JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

B.Tech. - III Semester | SUPPLEMENTARY EXAMINATION - JANUARY 2026

COURSE CODE (CREDITS): 18B11CI311 (3) MAX. MARKS: 75

COURSE NAME: Object-Oriented Systems and Programming (OOSP) MAX. TIME: 2 Hours

COURSE INSTRUCTORS: A. Kumar, A. Sharma, D. Gupta (Coord.), E. Puthooran, H. Singh, N. Singla, R. Sharma.

Note: *1) All questions are compulsory. Marks and COs for each question are indicated. 2)* ***Answer the questions in the given order. 3) Be concise and write neatly.***

Q. No.	Question	CO	Marks
Q. 1	A software development team is migrating an existing payroll system written using structured programming techniques into a modern object-oriented system. As part of this migration, the team wants to understand how object-oriented programming differs from structured programming in terms of problem-solving approach, data security, modularity, and reusability. Explain the key differences between structured programming and object-oriented programming. Discuss how the object-oriented paradigm helps in managing large software systems more effectively. Support your explanation with a small illustrative C++ example.	1	8
Q. 2	An online shopping platform needs to maintain details of the products it sells, such as product ID and price. Each time a new product is added, the system should keep track of how many products currently exist. The price of a product may be updated either as a whole number or as a decimal value depending on discount schemes. Write a C++ program to model this scenario by defining a Product class. Use constructors to initialize product details, a static data member to track the total number of products, and function overloading to update the price using both integer and double values. Also overload the == operator to compare two products based on their prices.	5	8
Q. 3	A software company is designing an employee management system where different types of employees share some common properties but also have their own specific characteristics. The system requires reuse of code while maintaining a clear hierarchy. Explain the concept of inheritance in C++. Using suitable real-life examples in C++, demonstrate multilevel inheritance and hierarchical inheritance. Also explain order in which constructors and destructors are invoked when inheritance is used.	3	8
Q. 4	A graphics application allows users to work with different geometric shapes such as rectangles and circles. Although each shape calculates its area differently, the application should treat all shapes uniformly through a common interface. Design a base class Shape with a virtual function area(). Derive classes Rectangle and Circle that override this function. Write a C++ program to demonstrate runtime polymorphism using base class pointers. Additionally, explain the concepts of dynamic binding, comparing it with early binding.	4	8

Q. 5	A scientific computation system frequently performs similar operations on different data types such as integers, floating-point numbers, and characters. To avoid writing separate functions and classes for each data type, the system designers want a generic solution. Explain the concept of function templates and class templates in C++. Write a C++ program that uses a function template to find the maximum of two values and a class template to store and display a value of any data type. Also explain how template instantiation and type checking are performed by the compiler.	1	8
Q. 6	A data analytics application uses reusable data-processing components where some components are generic and others are specialized. To support flexibility and code reuse, the designers want to combine templates with inheritance. Write a C++ program to demonstrate any one of the following scenarios: • A template class inheriting from another template class • A non-template class inheriting from a template class Explain the advantages of using inheritance in combination with templates.	3	8
Q. 7	A banking application performs financial transactions such as withdrawals and fund transfers. To ensure system reliability, the application must handle exceptional situations such as division by zero, invalid input, or insufficient balance without crashing. Explain the exception-handling mechanism in C++. Write a program that demonstrates throwing and catching exceptions for a divide-by-zero error, uses a user-defined exception class, and re-throws an exception when required. Also explain the concept of stack unwinding.	7	8
Q. 8	A real-time server application dynamically creates and destroys objects during execution. Improper memory handling may lead to memory leaks and program crashes. To solve this problem, modern C++ provides smart pointers. Write a program in C++ to implement unique_ptr, shared_ptr, and weak_ptr with suitable examples. Discuss how these differ from each other and help in automatic memory management while preventing memory leaks.	1	9
Q. 9	A software development team is following standard software engineering practices while designing a library management system. To visualize and document the system, they use Unified Modeling Language (UML). Explain the purpose of UML and draw a class diagram and explain how it helps in analyzing and designing the system.	8	10