1st Choice of Position Holders

UNIQUE NOTES

COMPUTER SCIENCE & ENTREPRENEURSHIP 11

According to the new SLO-Based Curriculum of PECTAA

UNIQUE PUBLICATIONS
A Project of
UNIQUE GROUP OF INSTITUTIONS
106-A, New Muslim Town, Wahdat Road, Lahore. Tel: 92-42-35865461

CONTENTS

Publications.unique.edu.pk

Unit 01

Introduction to Software Development

- Introduction to SDLC
- Software Development Methodologies
- Introduction to Design Pattern
- Graphical Representation of Software
- Software Debugging & Testing
- Software Development Tools

Q.1 What is software development? 11501001

Ans: Software development is the process of creating computer programs that perform specific tasks. It includes writing code, testing it, and fixing any problems that come up. This process is important because it helps solve problems and makes life easier.

Example: It allows us to chat with friends on social media, manage money using banking apps, and learn through fun educational games.

Q.2 What is the Software Development Life Cycle (SDLC)? 11501002

Ans: The Software Development Life Cycle (SDLC) is a step-by-step process used to develop software from its initial idea to its final deployment and maintenance. Its main goal is to create high-quality software that meets customer needs, stays within budget, finishes on time, and works efficiently.

Q.3 What is a framework in software development? Explain. 11501003

Ans: A framework in software development is like a toolbox that provides ready-to-use tools, rules, and structures to help developers build software faster and more efficiently. It includes pre-made components so developers don't have to start from scratch. This saves time, ensures consistency, and makes the software easier to maintain.

Example: Imagine you want to create a website. Instead of writing all the code from scratch, you can use a framework like Django (for websites). Django comes with ready-made features like user login, database management, and page templates.

Q.4 Discuss the various stages (phases or steps) involved in the Software Development Life Cycle (SDLC). 11501004

Ans: Stages (Phases or Steps) involved in SDLC

The SDLC consists of several stages (phases or steps), each with specific goals and activities. These stages are interconnected and ensure that the software is developed in an organized and efficient manner.

1) Requirement Gathering

This is the first and most critical phase of the SDLC. Its primary objective is to understand what the software needs to achieve and gather detailed information about user expectations. This involves interacting with stakeholders, such as end-users, clients, and business analysts, to identify the software's purpose and functionality.

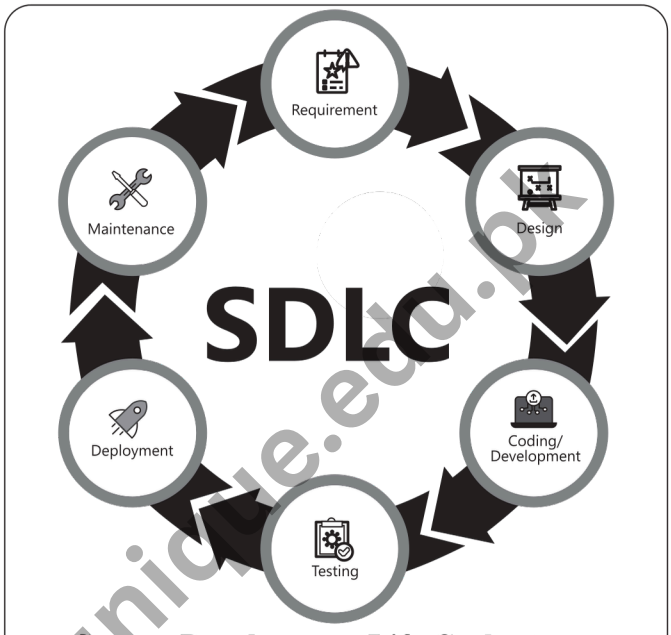
publications.unique.edu.pk

Key Activities in this Phase:

- **Interviews and Surveys:** Conducting interviews and surveys to collect feedback from potential users.
- **Observations:** Performing observations to analyze how users interact with existing systems.
- **Document Review:** Reviewing existing documents, such as reports, user manuals, and technical specifications, to gather additional insights.

Functional and Non-Functional Requirements

Requirements are generally categorized into two types, functional and non-functional requirements.



System Development Life Cycle stages

Functional Requirements

Functional requirements describe the specific behaviors or functions of a system. These requirements outline what the system should do and include tasks, services, and functionalities that the system must perform.

They define the interactions between the system and its users or other systems.

Example: Some functional requirements for a Library Management System are:

- **User Registration:** The system should allow users (students and faculty) to register and create an account.
- **Book Borrowing:** The system should enable users to search for books and borrow them.
- **Inventory Management:** Librarians should be able to add, update, and remove books from the inventory.

Non-Functional Requirements

Non-functional requirements define the quality attributes, performance criteria, and constraints of the system. These requirements specify how the system performs a function rather than what the system should do.

Example: Some non-functional requirements for a Library Management System are:

- **Performance:** The system should handle up to 1000 simultaneous users without performance decline.
- **Reliability:** The system should be available 99.9% of the time, ensuring high availability and minimal downtime.
- **Security:** User data should be encrypted, and access should be controlled through secure authentication mechanisms.

publications.unique.edu.pk

Differentiating Functional and Non-Functional Requirements

Functional	Non-Functional
Define specific behaviors or functions of the system	Define the quality attributes and constraints of the system
What the system should do	How the system should perform
Directly related to user interactions and system tasks	Related to system performance, usability, reliability, etc.

Comparison between Functional and Non-Functional Requirements

2) Design

Once the requirements are gathered, the next step is the Design phase, where developers create a blueprint for the software. This phase focuses on planning how the software will look, function, and interact with users and other systems.

Key Activities in this Phase

- **Create Diagrams:** Creating diagrams (e.g., flowcharts, UML diagrams) to visualize the software's structure and workflows.
- **Develop Models:** Developing models and mockups of the user interface to show how the software will appear to users and how it will look like.
- **Plan the Architecture:** Planning the architecture of the software, including how different components will interact and communicate.
- **Specify Requirements:** Specifying detailed design requirements to ensure all features are planned out accordingly.

TIDBITS

Q. Why is the design phase often compared to creating blueprints for a house in software development?

Ans: The design phase serves as the foundation for software development, much like blueprints guide the construction of a house. It involves visualizing and planning the structure, layout, and functionality of the software before actual development begins. This ensures clarity, organization, and alignment with user requirements, just as blueprints ensure a house is built according to plan and meets the needs of its residents.

3) Coding/Development

In the Development phase, programmers write the actual code for the software based on the design specifications. This is where the software begins to take shape.

Programmer will write code in a programming language (e.g., Python, Java, C#)

4) Testing

Testing phase is a critical step where the software is strictly checked for bugs, errors, and issues. The goal is to ensure that the software works as expected and meets all requirements.

Key Activities in this Phase

- **Functionality Testing:** Ensuring all functions of software work according to expectations.
- **Performance Testing:** Ensuring software performance how software behaves under various conditions such as high (heavy) data traffic load.

publications.unique.edu.pk

- **Compatibility Testing:** Running compatibility testing to ensure the software works across different devices, operating systems, and browsers.

5) Deployment

Once the software has been thoroughly tested and all issues have been resolved, it moves to the Deployment phase. In this phase, the software is installed and made available for users to access and use.

Key Activities in this Phase

- **Installation:** The software is installed on the user's system or server. This may involve running an installation program that copies files and sets up necessary configurations.
- **Configuration:** The software is adjusted to fit the specific needs of the user or organization. It can include setting up user preferences, network settings, and database connections.
- **Testing in the Real World:** After installation, the software is tested in its real-world environment to ensure it works correctly with other systems and meets user needs.

6) Maintenance

The final phase of the SDLC is Maintenance, where the software is continuously monitored, updated, and improved. This phase ensures that the software remains functional and adapts to changes in user needs or technology.

Key Activities in this Phase

- Fixing bugs or errors that arise after deployment.
- Adding new features or enhancements based on user feedback.
- Updating the software to address security vulnerabilities or compatibility issues with new technologies.

Q.5 What are software development methodologies? **11501005**

Ans: Software development methodologies are structured approaches that guide the planning, creation, and management of software projects. These methodologies ensure that the development process is systematic, efficient, and produces high-quality software. They provide a framework for teams to follow, helping them manage tasks, timelines, and resources effectively.

Q.6 Explain the importance of software process models in software development. **11501006**

Ans: **Software Process Models**

Software process models are abstract representations of the steps involved in the **Software Development Life Cycle (SDLC)**. These models help structure and control the development process, ensuring that software is built systematically and meets user needs.

Importance

The importance of software process models lies in their ability to provide:

- **Predictability:** By following a defined process, teams can predict outcomes and manage risks more effectively.
- **Efficiency:** Structured methodologies reduce wasted effort by streamlining workflows.
- **Quality:** Adhering to a process model ensures that quality assurance practices are integrated throughout the development lifecycle.

publications.unique.edu.pk

Q.7 What is the Waterfall model, and what are its benefits and limitations (drawback)? 11501007

Ans: Waterfall Model

The Waterfall model is one of the earliest and most straightforward software development methodologies. It follows a linear and sequential approach, where each phase of the project must be completed before moving on to the next.

Main Phases of Waterfall Model

Think of it as a waterfall flowing from one stage to the next without going back. The main phases of the Waterfall model are:

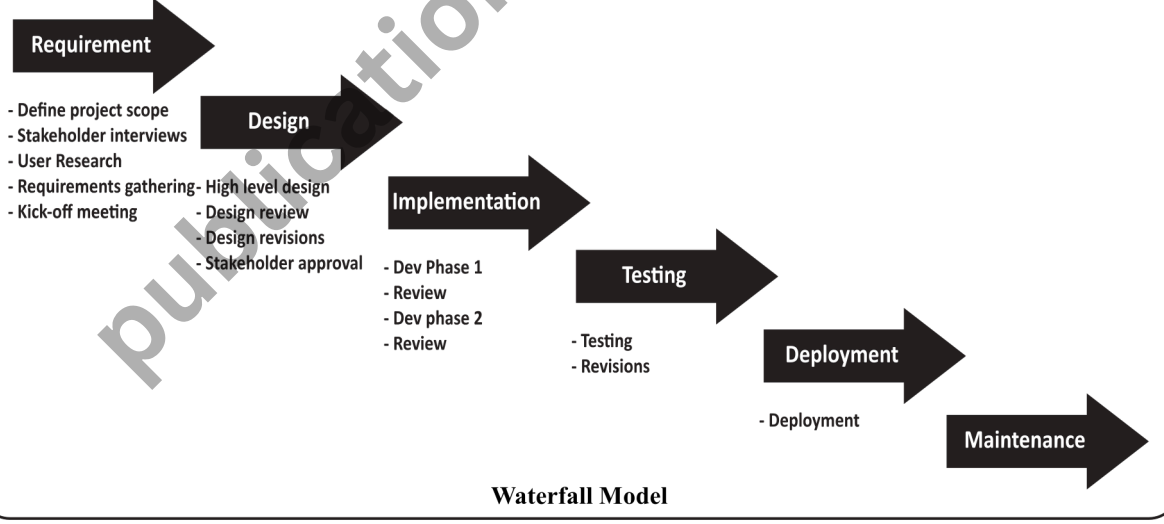
1) **Requirements:** Gather and document what the software needs to achieve.
2) **Design:** Plan how the software will be built, including its architecture and user interface.
3) **Coding:** Write the actual code to create the software based on the design specifications.
4) **Testing:** Check and fix any problems or bugs (errors) in the software.
5) **Deployment:** Release the software for users to access and use.
6) **Maintenance:** Make updates and fix issues that arise after the software is in use.

