

“PROGRAMMING STYLES AND PARADIGMS” COURSE
FINAL EXAM QUESTIONS
(for 3rd year students of the Software Engineering program)

1. What is a programming paradigm, and how does it influence the way we approach software development?
2. What is software architecture, and why is it crucial for successful software development?
3. What is MVC architecture? How does MVC work? Examples of MVC frameworks.
4. What is Client-Server Architecture? How does it work? Key characteristics, advantages and disadvantages.
5. What are the benefits of a client-server architecture in web applications?
6. What is layered architecture? How does it work? Key characteristics, advantages and disadvantages.
7. Describe the differences between layered architecture and microservices architecture.
8. Compare and contrast monolithic and microservices architecture.
9. What is software testing, and why is it an indispensable component of the software development lifecycle?
10. What is parallel computing, and how does it revolutionize the way we approach complex computational problems?
11. Differentiate between parallel computing and distributed computing.
12. Explain the concept of object-oriented programming in detail, including its principles, benefits, and how it differs from other programming paradigms.
13. Explain the concept of attributes in OOP, discussing their significance, how they are defined within classes, and their role in representing the state of an object.
14. Provide a comprehensive explanation of polymorphism in OOP, including its types, practical examples of how it is implemented.
15. Explain the concept of inheritance in OOP, covering its definition, the relationship between parent and child classes, the benefits it offers for code reuse and organization.
16. Provide a detailed explanation of encapsulation in OOP, discussing its definition, the mechanisms used to achieve it, such as access modifiers.
17. Explain the concept of abstraction in OOP, including its definition, the techniques used to implement it, such as abstract classes and interfaces.
18. Explain the concepts of instance variables and instance methods in OOP, including their definitions, examples of how they are utilized within classes.

19. Provide a comprehensive explanation of the process of creating classes in OOP, including detailed explanations of the syntax and structure, the importance of constructors.
20. Explain the concept of method overriding in OOP.
21. What are the steps involved in designing a class in OOP?
22. Compare procedural programming with object-oriented programming.
23. What is functional programming? Explain the advantages and potential challenges associated with using functional programming in software development.
24. Explain the concept of pure functions in functional programming. What are their advantages and disadvantages?
25. Analyze the advantages and disadvantages of different paradigms for specific problem domains.
26. How can higher-order functions enhance the expressiveness and flexibility of code in various programming languages?
27. Provide an explanation of iteration in the context of functional programming.
28. Explain the concept of recursion in the context of functional programming.
29. Explain the concept of type inference in the context of functional programming.
30. Discuss the concept of side effects in programming. How do functional programming languages try to minimize side effects?
31. What are the challenges of iteration in functional programming?
32. What is reflective programming? How does it allow a program to inspect and modify its own structure and behavior at runtime?
33. What is declarative programming? Explain the difference between declarative programming and imperative programming.
34. Discuss the relevance and applications of declarative programming in modern software development.
35. How does logic programming differ from other paradigms? Discuss the role of unification and backtracking in Prolog.
36. What is scripting programming paradigm? How does it differ from other programming paradigms?
37. Describe the characteristics of scripting languages.
38. What is aspect-oriented programming and how does it differ from OOP?
39. What is metaprogramming? Analyze the importance of metaprogramming in modern software development.
40. Discuss the different techniques and approaches used in metaprogramming, including reflection, code generation, and macros.
41. Explain the advantages of metaprogramming, such as increased abstraction, reduced code duplication, and enhanced flexibility in software design.

