🔷 General Concepts

1. What is object-oriented programming? How is it better than procedural programming?
2. Explain the features of object-oriented programming.
3. What are the characteristics of OOP?
4. Compare procedural and object-oriented programming with examples.
5. What are the advantages of OOP over POP (Procedure-Oriented Programming)?
6. Compare C++ with C. Why was C++ developed?

🔷 Classes, Objects, and Constructors

7. Define constructor and destructor with example.
8. What are the types of constructors? Explain with syntax and examples.
9. What is a copy constructor? How is it different from assignment operator?
10. Can we have more than one constructor in a class? Why?
11. Create a class `time` with hour, minute, and second and add two objects using constructor.
12. Create a class to find midpoint between two points using `this` pointer.
13. What is the use of dynamic constructor?
14. What is static data member and static member function? Write a program.
15. What is constant object and constant member function? How are they used?

🔷 Function Concepts

16. Define inline function with example. Mention advantages and disadvantages.
17. Explain default arguments. How do they support function overloading?
18. Write syntax and use of default arguments in function.
19. Explain function overloading with program.
20. What is function overriding? How is it different from overloading?
21. How do you access overridden base class members from derived class?
22. Explain the execution order of constructors and destructors in multilevel inheritance.

🔷 Operator Overloading

23. Why is operator overloading used?
24. Which operators cannot be overloaded in C++?
25. Explain syntax of operator overloading for binary and unary operators.
26. Write a program to overload + to add two time/complex objects.
27. Write a program to overload relational operators (<, >, ==) for user-defined class.
28. Overload the index operator [] to input and display array elements.
29. Explain how a user-defined type can be converted into a built-in data type.

🔷 Friend Function & Class

30. What is a friend function? Justify the statement: “Friend has full access”.
31. What is a friend class? Do friends violate encapsulation?
32. Write a program using friend function to access members of two different classes.

🔷 Inheritance & Polymorphism

33. Explain inheritance and its types with examples.
34. Write a program to show constructor invocation order in multiple inheritance.
35. What is ambiguity in inheritance? How is it resolved?
36. What is hybrid inheritance? Explain with program.
37. Define and explain virtual base class with program.
38. Define virtual function. Why is it used?
39. What is pure virtual function and abstract class?
40. Explain runtime polymorphism using virtual function.
41. Differentiate abstract class and concrete class.
42. Write an abstract class with a derived class implementation.
43. What is RTTI? How are `typeid` and `dynamic_cast` used?

🔷 Namespace & Memory

44. Why is namespace required? How is it created and used?
45. Explain the use of `new` and `delete` for dynamic memory allocation.
46. What is dynamic memory management? Why is it needed?

🔷 Templates

47. What is a class template? Write a program to implement a stack using class template.

48. What is a function template? Demonstrate function overloading with template.

49. Explain class template with multiple parameters.

50. What is the need for template in C++?

🔷 Exception Handling

51. How is exception handling better than traditional error handling?

52. Explain multiple exception handling in C++.

53. What is exception specification? How is exception rethrown?

54. Write a program using exception with arguments and specific catch blocks.

🔷 File Handling

55. What is stream class hierarchy in C++?

56. Explain file modes and stream manipulators.

57. Write a program to store and retrieve student/employee records from a file.

58. Write a program to open, read, search, delete and list student records using file.

59. Write a program to copy content from one file to another with case conversion.

60. What is random vs sequential file access? Which one do you prefer and why?

61. Explain file access pointers and manipulators.

62. Write a program for a billing/library management system using file handling.

🔷 Miscellaneous

63. What is encapsulation? How is it enforced in C++?

64. What is data abstraction? How does it differ from encapsulation?

65. What is a polymorphic class?

66. What is the use of static functions in C++?

67. What are the different iostream functions and manipulators?

68. Explain how casting operators like `const_cast`, `reinterpret_cast` are used.

69. What is the purpose of `const_cast`? Provide example.

70. How to convert object of one class into another using casting operator?