Benefits

1) **Simple and Easy to Understand:** The Waterfall Model is straightforward, with clear, distinct phases that are easy to follow.
2) **Sequential Process:** Each phase is completed one at a time, making it easier to manage and track progress.
3) **Suitable for Small Projects:** Works well for projects with clear, fixed requirements where changes are unlikely.

Limitations

1) **Inflexibility:** Once a phase is completed, going back to make changes is difficult and costly.
2) **Not Ideal for Complex Projects:** For large and complex designs, this model can be challenging to use effectively.
3) **Risk and Uncertainty:** The model assumes that all requirements are known from the start, which can be risky if new needs or issues arise later in the process.

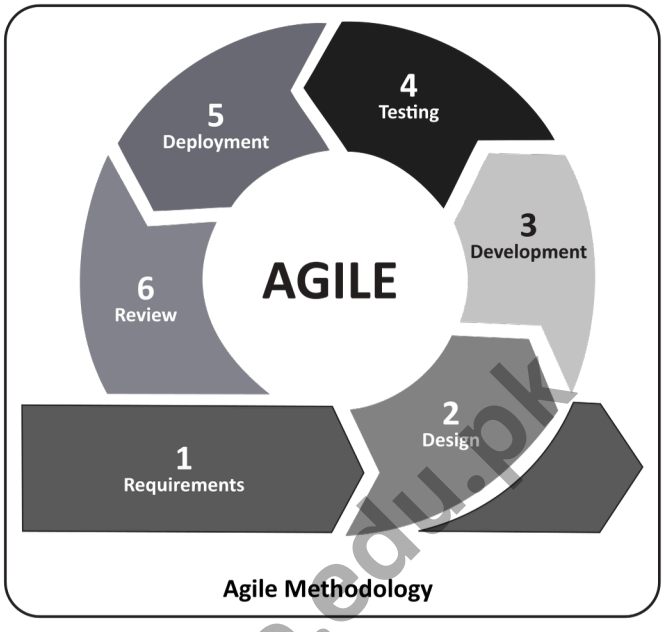


Waterfall Model

publications.unique.edu.pk

Q.8 What is Agile Methodology? Explain its key practices, benefits and limitations. 11501008

Ans: Agile Methodology is a flexible and adaptive approach to software development that emphasizes delivering small, functional parts of the software quickly and adapting to changes as the project progresses. Unlike the waterfall model, which follows a strict sequence of steps.



Agile Methodology

Agile focuses on iterative development through short cycles called iterations or sprints. These cycles enable teams to deliver working software rapidly and gather feedback early, ensuring continuous improvement throughout the project lifecycle.

Key Practices in Agile

Agile includes several practices that enhance collaboration, quality, and adaptability:

- **Continuous Integration:** Code changes are added regularly to one place. This helps to find and fix problems early.
- **Test-Driven Development (TDD):** Tests are written before the actual code, ensuring the software meets requirements and functions correctly.
- **Pair Programming:** Two developers collaborate at one workstation, with one writing code and the other reviewing it in real-time to improve quality and share knowledge.

Benefits

1) **High Flexibility:** Agile allows for changes in requirements even after development has started, making it easier to adapt to new needs or feedback.
2) **Improved Customer Satisfaction:** Frequent delivery of working software ensures customers can see progress and provide feedback regularly, leading to a product that better aligns with their expectations.

Limitations

1) **Scaling Challenges:** Managing large projects with multiple teams can be difficult due to the need for careful coordination and communication.
2) **Stakeholder Involvement:** Agile requires active participation from all stakeholders, which can be challenging if some are unavailable or not fully engaged.
3) **Less Predictable:** Since Agile projects evolve through feedback and changes, predicting the exact timeline and scope of the final product can be harder compared to traditional models like Waterfall.

publications.unique.edu.pk

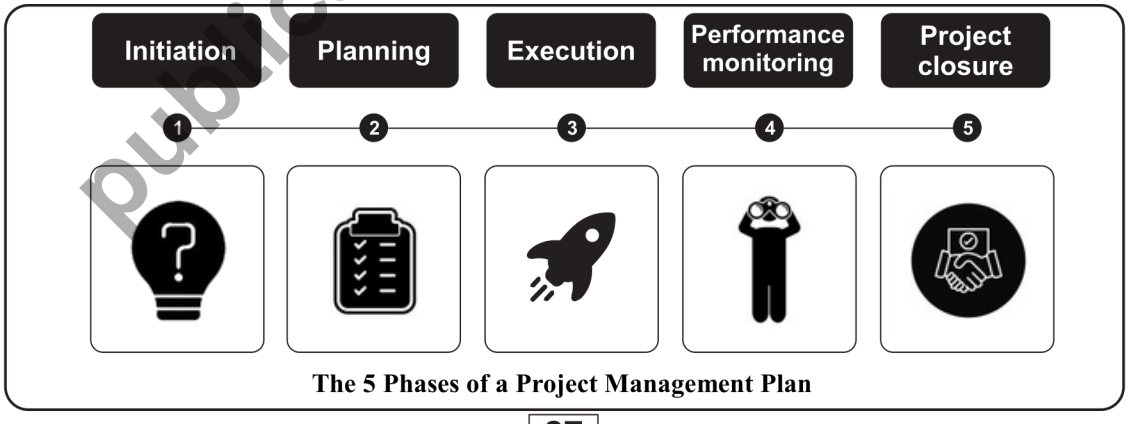
Q.9 Compare and Contrast Waterfall model and Agile methodology. 11501009

Ans:

Feature	Waterfall Model	Agile Methodology
Model Type	Linear and sequential	Iterative and incremental
Phases	Requirements → Design → Implementation → Testing → Maintenance	Iterative sprints with Planning, Development, Testing, Review
Flexibility to Change	Not flexible; hard to implement changes once started	Highly flexible; embraces changes at any stage
Customer Involvement	Limited; mainly at start and end	Continuous involvement throughout the project
Testing Phase	Occurs after development is complete	Continuous testing throughout development
Delivery Timeframe	Single final delivery	Frequent deliveries in small, functional increments
Focus	Fulfilling the predefined plan	Delivering working software and adapting to feedback
Risk Handling	Higher risk due to rigid process	Lower risk due to continuous feedback and adaptability
Documentation	Extensive and detailed documentation	Prioritizes working software over documentation
Best Suited For	Fixed-scope projects (e.g., government, regulatory)	Dynamic-scope projects (e.g., startups, evolving apps)
Communication	Siloed teams; limited phase-to-phase communication	Cross-functional teams; regular daily communication
Approach to Errors	Harder and costlier to fix issues found late	Issues are detected and resolved early and frequently

Q.10 What are the important steps involved in project planning and management for a software project? 11501010

Ans: Planning a software project is like planning a trip—you need to know your destination (project goals), the time it will take (timelines), and the budget required (cost estimation). Comprehensive project planning ensures that all details are considered before starting, reducing risks and ensuring smooth execution.



The 5 Phases of a Project Management Plan

publications.unique.edu.pk

Comprehensive Project Planning

Comprehensive project planning involves thinking through all aspects of the project before implementation. This includes:

- **Understanding Requirements:** Clearly defining what needs to be done to meet user needs.
- **Assigning Roles:** Identifying who will perform each task, such as developers, designers, and testers.
- **Defining Processes:** Outlining how tasks will be completed, including tools and methodologies to be used.

This step is critical because it sets the foundation for the entire project, ensuring clarity and alignment among team members.

DO YOU KNOW?

Q. What makes software companies like Microsoft so valuable?

Ans: Software companies are valued highly due to scalable products, recurring revenues, innovation, and strong eco-systems. Their success underscores software's essential role in driving global digital transformation and economic growth.

Setting Project Timelines

Setting project timelines involves deciding how long each phase or task will take. Timelines help keep the project on track and ensure timely delivery.

Example: Website Development Timeline

- **Design Phase:** 2 weeks
- **Content Writing:** 1 week
- **Testing Phase:** 1 week

Timelines provide structure and accountability, allowing teams to monitor progress and address delays promptly.

Estimating Costs

Estimating the cost of a software project is a critical step in project planning and management. It involves predicting the total expenses required to complete the project successfully. Accurate cost estimation helps with budgeting, resource allocation, and setting realistic expectations.

Key Factors in Cost Estimation

Key factors involved in cost estimation include:

- **Development Team:** The cost depends on the number of developers, their expertise, and their hourly rates.
- **Technology Stack:** The choice of technology, programming languages, and tools can affect the cost. Some technologies require more resources or specialized knowledge.
- **Project Duration:** Longer projects generally required higher costs due to prolonged resource engagement and potential changes in scope.
- **Risk Management:** Identifying potential risks and their mitigation strategies can add to the overall cost. Contingency funds are often included to address unforeseen issues.
- **Quality Assurance:** Costs associated with testing, bug fixing, and ensuring the software meets quality standards are also part of the estimation.

publications.unique.edu.pk

Q.11 What are the steps involved in Risk assessment and management in a software project? 11501011

Ans: Risk Assessment and Management

Risk assessment and management are critical components of software projects. They involve identifying potential risks that could impact the project's success, analyzing their likelihood and impact, and developing strategies to address them. Effective risk management ensures projects stay on track, within budget, and meet quality standards.

Steps in Risk Assessment and Management

1) **Identify Risks:** List all potential risks that could affect the project. These may include technical risks (e.g., untested technology), operational risks (e.g., resource shortages), or external risks (e.g., market fluctuations).
2) **Analyze Risks:** Evaluate the likelihood of each risk occurring and its potential impact on the project.
3) **Develop Mitigation Strategies:** For each significant risk, create a plan to reduce its likelihood or minimize its impact. Strategies might include adding schedule buffers, securing backup resources, or conducting additional testing.
4) **Monitor and Review:** Continuously monitor the project for new risks and review existing ones to adjust mitigation strategies as needed. This ensures proactive risk management throughout the project lifecycle. 11501012

Q.12 What happens during the Execution phase of software development?

Ans: During execution, the team writes code, creates designs, and builds the software based on the project plan. It requires teamwork, coordination, and regular updates to stay on track and deliver the product successfully.