- 42.Explain the concept of vector programming. Discuss its advantages, disadvantages and applications.
- 43.What is procedural programming? Discuss its advantages, disadvantages and applications.
- 44.Explain the concept of culturally competent programming.
45. How can culturally competent programming manifest in various domains, such as user interface design, localization, accessibility features, and community engagement in software projects?
- 46.Discuss the advantages and disadvantages associated with each processing paradigm, including considerations related to code readability, maintainability, performance, and scalability.
- 47.Explain various processing paradigms and their application domains.
- 48.Explain the concept of event-driven programming.
49. What are low-level programming paradigms? Discuss the key characteristics that distinguish low-level programming from high-level programming.
- 50.Explain the concept of concurrency and parallelism.
- 51.Discuss the key principles that underpin parallel programming, such as the division of tasks into smaller sub-tasks, the coordination of these tasks.
52. Analyze the significance of parallel programming in the context of modern computing environments.
- 53.Explain the importance of unit testing and integration testing in the software development process.
- 54.Describe the stages of the Waterfall model in software development.
- 55.What are the key principles of Agile development methodologies?
- 56.Explain the difference between a stack and a queue and provide real-world examples of their use.
- 57.Implement a binary search tree and explain its advantages over a simple list for searching.
- 58.Explain the differences in time complexity between searching in a binary search tree and a sorted array.
- 59.Discuss the time and space complexity of different data structures (e.g., arrays, linked lists, trees).
- 60.What are the advantages and disadvantages of using linked lists compared to arrays?
- 61.Explain the concept of dynamic programming and provide an example of a problem that can be solved using dynamic programming.
- 62.Describe how dynamic programming can optimize recursive algorithms with overlapping subproblems.
- 63.Explain the concept of program execution control in programming.

64. Describe various techniques of code optimization.
65. What is the difference between synchronous and asynchronous programming?
66. Write a program that asks the user for a number n and prints the sum of the numbers from 1 to n .
67. Write a function that returns the largest element in a list.
68. Implement a function that returns the second largest element in a list.
69. Write a function that checks whether an element occurs in a list.
70. Write a program that prints the elements on odd positions in a list.
71. Write a program that prints the elements on even positions in a list.
72. Write a program that displays the index of an element in a list.
73. Write a program that prints all prime numbers from 1 to 1000.
74. Implement a program that generates the first n prime numbers.
75. Write a function that merges two sorted lists into a new sorted list.
 $[1,4,6,7,9], [2,3,5,8,10] \rightarrow [1,2,3,4,5,6,7,8,9,10]$.
76. Write a function that combines two lists by alternately taking elements, e.g.
 $[a,b,c], [1,2,3] \rightarrow [a,1,b,2,c,3]$.
77. Implement Bubble sort using any programming language.
78. Implement random password generator using letters, numbers and symbols.
79. Implement binary search using any programming language.
80. Implement a lazy Fibonacci sequence generator.
81. Implement the same algorithm (e.g., sorting, searching) in different programming paradigms (functional, OOP, and logic) and compare their efficiency and readability.
82. Create a class to represent a bank account with methods for deposit, withdrawal, and checking balance.
83. Create a program that integrates both object-oriented and functional programming paradigms.
84. Design a class hierarchy for a library system, including classes for books, members, and librarians.
85. Implement a function that takes a list of integers and returns a new list containing only the even numbers using higher-order functions like map, filter, and reduce.
86. Write a program to represent family relationships (parent, child, sibling, etc.) and query for specific relationships.
87. Write a program that simulates a bank account. Include classes for Account, SavingsAccount, and CheckingAccount, and demonstrate polymorphism through method overriding.
88. Write a function to calculate the factorial of a number using recursion.
89. Write a function to reduce a list to a single value using a binary operator.
90. Write a function that filters a list of strings based on their length.

91. Implement a function that rotates a list by a given number of positions.
92. Implement a program that checks if a string is a palindrome.
93. Create a function that finds all unique elements in a list.
94. Write a function that finds the intersection of two lists.
95. Implement a function that takes a list of integers and returns the sum of the squares of those integers.
96. Create a function that checks whether a list is sorted in ascending order.
97. Write a function that checks if two strings are anagrams of each other.
98. Write a program to count words in a text.
99. Write a function that returns the longest word in a given sentence.
100. Write a program to print the student number and grade given the student number and four test marks. The grade is based on the following rules:

Average mark	Grade
≥ 90	A
≥ 70 and < 90	B
≥ 60 and < 69	C
< 60	Fail

Instructor:

Atoev S.G.

Head of Department:

Rakhimov N.O.

Approved by the protocol of the department meeting No. ____ dated August 26, 2024.