JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

B.Tech. - III Semester | T3 EXAMINATION - DECEMBER 2025

COURSE CODE (CREDITS): 18B11CI311 (3) MAX. MARKS: 35

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 company maintains information about its products, specifically the product ID and price. Write a C++ program in which you define a class Product that stores these details. The class should initialize product information using a constructor and include a static data member to track the total number of products created. The program must demonstrate function overloading by providing two versions of a function to update the price: one that accepts an integer value and another that accepts a double value. You should also overload the == operator to compare two Product objects based on their price.	1, 2	5
Q. 2	Define a base class A with a private pointer data member, dynamically allocated in its constructor. Derive a class B from A that has its own private pointer, also dynamically allocated in B's constructor. Implement destructors for both classes to free the memory. In main(), create a dynamic B object but store it in a pointer of type A, then delete the pointer. Explain the problem that occurs and how it can be resolved.	2	5
Q. 3	Write a C++ program that implements a Stack class using templates, including the member functions createStack(), displayStack(), Push(), Pop(), isFull(), and isEmpty(). In the main() function, demonstrate stack operations using both a list of integers and a list of strings.	2	4
Q. 4	Create a class Distance with private data members int feet and float inches, ensuring that the inches value is always less than 12.0. Include a no-argument constructor, a two-argument constructor, and a getdist() function to read values from the keyboard. Define a separate custom exception class DistanceException to store and display both an error message and the invalid inches value. The program should raise this exception whenever an inches value greater than or equal to 12.0 is entered or passed to the constructor, and handle it in main() to report the cause of the error along with the value that triggered it.	2	4

Q. 5	In a manufacturing automation system, a base class Robot represents generic robots. Two specific types exist: • WeldingRobot: Contains an integer member: weldIntensity • PaintingRobot: Contains a double member: paintLevel Write a program that: a) Defines the three classes (Robot, WeldingRobot, PaintingRobot) using proper inheritance and ensures the base class supports RTTI. b) Creates a function monitorRobotStatus(Robot* r) that: • Uses RTTI (typeid) to determine whether the object referred to by r is a WeldingRobot or a PaintingRobot. • Uses a pointer-to-data-member to access the correct member (weldIntensity or paintLevel) and prints its value. c) Demonstrates the working of your function in main() by creating one object of each derived class and passing them to monitorRobotStatus().	1, 2	4
Q. 6	For an Online Food Delivery System, identify all relevant classes, their attributes, and operations. Determine the associations, dependencies, and aggregations among the classes along with appropriate multiplicities. Highlight key constraints, generalizations, and interactions where applicable. Construct a well-structured class diagram that accurately represents the system architecture, showing the responsibilities of each class and the complete relationships among them.	3	4
Q. 7	Provide clear and insightful theoretical explanations for the following (*max. 6-8 sentences*): a) If templates in C++ are meant to reduce code duplication by enabling generic programming, then why do they still lead to code bloat during compilation? Explain the reason behind this contradiction with a brief justification. b) If using const widely improves program reliability and helps prevent unintended modifications, why should mutable be used only in rare cases, and in what practical situations such as caching, logging, or maintaining performance counters, is its use appropriate? c) Differentiate unique_ptr and shared_ptr in C++ in terms of ownership and typical usage. d) Differentiate between the <<include>> and <<extend>> relationships in a UML use-case diagram. Provide one real-world example of each, and draw a simple use-case diagram fragment that illustrates both relationships.	1, 2, 3	2, 3, 2, 2 = 9