Q.13 What is quality assurance, and how does it help ensure a software project works properly? 11501013

Ans: Quality assurance (QA) makes sure a project meets the required standards and works the way it should. It uses simple steps like testing the software, checking the code for errors, getting feedback from users or stakeholders, and keeping track of the project's progress regularly.

Example: QA ensures that the code is written correctly and that the software does what it's supposed to do. This helps avoid mistakes and makes sure the final product is reliable and user-friendly.

Q.14 What is the purpose of graphical representation of software systems? 11501014

Ans: Graphical representation of software systems involves using visual diagrams to represent various aspects of a software system's structure and behavior. This approach helps in simplifying complex systems, making it easier for developers and stakeholders to understand, communicate, and manage the system.

Q.15 What is UML? Describe Use Case Diagram in details. Also explain how to identify use cases. 11501015

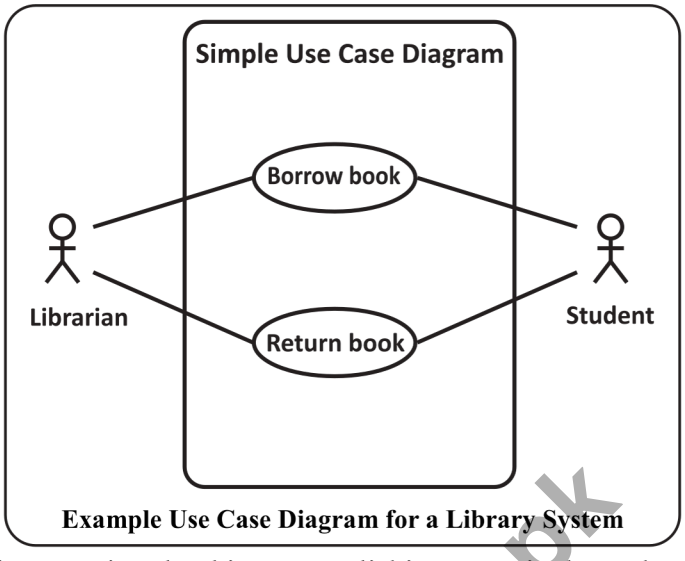
Ans: Unified Modeling Language (UML) is a standardized way to visually represent the design of a software system. It simplifies understanding and managing complex systems by providing clear visuals of how components interact.

publications.unique.edu.pk

Types of UML Diagrams

There are four types of diagrams:

1) Use Case diagram
2) Class diagram
3) Sequence diagram
4) Activity diagram



Example Use Case Diagram for a Library System

Use Case Diagrams

A use case is a description of a set of interactions between a user (actor) and a system to achieve a specific goal. Use cases are identified based on the functionalities that the system must support to meet the user's needs. Each use case represents a complete workflow from the user's perspective, detailing the steps involved in accomplishing a particular task.

Purpose of use case diagram

Use Case Diagrams are used for several purposes:

1) **Capturing Functional Requirements:** They help in identifying and documenting the functional requirements of the system.
2) **Understanding User Interactions:** They illustrate how different users will interact with the system.
3) **Planning and Testing:** They aid in planning the development process and in designing test cases for validating system functionalities.

Identifying Use Cases

The process of identifying use cases involves several steps:

1) **Identify Actors:** Determine who or what interacts with the system (e.g., users or other systems).
2) **Define Goals:** Identify the goals or tasks actors need to accomplish.
3) **Outline Interactions:** Describe how actors interact with the system to achieve these goals.
4) **Validate Use Cases:** Review use cases with stakeholders to ensure accuracy.

CLASS ACTIVITY **11501015 (a)**

Statement: Imagine you are designing an online shopping platform. The platform allows customers to browse products, add items to their cart, and make purchases. Additionally, the platform includes features for administrators to manage product listings, process orders, and handle customer inquiries. There is also a feature for delivery personnel to update the status of deliveries.

In the above class activity, you can compare your findings with the following:

- **Actors**
  Customer Administrator Delivery Personnel
- **Use Cases**
  - Browse Products
  - Add Items to Cart

publications.unique.edu.pk

- Make Purchase
- Manage Product Listings
- Process Orders
- Handle Customer Inquiries
- Update Delivery Status

Solution

In this class activity, you are tasked with designing an online shopping platform and identifying the actors and use cases involved. The actors in the system include:

- **Customer:** Browses products, adds items to their cart, and makes purchases.
- **Administrator:** Manages product listings, processes orders, and handles customer inquiries.
- **Delivery Personnel:** Updates the status of deliveries.

The corresponding use cases for these actors are:

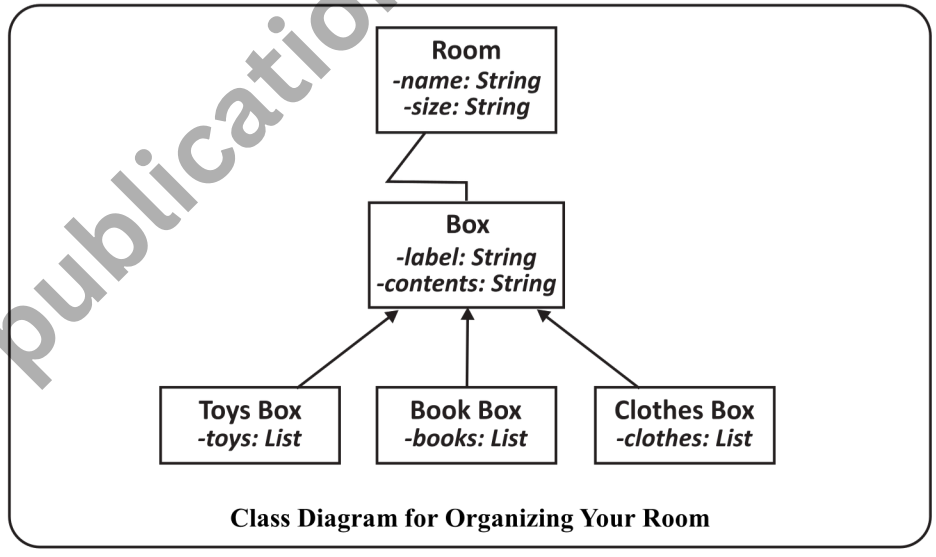
- **Customer:** Browse Products, Add Items to Cart, Make Purchase.
- **Administrator:** Manage Product Listings, Process Orders, Handle Customer Inquiries.
- **Delivery Personnel:** Update Delivery Status.

Q.16 What is a Class Diagrams? Explain with the help of an example. 11501016

Ans: A Class Diagram is like a map that shows how things are organized in a system.

Example: Room Organization

- **Room:** Represents the overall space encompassing all other elements, analogous to the main structure in a class diagram.
- **Box:** Serves as a container within the room, akin to a class in a diagram.
- **Attributes:** Each box contains specific items, such as a 'ToyBox' holding toys or a 'BookBox' containing books.
- **Methods:** Boxes can perform actions like 'open' or 'close,' similar to methods in a class diagram that define what the box can do.
- **Specific Boxes:** Examples of specialized boxes include a 'ToyBox' for toys, a 'BookBox' for books, and a 'ClothesBox' for clothes, representing distinct instances of the general 'Box' class.



Class Diagram for Organizing Your Room

publications.unique.edu.pk

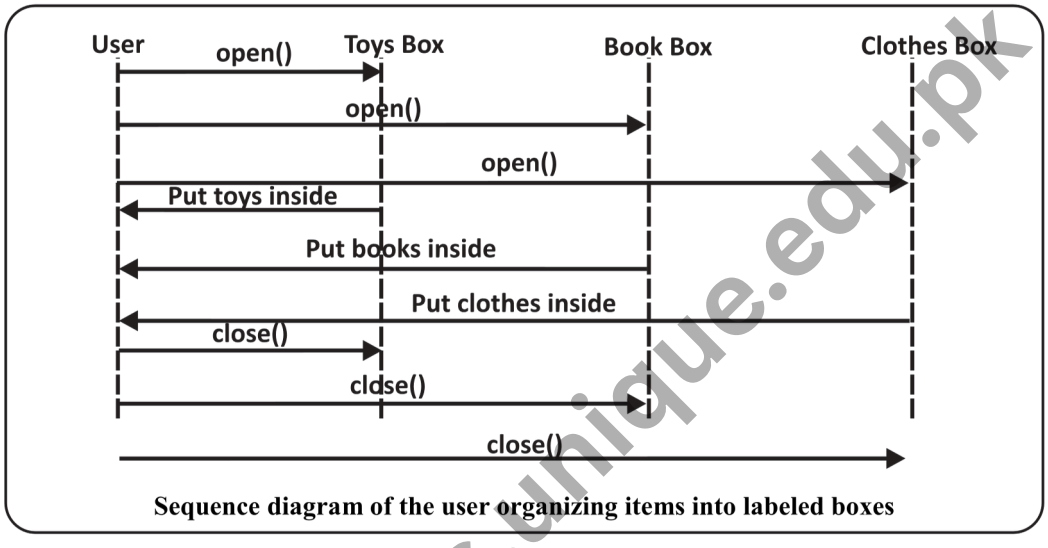
Q.17 What are Sequence Diagrams? Explain with the help of an example. 11501017

Ans: Sequence diagrams are visual tools that show how objects in a system interact with each other in a specific order. They focus on the flow of messages or actions between objects over time, helping to clarify the sequence of events in a process.

Example: Interactions Between Objects (Boxes)

- **open():** The user opens a box to access its contents.
- **put toys/books/clothes inside:** The user places specific items (toys, books, or clothes) into the respective boxes.
- **close():** The user closes the box after adding the items.

These interactions represent the sequence of actions performed by the user (an object) and the boxes (other objects) over time.



Sequence diagram of the user organizing items into labeled boxes

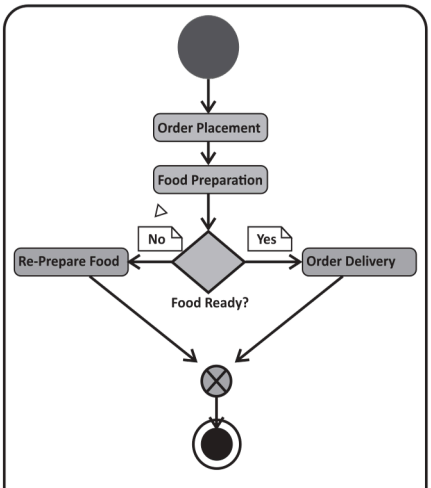
Q.18 What are Activity Diagrams? Explain with the help of an example. 11501018

Ans: Activity diagrams are visual tools used to illustrate the flow of activities or steps in a process. They are particularly useful for modeling the logic of complex operations by showing how tasks progress from start to finish.

Example: Restaurant Management System

In a restaurant management system, an activity diagram can represent the process of handling a customer order. The process includes the following steps:

1) **Start:** The process begins.
2) **Order Placement:** The customer places an order.
3) **Food Preparation:** The kitchen prepares the food based on the order.
4) **Order Delivery:** The prepared food is delivered to the customer.
5) **End:** The process concludes.



