SSR Degree and PG College (Autonomous)

BA (CA) YEAR-II SEM-III

OBJECT ORIENTED PROGRAMMING WITH C++

Important Short Answer Questions

UNIT I

1. What is OOP?
2. List the principles of OOP.
3. Define class and object.
4. What are the advantages of OOP?
5. What are tokens in C++?
6. What are keywords?
7. Define identifiers.
8. What are constants?
9. Explain data types.
10. What is implicit type conversion?
11. What is operator precedence?
12. What are control structures?

UNIT II

13. What is a function prototype?
14. Define inline function.
15. What is function overloading?
16. What is call by reference?
17. What is return by reference?
18. Define friend function.
19. What is a virtual function?
20. Define class and object.
21. What are member functions?
22. What are static member functions?
23. What is an array of objects?
24. What is an array within a class?

UNIT III

25. What is a constructor?
26. What is a parameterized constructor?
27. What is a copy constructor?
28. What is a dynamic constructor?
29. What is a destructor?
30. Define operator overloading.
31. List the rules for operator overloading.
32. What is type conversion?
33. What is inheritance?
34. What is single inheritance?
35. What is multilevel inheritance?
36. What is multiple inheritance?

37. What is hierarchical inheritance?
38. What is hybrid inheritance?

UNIT IV

39. What is a pointer?
40. What is a pointer to an object?
41. What is the “this” pointer?
42. What is a pointer to a derived class?
43. What is polymorphism?
44. What is a virtual function?
45. What is file handling?
46. Name the file stream classes in C++.
47. What are file modes?
48. What is a file pointer?
49. What is exception handling?
50. What are try, throw, and catch?

Important Long Answer Questions

UNIT I

1. Explain the Object-Oriented Programming (OOP) paradigm. Discuss its principles, advantages, and applications.
2. Explain the basic concepts of OOP with suitable examples.
3. Discuss the benefits of OOP over procedure-oriented programming.
4. Explain the structure of a C++ program and write a simple C++ program.
5. Explain Tokens in C++. Describe keywords, identifiers, constants, and variables.
6. Explain data types in C++ and discuss type compatibility and implicit type conversions.
7. Explain operators in C++ and discuss operator precedence with examples.
8. Explain control structures (selection, iteration, and branching statements) with examples.

UNIT II

9. Explain functions in C++. Discuss function prototyping, call by reference, and return by reference.
10. Explain inline functions and function overloading with suitable programs.
11. Explain friend functions and virtual functions with examples.
12. Define a class and object. Explain how to define a class and access member functions.
13. Explain arrays within a class and arrays of objects with suitable programs.
14. Explain static data members and static member functions with examples.
15. Write a C++ program to demonstrate class, objects, and member functions.

UNIT III

16. Explain different types of constructors in C++ with examples.
17. Explain parameterized constructor, copy constructor, and dynamic constructor.
18. Explain destructors and their characteristics with examples.
19. Explain operator overloading, rules, and advantages with examples.
20. Explain type conversion in C++ with examples.
21. Explain inheritance and different types with diagrams.
22. Differentiate between single, multilevel, multiple, hierarchical, and hybrid inheritance.
23. Write C++ programs demonstrating different types of inheritance.

UNIT IV

24. Explain pointers in C++. Discuss pointers to objects, this pointer, and pointers to derived classes.
25. Explain virtual functions and runtime polymorphism with examples.
26. Explain file handling in C++. Discuss file stream classes, file modes, opening and closing files.
27. Explain file pointers and input/output operations on files.
28. Explain updating a file with suitable examples.
29. Explain exception handling in C++. Discuss try, throw, and catch blocks with examples.
30. Explain advantages of virtual functions over normal member functions.
31. Write programs demonstrating file handling and exception handling.