SOFTWARE DEVOLOPMENT

Section A

1. Define the program development lifecycle and list its stages.
2. What is the purpose of the analysis stage in software development?
3. How is a structure chart used in program design?
4. Explain what a state-transition diagram represents.
5. Describe the waterfall model in software development.
6. What are the principles of the iterative model?
7. How does Rapid Application Development (RAD) differ from the waterfall model?
8. Define syntax, logic, and run-time errors with examples.
9. What is a trace table, and how is it used?
10. Describe the purpose of a dry run in testing.
11. What is a test strategy, and why is it important?
12. Define "normal test data" with an example.
13. Explain what a test plan includes.
14. What is alpha testing, and when is it used?
15. Define corrective maintenance.
16. Give an example of a program's boundary test data.
17. Describe white-box testing and when it is applied.
18. What is a module in programming, and why is modular design used?
19. Explain the concept of pseudocode.
20. Describe encapsulation in the context of programming.
21. What is acceptance testing, and who performs it?
22. Give an example of a syntax error.
23. Explain how structure charts can help in debugging.
24. Define extreme test data with an example.
25. Describe adaptive maintenance.
26. What is a finite state machine (FSM)?
27. Define a state-transition table and its purpose.

28. What is stub testing?
29. Why is the design stage critical in the software development lifecycle?
30. Define black-box testing and its main purpose.
31. What is integration testing, and how does it differ from unit testing?
32. Describe a scenario where corrective maintenance is necessary.
33. What is a walkthrough, and how is it used in testing?
34. Explain the role of a flowchart in program design.
35. Define a logical error with an example.
36. What is the purpose of using a trace table in debugging?
37. Describe how RAD incorporates user feedback.
38. What is the purpose of testing during the coding phase?
39. Define perfective maintenance.
40. What is a model in the program development lifecycle?
41. How does a trace table help in a dry run?
42. Describe the role of maintenance in software development.
43. Define a structure chart with an example.
44. What is the benefit of modular design in complex systems?
45. Describe the key principles of the waterfall model.
46. What is the difference between iterative and incremental development?
47. Give an example of using boundary test data in validation.
48. Explain how RAD can reduce development time.
49. Define the role of a test plan in program testing.
50. Describe the main benefit of using state-transition diagrams?

Section B

1. Explain the differences between corrective, perfective, and adaptive maintenance.
2. Why are finite state machines (FSM) essential for program design?
3. Describe the process of constructing a state-transition table.
4. Explain the importance of normal, extreme, and boundary test data in testing.
5. What are the drawbacks of the waterfall model in software development?
6. How does an iterative model improve flexibility in software projects?
7. Describe a scenario where white-box testing would be preferable.
8. Define encapsulation and information hiding with examples.
9. Describe how a program can be tested using a walkthrough.
10. Explain how boundary test data is used to prevent errors.
11. What are the primary differences between alpha and beta testing?
12. Define a state-transition diagram and give an example.
13. How does adaptive maintenance support evolving user needs?
14. Describe the advantages of modular design in debugging.
15. Explain the role of a trace table in identifying logic errors.
16. How does integration testing help in large projects?
17. Describe the purpose and process of black-box testing.
18. How does the RAD model support rapid development?
19. Write pseudocode for a dry run on a simple algorithm.
20. What are some benefits and drawbacks of beta testing?
21. Define iterative development and provide a practical example.
22. How can perfective maintenance improve user experience?
23. Describe the importance of the testing phase in the program lifecycle.
24. Define encapsulation and its significance in programming.
25. Write pseudocode for calculating the area and volume of a sphere.
26. Describe the process of creating a structure chart.
27. What is a finite state machine, and how is it applied in program design?
28. Describe the steps for debugging a logic error.
29. Explain how a test strategy is implemented in software development.
30. Define extreme and abnormal test data with examples.

31. Describe the concept of stub testing with an example.
32. How can a trace table be used to track variable values?
33. Describe the benefits of using white-box testing.
34. Explain the main benefit of modular programming in large projects.
35. How does the iterative model aid in customer feedback?
36. Describe the purpose of acceptance testing in software.
37. Explain the function of a state-transition table.
38. Write pseudocode for a basic validation check on user input.
39. How do state-transition diagrams simplify complex algorithms?
40. Describe a scenario where the waterfall model would be effective.
41. Explain the importance of boundary data in test plans.
42. Write a dry run for calculating the hypotenuse of a triangle.
43. How does encapsulation contribute to data security?
44. Explain the purpose of RAD in complex software projects.
45. Describe a strategy for implementing beta testing.
46. How does adaptive maintenance respond to user feedback?
47. Explain the benefits of prototyping in RAD.
48. Describe how a finite state machine operates with an example.
49. Write pseudocode for a login validation system.
50. How does program documentation support future maintenance?

Section C

1. Describe the purpose and process of creating a finite state machine.
2. Write a complex pseudocode for a state machine handling a 4-digit PIN.
3. Explain the differences between integration and acceptance testing.
4. Write a state-transition diagram for a ticketing system with multiple states.
5. Explain the purpose of a trace table in a recursive function.
6. Write pseudocode to handle adaptive maintenance of a menu-driven system.
7. Describe the drawbacks of RAD for large-scale, high-stakes projects.
8. Explain how encapsulation supports modularity in programming.
9. Design a structure chart for a basic online shopping system.
10. Describe the challenges of maintaining a program with extensive adaptive maintenance.
11. Write a pseudocode algorithm for a login system with multiple user levels.
12. Explain how white-box and black-box testing complement each other.
13. Define modularity and illustrate its application in a multi-function program.
14. Create a state-transition table for a password management system.
15. Write pseudocode for a file management system with multiple options.
16. Describe the challenges of iterative development with complex systems.
17. Write a structure chart for a basic bank account management system.
18. How does black-box testing support integration testing?
19. Write a trace table for a recursive algorithm that calculates factorial.
20. Describe a state-transition diagram for a vending machine operation.
21. Explain the impact of iterative development on testing cycles.
22. Write pseudocode to implement bubble sort on a 1D array.
23. How do finite state machines simplify complex logic in applications?
24. Describe a scenario where perfective maintenance is critical.
25. Write a state-transition diagram for a light switch with multiple modes.
26. Describe the main benefits of black-box testing in high-security software.
27. Write pseudocode to validate a 6-character alphanumeric password.
28. How does the RAD model enable fast feedback from users?
29. Describe how encapsulation helps in multi-developer projects.
30. Write pseudocode for a search function in a sorted list.

31. Describe the importance of test cases for boundary and extreme data.
32. Write pseudocode for a function that handles user sessions.
33. Define and compare unit and integration testing in detail.
34. Create a state-transition diagram for a home security system.
35. Explain how trace tables help in recursive functions.
36. Write a structure chart for a file encryption and decryption program.
37. Describe the process of creating a pseudocode-based test plan.
38. Write pseudocode for sorting data based on multiple conditions.
39. Define stub testing and illustrate with a complex example.
40. Write a program in pseudocode to handle error logging and reporting.
41. Describe the limitations of the iterative model for short projects.
42. Write pseudocode for a queue data structure with enqueue and dequeue.
43. How does black-box testing assist in system integration?
44. Write a state-transition diagram for a three-state thermostat control.
45. Describe the benefits of modularity in error detection and debugging.
46. Explain the significance of state-transition tables in FSM design.
47. Write pseudocode for a multi-step registration and validation system.
48. Describe the importance of acceptance testing in large-scale software.
49. Write a state-transition table for an elevator with multiple floors.
50. Discuss how RAD’ s iterative approach can challenge beta testing.