Activity diagram with decision and connector symbol

publications.unique.edu.pk

The arrows in the diagram indicate the sequence of these steps, making it easy to understand how an order progresses through the system.

Q.19 How is UML diagram used in different stages of software development to enhance understanding and communication? **11501019**

Ans: Practical Applications of UML

UML (Unified Modeling Language) plays a vital role in various stages of software development by improving clarity, collaboration, and planning. There are some key applications:

- **Planning:** UML diagrams are used to map out the system's requirements and design before any code is written. This helps developers and stakeholders visualize the system's structure and functionality early in the process.
- **Development:** During development, UML diagrams serve as a reference for developers to understand the system's structure, relationships between components, and how different parts of the system interact.
- **Communication:** UML diagrams act as a common language that bridges technical and non-technical team members. They help stakeholders, including clients and managers, understand how the system works without needing deep technical knowledge.

Q.20 Explain the concept of design patterns in software development. Discuss some commonly used design patterns. **11501020**

Ans: Design Patterns

Design patterns are common solutions to common problems that arise during software development. They act as blueprints or templates that developers can adapt to solve design challenges efficiently. By using design patterns, developers can create systems that are flexible, maintainable, and easy to understand.

Commonly Used Design Patterns

1) **Singleton Pattern**

The Singleton design pattern is a way to make sure that a specific object or resource is created only once in a program and reused whenever needed.

Example: Database Connection

This ensures only one database connection is created throughout the application lifecycle. Any attempt to create another instance just returns the existing one.

2) **Factory Pattern**

The Factory design pattern is like having a special workshop that knows how to create different products, but you don't need to worry about the details of how those products are made. Instead, you just tell the factory what you need, and it gives you the finished product.

Example: Vehicle Factory

The factory handles object creation based on input, hiding implementation details from the user.

3) **Observer Pattern**

The Observer design pattern is like having a group of people who are interested in getting updates from one particular source. Whenever something important happens, the source automatically notifies all the interested people. It's a way to keep things in sync without everyone constantly checking for updates.

publications.uniqe.edu.pk

Example: News Publisher and Subscribers
When the publisher sends out news, all subscribers receive it automatically.

4) Strategy Pattern

The Strategy design pattern is like having a toolbox full of different tools, each designed for a specific job. When you face a problem, you can pick the right tool from the box based on the task at hand.

Example: Payment Strategies

The Shopping Cart uses different payment strategies depending on what's passed to it at runtime.

CLASS ACTIVITY 11501020(a)

Identify a real-world scenario around you where you can apply one of these design patterns.

Solution

In this activity, the Observer pattern is applied to an online food delivery app to manage real-time updates for customers and drivers. The order status acts as the "subject," while users (customers and drivers) act as "observers" who are notified of changes like order confirmation or delivery status. This pattern ensures loose coupling, scalability, and dynamic updates, enhancing the system's efficiency and user experience. It demonstrates how design patterns like the Observer pattern can solve real-world problems in systems requiring one-to-many communication.

Q.21 What are the applications of Design Patterns in Software design? 11501021

Ans: Applications of Design Patterns in Software Design

Design patterns are widely used in software development to solve common problems and create robust and maintainable code. They help in:

- **Reducing Code Complexity:** Providing a clear and organized structure to the code.
- **Enhancing Code Reusability:** Using proven solutions to avoid reinventing the wheel.
- **Improving Communication:** Offering a common vocabulary among developers to discuss design decisions effectively.

By applying design patterns, developers can create robust, maintainable, and scalable systems that are easier to adapt to changing requirements.

DO YOU KNOW?

Q. What is an example of a design pattern used in popular software frameworks?

Ans: The Model-View-Controller (MVC) pattern is widely used in frameworks like Ruby on Rails and Angular.

Q.22 Why are software debugging and testing critical stages in the development process? 11501022

Ans: Software debugging and testing are critical stages in the software development process that ensure the quality and reliability of a software product. Effective debugging and testing help developers to confirm that the software meets the required specifications, functions as intended, and is free of critical errors.

publications.unique.edu.pk

Q.23 What is debugging, and why is it important in software development? 11501023

Ans: Debugging is the process of finding and fixing bugs or errors in a software. Software debugging and testing are essential parts of software development. They ensure that the software works correctly and meets the user's requirements.

Bugs are errors or mistakes in the software that cause it to behave unexpectedly. Identifying bugs involves observing the software's behavior and finding the source of the problem. Once identified, solving bugs requires making changes to the code to correct the error.

Tools and Best Practices

There are various tools and best practices for debugging, including:

- **Debuggers:** Software tools that help programmers find bugs by allowing them to step through code, inspect variables, and monitor program execution.
- **Print Statements:** Adding print statements in the code to display the values of variables at different points in the program.
- **Code Reviews:** Having other developers review your code to spot potential errors.

CLASS ACTIVITY 11501023(a)

Try writing a unit test for a simple function in your favorite programming language.

Solution: Write a unit test for a simple function that adds two numbers.

Code Example (Python)

```
# math_functions.py
def add_numbers(a, b):
    return a + b
# test_math_functions.py
import unittest
from math_functions import add_numbers
class TestMathFunctions(unittest.TestCase):
    def test_add_numbers(self):
        self.assertEqual(add_numbers(2, 3), 5)
        self.assertEqual(add_numbers(-1, 1), 0)
        self.assertEqual(add_numbers(0, 0), 0)
if __name__ == '__main__':
    unittest.main()
```

Running the Test

Save both files and run `test_math_functions.py`. You'll see output indicating whether the tests passed.

Explanation

- We define a function `add_numbers`.
- We write a test class `TestMathFunctions` that inherits from `unittest.TestCase`.
- Inside the class, we define a test method `test_add_numbers` to check various inputs.
- `assertEqual` checks if the actual output matches the expected result.

publications.unique.edu.pk

Q.24 What is Testing? Explain its types. 11501024

Ans: Testing

Testing is the process of checking if the software works as it should and meets the requirements. It starts with testing small parts and moves step-by-step to testing the whole system, including getting feedback from users.

Types of Testing

There are four types of testing:

1) **Unit Testing:** Unit testing is the first level of testing, where individual components or modules of the software are tested in isolation. Each unit is a small, testable part of the software, such as a function or method. The primary goal of unit testing is to verify that each component works correctly according to its design and performs as expected.
2) **Integration Testing:** After unit testing Integration Testing is performed to evaluate the interaction between different components or modules. While unit testing focuses on isolated units, integration testing ensures that these units work together correctly when combined. This type of testing checks for interface errors, data flow between modules and other integration-related issues.
3) **System Testing:** System Testing is a higher level of testing where the entire software system is tested as a whole. At this stage, the software is treated as a complete entity, and testers evaluate its overall functionality, performance, security, and compliance with specified requirements.
4) **Acceptance Testing:** Acceptance Testing is conducted to determine whether the software is ready for release. It is often performed by the end-users or clients to ensure that the software meets their expectations and requirements.

DO YOU KNOW?

Q. Why is acceptance testing also called User Acceptance Testing (UAT)?

Ans: Because it is typically performed by the end-users to ensure the software meets their requirements.

Q.25 What are software development tools? 11501025

Ans: Software development tools are essential for creating, testing, and maintaining software applications.

These tools help developers write code, find and fix errors, and manage software projects effectively.

Q.26 Discuss the purpose and examples of language editors, translators, debuggers, and IDEs in the software development process. 11501026

Ans: Language editors, also known as code editors, are tools that help developers write and edit code in different programming languages. The purpose of language editors is to provide a user-friendly interface for writing code.

Examples

- **Notepad++:** A simple yet powerful code editor.
- **VS Code:** A popular editor with many extensions.

publications.unique.edu.pk

Translators

Translators are tools that convert code written in one programming language into another language that the computer can understand. Translators convert high-level programming languages (like Python) into machine language (binary code) that computers can execute.

Types of Language Translators

It has two types:

1) **Interpreters:** Translate code line-by-line (e.g., Python interpreter).
2) **Compilers:** Translate the entire code at once (e.g., GCC for C/C++).

Debuggers

Debuggers are tools that help developers find and fix errors (bugs) in their code. The purpose of debuggers is to allow developers to test their code and identify where errors occur.

Examples

- **GDB:** GNU Debugger for C/C++.
- **Visual Studio Debugger:** Integrated with Visual Studio IDE.

Integrated Development Environments (IDEs)

IDEs are comprehensive software suites that provide all the tools needed for software development in one place. IDE integrate various development tools like editors, compilers, debuggers, and version control systems to streamline the development process. An IDE offers a unified interface where developers can write, test, and debug their code efficiently.

Common IDEs

- **Visual Studio:** Popular for .NET and C++ development.
- **PyCharm:** Preferred for Python development.

Q.27 What are online and offline computing platforms? **11501027**

Ans: Online and Offline Computing Platforms

These platforms provide environments where developers can write, run, and test their code.

- **Online Platforms:** Cloud-based platforms accessible via the internet (e.g., Repl.it, Gitpod).
- **Offline Platforms:** Local development environments on a computer (e.g., local installations of IDEs).

Q.28 Define Source Code Repositories. **11501028**

Ans: Source Code Repositories

Source code repositories are platforms where developers can store, manage, and track changes to their code. Repositories help in version control, allowing multiple developers to work on the same project without conflicts. Repositories keep track of code changes and maintain a history of all modifications.

Examples

- **GitHub:** Popular platform for open-source projects.
- **Bitbucket:** Used for both private and public repositories.

publications.unique.edu.pk

Solved Exercise

Multiple Choice Questions (MCQs)

1. **Primary purpose of the Software Development Life Cycle (SDLC) is to:** 11501029
   a) design websites
   b) deliver high-quality software within time and cost estimates
   c) manage database systems
   d) create hardware components
2. **A type of requirement specifies system performance:** 11501030
   a) Functional Requirements
   b) Non-Functional Requirements
   c) Technical Requirements
   d) Operational Requirements
3. **Role of a framework in the context of SDLC is to:** 11501031
   a) write code from scratch
   b) provide a structured foundation with predefined components and architectures
   c) manage hardware
   d) perform manual testing
4. **Software development model involves short cycles or sprints:** 11501032
   a) Waterfall Model
   b) Agile Methodology
   c) Lean Software Development
   d) Scrum
5. **Crucial aspect of comprehensive project planning:** 11501033
   a) Understanding the project scope and tasks
   b) Deciding the project's colour scheme
   c) Hiring a large development team
   d) Ignoring potential risks
6. **Factor that does not influence the cost estimation of a software project:**
   a) Scope of the project 11501034
   b) Technology stack
   c) Number of meetings held
   d) Operational costs
7. **The purpose of Use Case Diagrams is to:** 11501035
   a) document the system's architecture.
   b) identify and document the system's functional requirements.
   c) illustrate the database schema.
   d) define the system's user interface design.

