U.S.N.										

B.M.S. College of Engineering, Bengaluru-560019

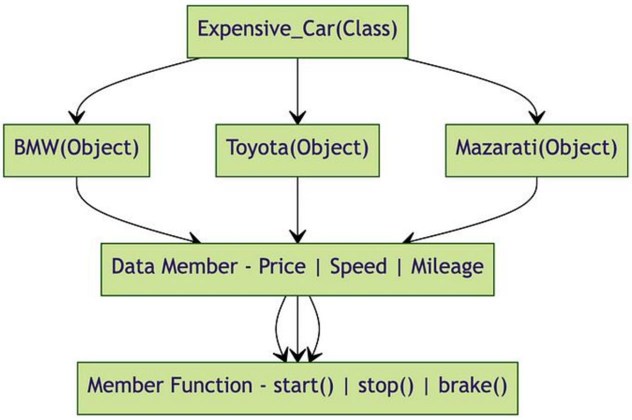
Autonomous Institute Affiliated to VTU

June 2025 Semester End Main Examinations

Programme: B.E. **Semester: V**
Branch: Artificial Intelligence and Machine Learning **Duration: 3 hrs.**
Course Code: 23AM5PCOOP **Max Marks: 100**
Course: Object Oriented Programming

Instructions: 1. Answer any FIVE full questions, choosing one full question from each unit.
2. Missing data, if any, may be suitably assumed.

Important Note: Completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages. Revealing of identification, appeal to evaluator will be treated as malpractice.

		UNIT - I	***CO***	***PO***	**Marks**
1	a)	Compare and contrast instance method and class method.	*CO1*	*PO2*	**06**
	b)	Enumerate four main principles of OOP. Illustrate each of them with an example	*CO1*	*PO1*	**08**
	c)	Provide a working code in python that demonstrates the usage of parameterized constructor and default constructor mechanisms.	*CO1*	*PO1*	**06**
		(OR)			
2	a)	Suppose you have L Programming Languages and M different Operating Systems, in what way Abstract machines helps in modeling L and M. Also, are Abstract Machines necessary?	*CO1*	*PO2*	**06**
	b)	Provide a full working code in python for the following scenario using i. non-parameterized constructor ii. parameterized constructor (default values can be assumed)  Expensive_Car(Class) BMW(Object) Toyota(Object) Mazarati(Object) Data Member - Price | Speed | Mileage Member Function - start() | stop() | brake()	*CO1*	*PO2*	**08**
	c)	Provide full working code to implement addition and subtraction operations without using instance method nor class method.	*CO1*	*PO1*	**06**

			UNIT - II			
	3	a)	Apply argument passing procedures for the following requirements i. Positional parameters type ii. keyword based parameter type.	*CO1*	*PO1*	**10**
		b)	Create a class named `BankAccount` with methods like `deposit`, `withdraw`, and `get_balance`. How would you handle attributes like account balance, account number, and account holder name? Demonstrate how to create multiple bank account instances and perform transactions.	*CO2*	*PO3*	**10**
			OR			
	4	a)	Create a class named `Book` with methods for checking out and returning books. Include attributes like book title, author, and availability status. How would you keep track of multiple books and their availability status using instances of the class?	*CO2*	*PO3*	**10**
		b)	Create a `HealthMonitor` class to store health-related data like heart rate and steps. Use encapsulation to make the health data private and provide methods to update and retrieve the information securely. Discuss the benefits of encapsulation in the context of health data privacy.	*CO2*	*PO3*	**10**
			UNIT - III			
	5	a)	Design an abstract base class named `Recipe` with abstract methods for `prepare`, `cook`, and `serve`. Instantiate concrete subclasses representing different types of recipes, such as `DessertRecipe` and `MainCourseRecipe`.	*CO2*	*PO3*	**10**
		b)	Explain the Diamond problem in the context of multiple inheritance. Why does the Diamond problem occur? Illustrate the solution proposed in python to resolve the same.	*CO2*		**10**
			OR			
	6	a)	Create a hierarchy of vehicle classes (e.g., `Vehicle`, `Car`, `Motorcycle`) with shared and specialized attributes/methods. Implement the `start` method in each class, overriding it when necessary. Utilize the `super` keyword to call the overridden method in the parent class.	*CO2*	*PO3*	**12**
		b)	Create an abstract base class named `Vehicle` with abstract methods like `start`, `stop`, and `fuel`. Implement concrete subclasses for different types of vehicles such as `Car`, `Motorcycle`, and `Truck`. Illustrate how abstraction enables you to define common behavior across diverse vehicle types.	*CO2*	*PO3*	**08**

			UNIT - IV			
	7	a)	Model a music player system with classes for different audio file formats (e.g., MP3, WAV, FLAC). Implement polymorphic methods for playing, pausing, and stopping audio playback. Demonstrate how polymorphism enables the music player to support various file formats seamlessly.	*CO3*	*PO3*	**10**
		b)	Design a program that reads data from a file and processes it. Implement **`try`** and **`except`** blocks to handle potential errors such as file not found or file format issues. Demonstrate how the program handles errors and provides meaningful error messages.	*CO3*	*PO3*	**10**
			OR			
	8	a)	Provide a full working code that demonstrates the concept of polymorphism for the following requirements. i. Two Guitar and Piano classes have a play method. ii. The play_instrument function demonstrates polymorphism by accepting objects of different types as long as they have a play method.	*CO3*	*PO3*	**10**
		b)	Differentiate bug, error and exception. Provide a working code that demonstrates ZeroDivisionError exception mechanism.	*CO3*	*PO3*	**10**
			UNIT - V			
	9	a)	What is garbage collection, and why is it necessary in programming languages? Describe the role of the garbage collector in Python. Explain the difference between automatic and manual garbage collection.	*CO3*	*PO1*	**10**
		b)	In what context multiprocessing is better than multithreading in python?	*CO3*	*PO2*	**05**
		c)	Describe the basic concept of a timer in programming. How would you implement a simple timer in Python without using external libraries?	*CO3*	*PO1*	**05**
			OR			
	10	a)	Differentiae reference counting and garbage collection with examples.	*CO3*	*PO1*	**10**
		b)	How is memory optimization recommended in python?	*CO3*	*PO2*	**05**
		c)	Illustrate the concept of Global Interpreter Lock (GIL). Is GIL necessary?	*CO3*	*PO1*	**05**

******