Coupling***, describe the two graphs below: what type of coupling does each of the graphs depict, and which one of them is better for software design? Justify with at least two reasons.

Note: Nodes in this graph are classes, whereas the edges define dependencies between classes.

Graph 1:



Graph 2:



Grading instruction

Coupling type name (Number of points: 3)

Best coupling answer (Number of points: 3)

Justification (Number of points: 4)

Correct justification with at least two reasons.

Question 13 – RANK_PATTERNS_YCS – 202199.2.0

Rank the following design patterns from most restrictive to least restrictive in terms of how much they constrain the structure or behavior of a class or system.

1. Singleton
2. Visitor
3. Observer

Question 14 – MC_OBS_1 – 201887.1.0

What is the primary purpose of the Observer pattern?

A To abstract instantiation of objects

B To define a one-to-many dependency between objects

C To provide global access to a class instance

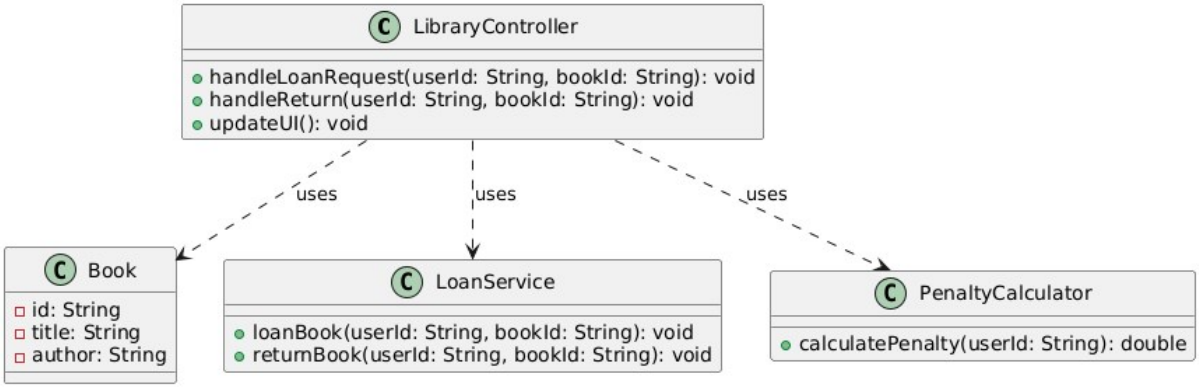
D To decouple interface from implementation

E To control access to the contents of an object

Question 15 – OP_UML_SOLID_1 – 201898.1.1

Consider a library system. Below is a UML class diagram for a feature that manages borrowing and returning books.

The `LibraryController` class is responsible for handling user input, updating the UI, coordinating book loans, checking for late returns, and applying penalties. It directly uses `Book`, `LoanService`, and `PenaltyCalculator`.



Tasks:

1. Identify which SOLID principle is most clearly violated in this design and explain why.
2. Propose a revised design or restructuring, and justify how it improves the design according to that principle.

Grading instruction

Correct Principle (Number of points: 3)

Violation: Primarily the *Single Responsibility Principle*—`LibraryController` mixes presentation logic (UI) with domain and policy logic (loaning, penalties).
A secondary case can be made for *Open/Closed Principle*, since UI changes or policy changes require modifying the same class.

Correct Improvement (Number of points: 7)

Improvement:
- Extract UI updates to a separate `LibraryView`.
- Move loan coordination to a `LibraryService` that delegates to `LoanService` and `PenaltyCalculator`.
- `LibraryController` becomes a thin layer that only passes inputs and outputs.

Question 16 – MC_Specifications_1_AS – 203005.2.1

Select the **WRONG** option about ***software specifications***.

A Specifications act as a contract and help both the implementer and the client. They are also important to guide the testing process.

B Specifications have pre and postconditions. Preconditions define what needs to be true before the method runs. Postconditions define what happens after the method runs, including return values and modifications in objects.

C If the preconditions of a method are not met, the method is obliged to tell the client.

D If the preconditions of a method are satisfied, the implementer must ensure that the postconditions are fulfilled.

Question 17 – MC_RESTAPI_YCS – 202273.1.0

Which of the following best reflects the core philosophy behind designing a RESTful API?

A REST APIs should tightly couple client and server logic to ensure better performance and control.

B REST APIs are primarily about optimizing server-side computations using low-level socket communication.

C REST APIs should expose a uniform interface where resources are represented by URIs and manipulated using standard HTTP methods.

D REST APIs aim to replace HTTP methods with custom operations for better business logic abstraction.

Question 18 – MC_REFACTOR_1 – 201889.2.0

Which of the following is a valid reason to refactor code?

A To increase the number of features

B To rewrite the codebase in a new language

C To improve readability and reduce technical debt

D To temporarily remove unit tests

E To optimize memory at the expense of clarity

Question 19 – MC_OPENCLOSED_1 – 201885.1.0

A violation of the Open/Closed Principle would typically result in:

A The need to modify existing classes for every new requirement

B Code that is too modular

C Reduced performance due to abstraction

D Excessive use of inheritance

E Classes that are open for extension and also for modification

Question 20 − MC_Debugging_2_AS − 203037.2.0

What is the main objective of debugging in software engineering?

A Write code more efficiently

B To identify and fix errors in the software

C Design new software features

D Test and use external libraries