Answers

1	b	**2**	b	**3**	b	**4**	b
5	a	**6**	c	**7**	b		

Short Questions

Q.1 Differentiate between functional and non-functional requirements. 11501036

Ans: Difference between Functional and Non-Functional Requirements

Functional	Non-Functional
Define specific behaviors or functions of the system	Define the quality attributes and constraints of the system
What the system should do	How the system should perform
Directly related to user interactions and system tasks	Related to system performance, usability, reliability, etc.

Publications.unique.edu.pk

Q.2 Explain why the testing phase is important in the SDLC, and provide two reasons for its significance. 11501037

Ans: The testing phase checks if the software works as planned and meets user needs.

Two Reasons for its Significance

1) **Improves Quality:** Finds and fixes errors early, making the software reliable.
2) **Saves Money and Trouble:** Fixing issues during development is cheaper than fixing them after the software is released.

Q.3 Illustrate the concept of continuous integration in agile methodology and discuss its importance in software development. 11501038

Ans: Continuous Integration (CI)

Developers regularly add their code to a shared project, often multiple times a day. Automated tests check the code to find problems quickly.

Importance

1) **Catches Problems Early:** Frequent checks help find errors fast, so they're easier to fix.
2) **Better Teamwork:** Developers share code often, avoiding conflicts and working better together.
3) **Faster Delivery:** Quick feedback and automation help release features sooner.

Q.4 Evaluate the main steps involved in risk assessment and management, and assess their importance in a software project.

Ans: Steps in Risk Management: 11501039

1) **Identify Risks:** List all potential risks that could affect the project. These may include technical risks (e.g., untested technology), operational risks (e.g., resource shortages), or external risks (e.g., market fluctuations).
2) **Analyze Risks:** Evaluate the likelihood of each risk occurring and its potential impact on the project. This helps prioritize risks and focus on those that pose the greatest threat.
3) **Develop Mitigation Strategies:** For each significant risk, create a plan to reduce its likelihood or minimize its impact. Strategies might include adding schedule buffers, securing backup resources, or conducting additional testing.
4) **Monitor and Review:** Continuously monitor the project for new risks and review existing ones to adjust mitigation strategies as needed. This ensures proactive risk management throughout the project lifecycle.

Importance

- Prevents surprises that could derail the project.
- Helps make smart choices about time and resources.
- Increases the chances of finishing the project successfully.

Q.5 Explain the purpose of a use case diagram in software development. 11501040

Ans: A Use Case Diagram shows how users interact with a system and what features it offers.

Purpose

1) **Shows What the System Does:** Helps everyone understand the system's features.
2) **Improves Communication:** Easy for both tech and non-tech people to understand.
3) **Guides Development:** Helps the team know what to build.
4) **Sets Boundaries:** Clarifies what the system will and won't do.

publications.unique.edu.pk

Q.6 Compare and contrast a sequence diagram with an activity diagram. 11501041

Ans: Difference between Sequence diagrams and Activity Diagrams

Aspect	Sequence Diagram	Activity Diagram
Focus	Object interactions over time	Workflow or process flow
Represents	Message exchange between objects	Steps, decisions, and control flow
Use Case	Modeling behavior in a single scenario	Describing business logic or workflows
Time	Time flows vertically (top to bottom)	Time is not explicitly represented
Best For	Visualizing object collaboration	Visualizing procedural or decision-based logic
Example:	Shows how a user and server communicate to log in.	Shows the steps to process an online order.

Q.7 Describe the Factory Pattern and explain how it differs from directly creating objects, with an example. 11501042

Ans: Factory Pattern

A way to create objects using a special "factory" class instead of making them directly in the code.

Difference between Factory pattern and Creating Objects Directly

Aspect	Factory Pattern	Direct Object Creation
Object Creation	Done via a factory method	Done using new keywords directly
Flexibility	High – easy to change object creation logic	Low – changes require modifying all instantiations
Code Coupling	Low – depends on abstraction	High – tightly coupled with specific classes
Maintainability	Easier to manage and update	Harder to scale and update
Usage Scenario	When object creation is complex or needs to vary	When simple, one-time object creation is sufficient
Design Principle	Follows Open/Closed Principle	Violates Open/Closed Principle

Long Questions

Q.1 Design a flowchart for a user registration process in a software application. Outline its key steps. 11501043

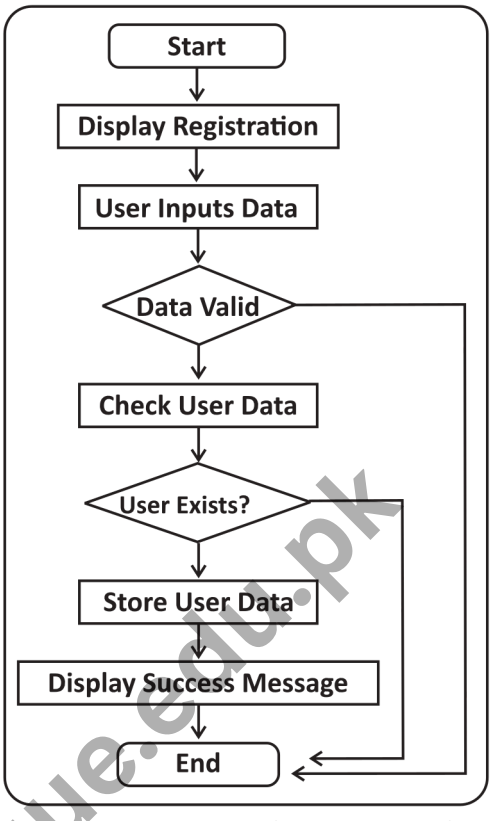
Ans: Flowchart for User Registration Process

Key Steps

1) **Start:** The process begins.
2) **Display Registration Form:** The user is given a form to fill out. They need to enter details like name, email, and password.

publications.uniquie.edu.pk

3) **User Inputs Data:** The user types in their information.
4) **Validate Data:** The system makes sure all fields are filled and the data is correct (like a valid email and strong password).
   - **If data is invalid:** Go back to step 3 so the user can fix the mistakes.
   - **If data is valid:** Move on to step 5.
5) **Check If User Already Exists:** The system checks if someone with the same email or username is already registered.
   - **If user exists:** Show an error message and return to step 3 so the user can try a different email or username.
   - **If user does not exist:** Move on to step 6.
6) **Store User Data:** If everything is okay, the user's details are saved in the database.
7) **Display Success Message:** A message appears saying something like "Registration Successful."
8) **End:** The registration process is done.



Q.2 Imagine you are managing a project to develop a simple mobile application. Describe how you would use the Agile Methodology to handle this project. 11501044

Ans: Using Agile Methodology for Mobile Application Development Approach

To build a simple mobile app, I'd use the Agile approach with these easy steps:

1) **Break the Project into Small Tasks**
   - Split the app into smaller pieces, like making a login screen, signup page, user profile, or settings. This makes the work easier to manage.
2) **Work in Short Timeframes (Sprints)**
   - Plan tasks in short cycles, called sprints, lasting 1–2 weeks. Each sprint focuses on completing specific app features.
3) **Hold Short Daily Meetings**
   - Have quick daily check-ins (stand-ups) where the team shares:
     - What they worked on yesterday
     - What they'll do today
     - Any problems they're facing
4) **Get Customer Feedback After Each Sprint**
   - Show the customer a working part of the app after every sprint. Use their feedback to make changes quickly.
5) **Test as You Go**
   - Test each feature right after it's built to catch and fix errors early.

publications.unique.edu.pk

6) **Keep Improving**
   - After each sprint, review what went well and what could be better. Use this to improve teamwork and the app's quality.

Q.3 Consider an online banking system. Create a Use Case Diagram to show the interactions between customers, bank staff, and the system. **11501045**

Ans: Use Case Diagram for Online Banking System

A Use Case Diagram shows how different users (called actors) interact with a system's features. In an Online Banking System, actors like Customers, Cashiers, Managers, and the Bank itself perform various tasks.

Actors in the Online Banking System

1) **Customer:** A person using the bank's services, such as opening accounts or requesting loans.
2) **Cashier:** A bank staff member who updates account balances, processes credit/debit transactions, and manages loan details.
3) **Manager:** A senior staff member who approves loans and oversees bank staff.
4) **Bank:** A general entity handling core operations like approving accounts and managing transactions.

Actors and Their Tasks (Use Cases)

Customer

1. Add Account **2.** Check Balance **3.**Transfer Money **4.** Request Loan

Cashier

1. Update Balance **2.** View Loan Terms **3.** Process Credit/Debit Transactions

Manager

1. Approve Loan **2.** Manage Staff **3.** Handle Promotions and Salaries

Bank

1. Add Account **2.** Check Balance **3.** Transfer Money **4.** Approve Account **5.** Monitor Transactions

Each actor is connected to the tasks they can perform in the system.

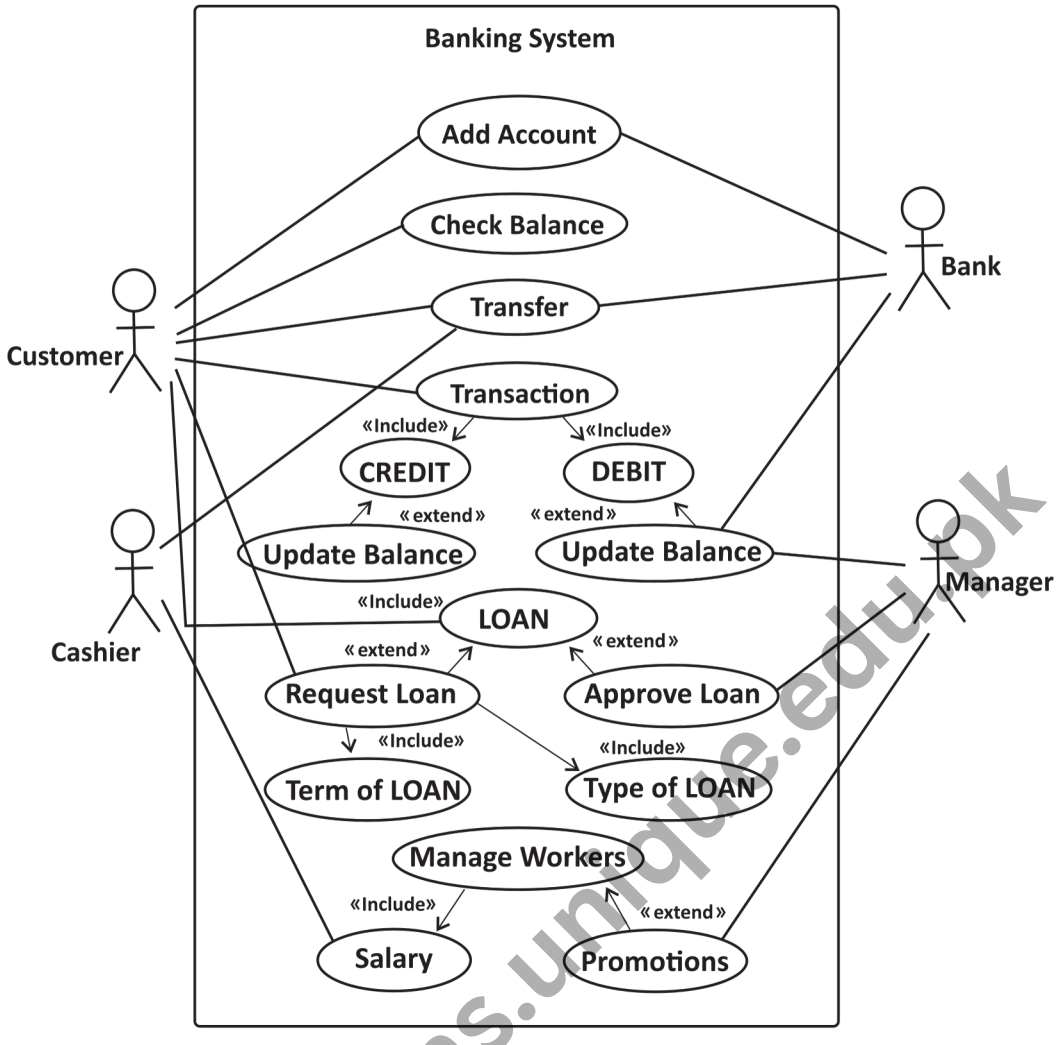
Use Case Diagram Connections

1. **Customer is linked to:** Add Account, Check Balance, Transfer Money, Request Loan.
2. **Cashier is linked to:** Update Balance, View Loan Terms, Credit/Debit Transactions.
3. **Manager is linked to:** Approve Loan, Manage Staff, Promotions, Salary.
4. **Bank is linked to:** Add Account, Check Balance, Transfer Money, Approve Account, Monitor Transactions.

Use of <<include>> and <<extend>>

- **<<include>>:** Used when one task always requires another.
  Example: A "Transaction" always includes "Credit" or "Debit" because every transaction involves one of these actions.
- **<<extend>>:** Used when a task optionally adds to another.
  Example: A "Loan" task might include "Approve Loan" or "Check Interest," but these only happen when needed.

publications.unique.edu.pk



Q.4 You are developing a food delivery application. Create a Sequence Diagram to show the process of placing an order, from the customer selecting items to the delivery of the order. **11501046**

Ans: Objects

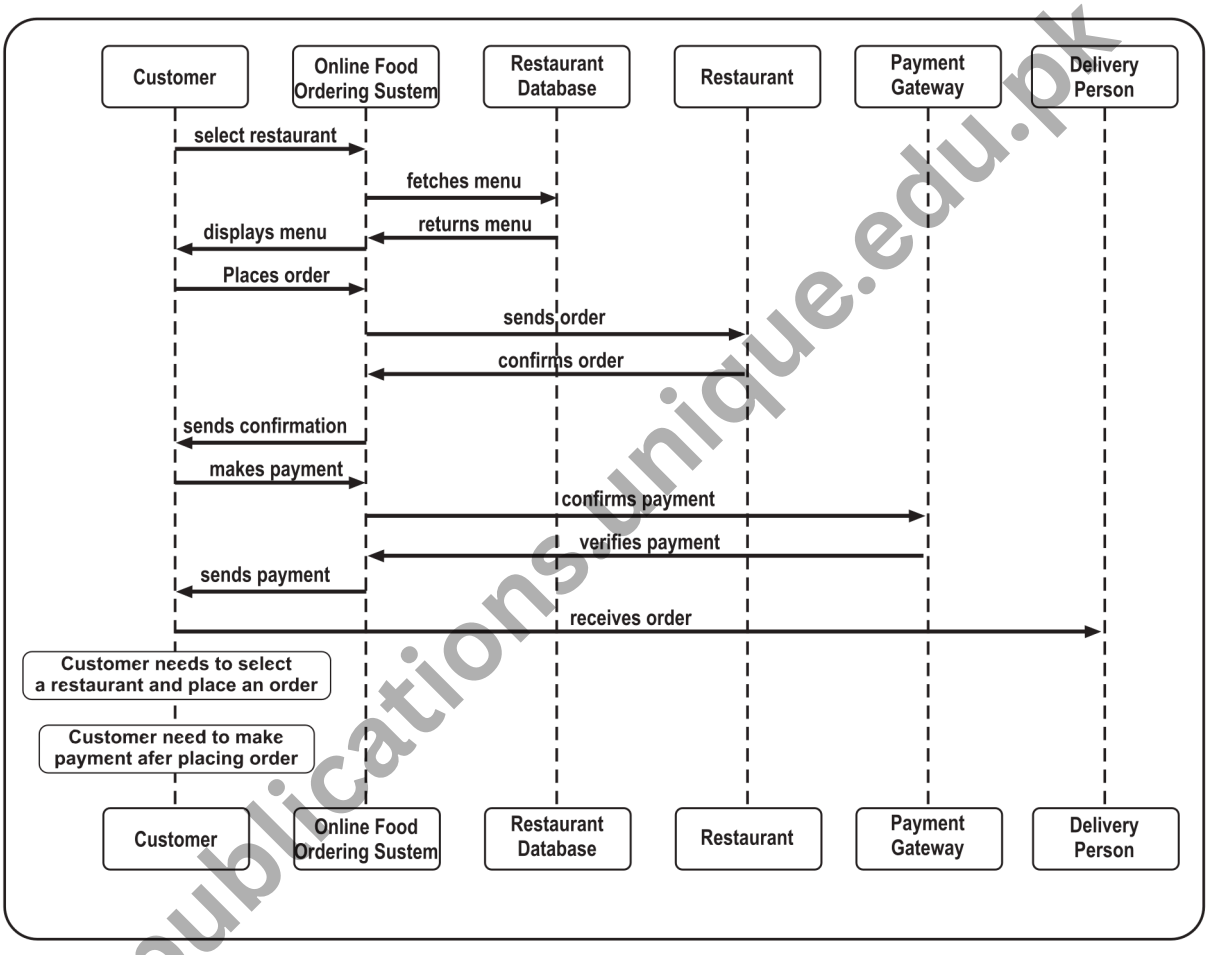
- **Customer:** Initiates the order.
- **Mobile App:** Interface for placing orders.
- **Order Service:** Manages order processing.
- **Restaurant:** Prepares the order.
- **Delivery Service:** Assigns a driver.
- **Driver:** Delivers the order.

Sequence

- Customer browses menu on Mobile App.
- Customer selects items and submits order to Mobile App.
- Mobile App sends order details to Order Service.

publications.unique.edu.pk

- Order Service validates order and sends confirmation to Mobile App.
- Mobile App displays confirmation to Customer.
- Order Service notifies Restaurant of the order.
- Restaurant confirms order preparation to Order Service.
- Order Service requests delivery from Delivery Service.
- Delivery Service assigns Driver and confirms to Order Service.
- Order Service notifies Mobile App of delivery status.
- Mobile App updates Customer with delivery status.
- Driver picks up order from Restaurant.
- Driver delivers order to Customer.
- Customer confirms delivery via Mobile App.
- Mobile App notifies Order Service of completion.



publications.unique.edu.pk

Q.5 Discuss the importance of software development tools in the software development process. 11501047

a) Explain the role of language editors, translators, and debuggers in creating and maintaining software.

b) Provide examples of each tool and describe how they contribute to the

efficiency and accuracy of software development.

Ans: Importance of Software Development Tools

Software development tools streamline the development process, improve code quality, reduce errors, and enhance collaboration. They automate repetitive tasks, enforce standards, and provide insights into code performance, enabling developers to focus on creative problem-solving.

a) Role of Language Editors, Translators, and Debuggers Language Editors

- **Role:** Provide an environment for writing and editing code with features like syntax highlighting, autocompletion, and code formatting.
- **Contribution:** Enhance productivity by reducing typing errors, improving code readability, and suggesting code improvements.

Translators (Compilers/Interpreters)

- **Role:** Convert high-level code into machine-readable instructions (compilers translate entire programs; interpreters execute code line-by-line).
- **Contribution:** Enable execution of programs, catch syntax errors during compilation, and optimize code for performance.

Debuggers

- **Role:** Allow developers to step through code, inspect variables, and identify runtime errors or logical issues.
- **Contribution:** Speed up error resolution, ensure software reliability, and reduce the risk of bugs in production.

b) Examples and Contributions

Language Editor: Visual Studio Code (VS Code)

- **Contribution:** Supports multiple languages, offers extensions for code analysis and formatting, and integrates with Git for version control, improving coding efficiency and collaboration.

Translator: GCC (GNU Compiler Collection)

- **Contribution:** Compiles C/C++ code into executables, provides detailed error messages, and optimizes code for performance, ensuring accurate and efficient programs.

Debugger: GDB (GNU Debugger)

- **Contribution:** Allows breakpoint setting, variable inspection, and stack trace analysis, enabling developers to pinpoint and fix bugs quickly, enhancing software reliability.

Short Question Answers (Additional)

Software Development

Q.1 What is software development? 11501049

Ans: Software development is the process of creating computer programs that perform specific tasks. It includes writing code, testing it, and fixing any problems that come up. This process helps solve real-world problems and makes life easier through technology.

Introduction to SDLC

Q.2 Why is the Software Development Life Cycle (SDLC) important? 11501050

Ans: The SDLC provides a structured and systematic approach to software development. It helps ensure that software meets user requirements, stays within budget, and is delivered on time and with

publications.unique.edu.pk

high quality. By following the SDLC, teams can reduce risks, improve planning, and manage resources more efficiently.

Q.3 What is a framework in software development? 11501051

Ans: A framework provides ready-to-use tools and structures to help developers build software faster and more efficiently. It includes pre-made components so developers don't have to start from scratch. This saves time, improves consistency, and makes maintenance easier.

Q.4 Give an example of a web development framework. 11501052

Ans: Django is a popular framework used for building websites quickly. It comes with built-in features like login systems, database management, and page templates. Using Django helps developers avoid writing code from scratch and speeds up the development process.

Q.5 What is the purpose of Requirement Gathering in SDLC? 11501053

Ans: Requirement gathering is the first phase of SDLC where developers understand what users need from the software. It involves interviews, surveys, and observations to collect detailed information. Accurate requirements ensure the final product meets user expectations and avoids costly rework.

Q.6 How many phases are SDLC? 11501054

Ans: The Software Development Life Cycle (SDLC) consists of six phases:

1) Requirement Gathering
2) Design
3) Coding (or Implementation)
4) Testing
5) Deployment
6) Maintenance

Q.7 What are functional requirements? 11501055

Ans: Functional requirements describe what the system should do, such as user registration or order processing. They define specific behaviors and interactions between the system and users. These requirements are essential for outlining core functionalities during development.

Q.8 What are non-functional requirements? 11501056

Ans: Non-functional requirements define how the system performs, including speed, reliability, and security. They include constraints like response time, availability, and data protection. These requirements ensure the system works well under real-world conditions.

Q.9 How do functional and non-functional requirements differ? 11501057

Ans: Functional requirements focus on what the system does, while non-functional requirements focus on how well it does it. Functional ones relate to user actions and system tasks, whereas non-functional ones deal with performance, usability, and reliability.

Q.10 What happens during the Design phase of SDLC? 11501058

Ans: In the design phase, developers create blueprints for the software using diagrams and models. They plan the architecture, layout, and interaction between system components. This phase ensures clarity before actual coding begins.

Q.11 Why is the design phase compared to creating house blueprints? 11501059

Ans: Like blueprints guide the construction of a house, the design phase guides software development by visualizing

publications.unique.edu.pk

the structure and layout. It ensures all team members understand the system's flow and functionality. This reduces confusion and errors during coding.

Q.12 What occurs during the Coding phase? 11501060

Ans: During the coding phase, programmers write the actual code based on design specifications. They use programming languages like Python, Java, or C# to implement features. This phase turns design into a working software solution.

Q.13 What is the Testing phase in SDLC? 11501061

Ans: The Testing phase checks the software for bugs, errors, and issues to ensure it works correctly. It includes functionality, performance, and compatibility tests. Testing confirms that the software meets all user requirements and works smoothly.

Q.14 Why is testing important in software development? 11501062

Ans: Testing helps detect bugs early, making them cheaper and easier to fix. It also ensures the software behaves as expected and satisfies user needs. Without proper testing, software may fail after release, leading to poor user experience and reputational damage.

Q.15 What is the Deployment phase? 11501063

Ans: Deployment is when the software is installed and made available for users to access and use. It includes configuration, installation, and testing in the real environment. The goal is to ensure the software works as intended in live conditions.

Q.16 What is the Maintenance phase of SDLC? 11501064

Ans: Maintenance involves updating, improving, and fixing issues in the software after deployment. It includes adding new features, patching security flaws, and enhancing performance. This phase ensures the software remains useful over time.

Software Development Methodologies

Q.17 What are software development methodologies? 11501065

Ans: Methodologies are structured approaches that guide how software is developed. Examples include Waterfall, Agile, and DevOps. They provide frameworks for managing tasks, timelines, and team collaboration.

Q.18 What role do software process models play in development? 11501066

Ans: Process models bring predictability, efficiency, and quality to software projects. They help teams follow a clear path, manage risks, and maintain standards throughout development. Models like SDLC ensure consistent results across different projects.

Q.19 What is the Waterfall model? 11501067

Ans: The Waterfall model is a linear and sequential approach to software development. Each phase must be completed before moving to the next. It works best for small projects with fixed requirements.

Q.20 What are the benefits of the Waterfall model? 11501068

Ans: It is simple to understand and easy to manage due to its linear structure. Progress tracking is straightforward, and documentation is thorough. It suits projects with clearly defined goals and minimal changes expected.

Q.21 What are the limitations of the Waterfall model? 11501069

Ans: Once a phase is complete, going back to make changes is difficult and costly. It is not suitable for large or complex projects

publications.unique.edu.pk

with evolving requirements. It assumes all requirements are known upfront, which is rarely the case in real-world scenarios.

Q.22 What is Agile Methodology? 11501070

Ans: Agile is a flexible and iterative approach to software development. It focuses on delivering small parts of the software quickly and adapting to changes along the way. Teams work in sprints and involve users regularly for feedback.

Q.23 What are key practices in Agile methodology? 11501071

Ans: Agile uses continuous integration, test-driven development, and pair programming. These practices improve code quality, encourage teamwork, and allow for frequent updates based on user input.

Q.24 What are the benefits of Agile? 11501072

Ans: Agile allows for flexibility and quick adaptation to changing requirements. It boosts customer satisfaction by involving them throughout the process. Delivering working software in short cycles ensures continuous improvement.

Q.25 What are the limitations of Agile? 11501073

Ans: Managing large-scale projects with multiple teams can be challenging. It requires active involvement from stakeholders, which may not always be possible. Predicting timelines and budgets can be harder due to evolving scope.

Project Planning and Management

Q.26 What is comprehensive project planning? 11501074

Ans: Project planning involves defining goals, setting timelines, assigning roles, and estimating costs. It ensures that all aspects of the project are considered before starting. Good planning reduces risks and increases chances of success.

Q.27 How do timelines contribute to successful project execution? 11501075

Ans: Timelines provide structure and help track progress during development. They set expectations for when each task should be completed. Timely delivery becomes easier when milestones are clearly defined.

Q.28 Why is cost estimation important in project planning? 11501076

Ans: Cost estimation helps determine the budget required for the project. It considers factors like team size, technology, and risk management. Accurate estimates prevent financial surprises and help secure stakeholder approval.

Q.29 What is risk assessment in software projects? 11501077

Ans: Risk assessment involves identifying potential threats to a project's success. It evaluates the likelihood and impact of these risks. Proactive identification helps teams prepare mitigation strategies.

Q.30 How does risk management contribute to project success? 11501078

Ans: By continuously monitoring and addressing risks, teams can avoid major setbacks. Risk management ensures smoother execution, better resource allocation, and timely delivery. It builds confidence among stakeholders.

Q.31 What happens during the Execution phase? 11501079

Ans: During execution, the team writes code, designs interfaces, and builds the software. It requires coordination, communication, and regular updates to ensure everything stays on schedule. This is where the software starts taking shape.

publications.unique.edu.pk

Q.32 What is Quality Assurance (QA)? 11501080

Ans: QA ensures that the software meets required standards and functions properly. It involves testing, code reviews, and feedback collection. QA helps identify and fix issues before the software reaches users.

Q.33 How does QA help ensure software works properly? 11501081

Ans: QA catches bugs early, improves code quality, and validates user experience. It ensures the software performs reliably under various conditions. Without QA, defects may go unnoticed until after release.

Graphical Representation of Software Systems

Q.34 What is graphical representation of software systems? 11501082

Ans: Graphical representation uses diagrams to show how a system is structured and how its parts interact. It simplifies complex ideas and improves communication between developers and stakeholders. Tools like UML are commonly used for this.

Q.35 What is UML and why is it important? 11501083

Ans: UML (Unified Modeling Language) is a standardized way to visually represent software design. It helps developers understand system behavior and structure. UML improves clarity, especially when working in teams.

Q.36 How many types of UML diagrams? 11501084

Ans: There are four main types: Use Case, Class, Sequence, and Activity diagrams. Each type serves a different purpose in modeling different aspects of a system. These diagrams support both planning and documentation.

Q.37 What is a Use Case Diagrams? 1150185

Ans: A Use Case Diagram shows how users interact with a system to achieve goals. It identifies actors and their interactions with the system. This diagram helps clarify functional requirements and system boundaries.

Q.38 What is a Class Diagrams? 1150186

Ans: A Class Diagram represents the structure of a system by showing classes, their attributes, and methods. It acts as a blueprint for object-oriented design. It helps organize code and understand relationships between objects.

Q.39 What are Sequence Diagrams? 1150187

Ans: Sequence Diagrams show the sequence of interactions between objects over time. They help visualize how messages flow between components. These diagrams are useful for understanding system behavior step-by-step.

Q.40 What are Activity Diagrams? 1150188

Ans: Activity Diagrams model the flow of activities or steps in a process. They are useful for visualizing business workflows or system logic. They show decision points, parallel processes, and overall control flow.

Q.41 How is UML used in different stages of software development? 11501089

Ans: UML supports planning by modeling requirements, designing system structure, and documenting code. During development, it helps developers understand system components. It also aids communication between technical and non-technical stakeholders.

Introduction to Design Patterns

Q.42 What are design patterns in software development? 1501090

Ans: Design patterns are ready-made solutions to common problems faced during software design. They act like templates that

publications.unique.edu.pk

help developers write better, organized, and easy-to-maintain code.

Q.43 Why do developers use design patterns? 1501091

Ans: Developers use design patterns to solve problems quickly without starting from scratch. These patterns make the code more flexible, reusable, and easier to understand by everyone in the team.

Q.44 Name some commonly used Design Patterns. 11501092

Ans: Common patterns include Singleton, Factory, Observer, and Strategy. Singleton ensures one instance of a class exists. Factory handles object creation. Observer notifies dependent objects of changes. Strategy allows interchangeable algorithms.

Q.45 What is the Singleton Pattern? 11501093

Ans: The Singleton pattern ensures that only one object of a class is created in the whole program. For example, it's used to create a single database connection that can be reused whenever needed.

Q.46 What is the Factory Pattern? 11501094

Ans: The Factory pattern helps create objects without showing how they are made. Just like a factory produces different items, this pattern gives the right object based on user input, hiding complex details behind a simple interface.

Q.47 Explain the Observer Pattern with an example. 11501095

Ans: In the Observer pattern, one object notifies others when something changes. For example, subscribers get automatic updates from a news publisher whenever new content is posted.

Q.48 What is the Strategy Pattern used for? 11501096

Ans: The Strategy pattern allows using different methods or strategies to solve similar tasks. For example, a shopping cart can use different payment methods like credit card or PayPal depending on what the user chooses.

Software Debugging and Testing

Q.49 What is Debugging? 11501097

Ans: Debugging is the process of finding and fixing errors in the code. It involves analyzing program behavior and identifying the root cause of issues. Effective debugging ensures the software runs as intended.

Q.50 Why is Debugging important in software development? 11501098

Ans: Bugs can cause crashes, incorrect outputs, or security vulnerabilities. Debugging helps resolve these issues early, saving time and money. It ensures software reliability and user satisfaction.

Q.51 What is testing in software development? 11501099

Ans: Testing is checking if a software works properly and does what it's supposed to do. It helps find mistakes before users start using it.

Q.52 Why is testing important? 11501100

Ans: Testing makes sure the software has no big errors and works well. It helps avoid problems like crashes or wrong results after the software is released.

Q.53 Define types of Testing. 11501101

Ans: Types of testing include Unit, Integration, System, and Acceptance Testing. Unit tests individual components, Integration tests how modules work together, System tests the whole application, and Acceptance tests whether it meets user needs.

Q.54 What is unit testing? 11501102

Ans: Unit testing is when small parts of the software, like one function at a time, are

publications.unique.edu.pk

tested. This helps make sure each part works correctly on its own.

Q.55 What is integration testing? 11501103

Ans: Integration testing checks how different parts of the software work together. It finds issues that happen when these parts are connected.

Q.56 What is system testing? 11501104

Ans: System testing checks the whole software as one complete system. It makes sure everything works well together and meets all the user's needs.

Q.57 What is acceptance testing? 11501105

Ans: Acceptance testing is done by real users or clients to see if the software is ready to use. It checks if the software meets their expectations.

Q.58 Which type of testing comes first? 11501106

Ans: Unit testing is done first. After that, integration testing, then system testing, and finally acceptance testing.

Q.59 Who usually does acceptance testing? 11501107

Ans: Acceptance testing is usually done by end users or clients, not developers. They check if the software works well for them before accepting it.

Software Development Tools

Q.60 What are software development tools? 11501108

Ans: Software development tools are programs that help developers write, test, and manage code. They make it easier to create software by providing helpful features like error checking and code organization.

Q.61 What is a code editor? Give two examples. 11501109

Ans: A code editor is a tool used to write and edit programming code. It makes coding easier with features like color-coding and auto-completion. Examples are Notepad++ and Visual Studio Code (VS Code).

Q.62 What do translators do in software development? 11501110

Ans: Translators convert code written by humans into code that computers can understand. There are two types: compilers, which translate all code at once, and interpreters, which translate line by line.

Q.63 What is a debugger? Why is it useful? 11501111

Ans: A debugger is a tool that helps find and fix errors in code. It lets developers see what's happening inside the program while it runs. This makes fixing problems faster and easier.

Q.64 What is an IDE? Name two popular IDEs. 11501112

Ans: An IDE (Integrated Development Environment) is a complete package that includes tools like code editor, compiler, and debugger in one place. It makes coding more efficient. Two examples are Visual Studio and PyCharm.

Q.65 Define Source Code Repositories. 11501113

Ans: Source code repositories store and manage code versions. Platforms like GitHub and Bitbucket allow developers to collaborate, track changes, and maintain history. They are essential for version control and teamwork.

publications.unique.edu.pk

Multiple Choice Questions (Additional)

Software Development

1. What is the main goal of software development? 11501114
a) Design hardware
b) Create task-specific programs
c) Manage databases
d) Develop marketing plans

Introduction to SDLC

2. Which phase of SDLC involves understanding user needs and expectations? 11501115
a) Design b) Testing
c) Requirement Gathering
d) Deployment

3. What is a framework in software development? 11501116
a) A language
b) Coding rules
c) Testing tool
d) Pre-built structure

4. Which of the following is a web development framework? 11501117
a) Java b) Python
c) Django d) MySQL

5. What is the benefit of using frameworks in development? 11501118
a) Slower b) More bugs
c) Faster with reusable parts
d) Less secure

6. Which of the following is a functional requirement? 11501119
a) The system should be fast
b) The system should allow user login
c) The system should be secure
d) The system should be scalable

7. Which of the following is a non-functional requirement? 11501120
a) User registrations
b) Data encryption
c) Online payment
d) Report generation

8. What is the purpose of the Design phase in SDLC? 11501121
a) To write code
b) To gather user feedback
c) To create a blueprint of the system
d) To test the software

9. Which SDLC phase involves installing the software for users? 11501122
a) Testing b) Coding
c) Deployment d) Maintenance

10. Which SDLC phase includes continuous updates and bug fixes? 11501123
a) Design
b) Development
c) Maintenance
d) Requirement Gathering

11. Which of the following ensures software meets user expectations? 11501124
a) Debugging b) Testing
c) Deployment d) Documentation

12. Which SDLC phase creates visual models of the system? 11501125
a) Requirement b) Design
c) Coding d) Testing

13. Which of the following is NOT part of SDLC? 11501126
a) Requirement gathering
b) Coding c) Marketing
d) Testing

14. Which phase comes after Testing in SDLC? 11501127
a) Requirement Gathering
b) Deployment
c) Design d) Debugging

15. Which of the following is a reason for testing software? 11501128
a) To make it look better
b) To find and fix bugs early
c) To delay delivery
d) To increase costs

publications.unique.edu.pk

Software Development Methodologies

16. Why is the Waterfall model not suitable for dynamic projects? 11501129
a) It requires too many resources
b) It follows a rigid, linear approach
c) It lacks documentation
d) It is outdated

17. In Agile methodology, work is divided into short cycles called: 11501130
a) Phases b) Sprints
c) Stages d) Modules

18. Which of the following is a key practice in Agile? 11501131
a) Sequential planning
b) Fixed scope
c) Continuous integration
d) No feedback loops

19. Which Agile practice involves writing tests before code? 11501132
a) Pair Programming
b) Continuous Integration
c) Test-Driven Development
d) Documentation First

20. Which Agile practice involves two developers working together? 11501133
a) Test-Driven Development
b) Pair Programming
c) Sprint Planning
d) Daily Standups

21. What is the main disadvantage of the Waterfall model? 11501134
a) High cost
b) Lack of flexibility
c) Poor documentation
d) Slow development

22. Which model works well for small projects with fixed requirements? 11501135
a) Agile b) Spiral
c) Waterfall d) DevOps

23. Which methodology emphasizes customer collaboration and responding to change? 11501136
a) Waterfall b) V-model
c) Agile d) RAD

24. Which model supports iterative development? 11501137
a) Waterfall
b) Agile
c) V-model
d) None of the above

25. Which of the following helps in managing large-scale software projects? 11501138
a) Frameworks
b) Methodologies
c) Tools
d) All of the above

Project Planning and Management

26. What is risk assessment in software projects? 11501139
a) Writing code
b) Identifying potential threats
c) Creating diagrams
d) Managing teams

27. What is the first step in risk management? 11501140
a) Analyzing impact
b) Monitoring risks
c) Identifying risks
d) Mitigating risks

28. Which activity happens during the Execution phase? 11501141
a) Gathering requirements
b) Writing code and building software
c) Planning timelines
d) Testing software

29. What is the purpose of project planning? 11501142
a) To ignore user needs
b) To define goals, roles, and timelines
c) To skip testing
d) To reduce team size

publications.unique.edu.pk

30. Which factor affects software cost estimation? 11501143
a) Team size
b) Number of stakeholders
c) Technology stack
d) All of the above

31. Which of the following is a quality assurance activity? 11501144
a) Coding
b) Code reviews and testing
c) Marketing
d) Hiring

Graphical Representation of Software Systems

32. What is the role of UML in software development? 11501145
a) Writing code
b) Managing team communication
c) Visualizing system design
d) Deploying software

33. Which UML diagram shows how users interact with the system?
a) Class Diagram 11501146
b) Use Case Diagram
c) Sequence Diagram
d) Activity Diagram

34. Which UML diagram represents the structure of a system using classes and objects? 11501147
a) Use Case Diagram
b) Class Diagram
c) Sequence Diagram
d) Activity Diagram

35. What is the purpose of a Sequence Diagram? 11501148
a) Show data flow
b) Show object interactions
c) Model workflows
d) Define system limits

36. What does an Activity Diagram illustrate? 11501149
a) System architecture
b) Flow of activities in a process
c) Code structure
d) Database schema

37. Which of the following is a static diagram in UML? 11501150
a) Sequence Diagram
b) Use Case Diagram
c) Activity Diagram
d) Class Diagram

38. Which of the following is a dynamic diagram in UML? 11501151
a) Class Diagram
b) Use Case Diagram
c) Sequence Diagram
d) Component Diagram

Introduction to Design Patterns

39. Which of the following best describes a design pattern? 11501152
a) New language
b) Reusable solution
c) Type of DB
d) Testing method

40. Which design pattern ensures only one instance of a class exists? 11501153
a) Factory b) Strategy
c) Singleton d) Observer

41. Which design pattern allows interchangeable algorithms at runtime? 11501154
a) Singleton b) Factory
c) Strategy d) Observer

42. Which design pattern helps notify multiple objects about changes in one object? 11501155
a) Factory b) Strategy
c) Observer d) Singleton

43. Which design pattern provides a way to create objects without exposing logic? 11501156
a) Singleton b) Factory
c) Strategy d) Observer

publications.unique.edu.pk

Software Debugging and Testing

44. What is debugging in software development? 11501157
a) Writing new code
b) Finding and fixing errors in code
c) Planning software features
d) Deploying software

45. What is the main purpose of testing in software development? 11501158
a) To write better code
b) To find bugs and ensure quality
c) To document requirements
d) To plan timelines

46. Which type of testing checks individual components of software?
a) Integration Testing 11501159
b) System Testing
c) Unit Testing
d) Acceptance Testing

47. Which testing level checks how different modules work together?
a) Unit Testing 11501160
b) Integration Testing
c) System Testing
d) Acceptance Testing

48. Which testing checks the entire system as a whole? 11501161
a) Unit Testing
b) Integration Testing
c) System Testing
d) Debugging

49. Who usually performs Acceptance Testing? 11501162
a) Developers b) Testers
c) End-users d) Managers

Software Development Tools

50. What is the purpose of source code repositories? 11501163
a) To write code
b) To store and track code changes
c) To compile code
d) To deploy applications

51. Which platform is commonly used for version control? 11501164
a) Visual Studio b) GitHub
c) Notepad++ d) MySQL

52. Which of the following is an online computing platform? 11501165
a) Visual Studio b) Repl.it
c) PyCharm d) Eclipse

53. Which of the following is an offline computing platform? 11501166
a) Gitpod b) Repl.it
c) Local installation of an IDE
d) Cloud

54. What is an IDE? 11501167
a) A code writer
b) Tools in one interface
c) Testing framework
d) Deployment tool

55. Which IDE is commonly used for Python development? 11501168
a) Visual Studio b) PyCharm
c) Notepad++ d) GDB

56. Which tool translates high-level code line-by-line? 11501169
a) Compiler b) Interpreter
c) Assembler d) Debugger

57. Which tool translates the entire code at once? 11501170
a) Interpreter b) Compiler
c) Editor d) Debugger

58. What is the purpose of a debugger? 11501171
a) To write code
b) To find and fix errors
c) To design interfaces
d) To manage databases

59. Which of the following is a code editor? 11501172
a) GCC b) GDB
c) VS Code d) Python

60. Which of the following is a compiler? 11501173
a) Python interpreter b) GCC
c) GDB d) VS Code

publications.unique.edu.pk

Answers

1	b	**2**	c	**3**	d	**4**	c	**5**	c
6	b	**7**	b	**8**	c	**9**	c	**10**	c
11	b	**12**	b	**13**	c	**14**	b	**15**	b
16	b	**17**	b	**18**	c	**19**	c	**20**	b
21	b	**22**	c	**23**	c	**24**	b	**25**	d
26	b	**27**	c	**28**	b	**29**	b	**30**	d
31	b	**32**	c	**33**	b	**34**	b	**35**	b
36	b	**37**	d	**38**	c	**39**	b	**40**	c
41	c	**42**	c	**43**	b	**44**	b	**45**	b
46	c	**47**	b	**48**	c	**49**	c	**50**	b
51	b	**52**	b	**53**	c	**54**	b	**55**	b
56	b	**57**	b	**58**	b	**59**	c	**60**	b

publications.uniqu.edu.pk