U.S.N.										

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

Autonomous Institute Affiliated to VTU

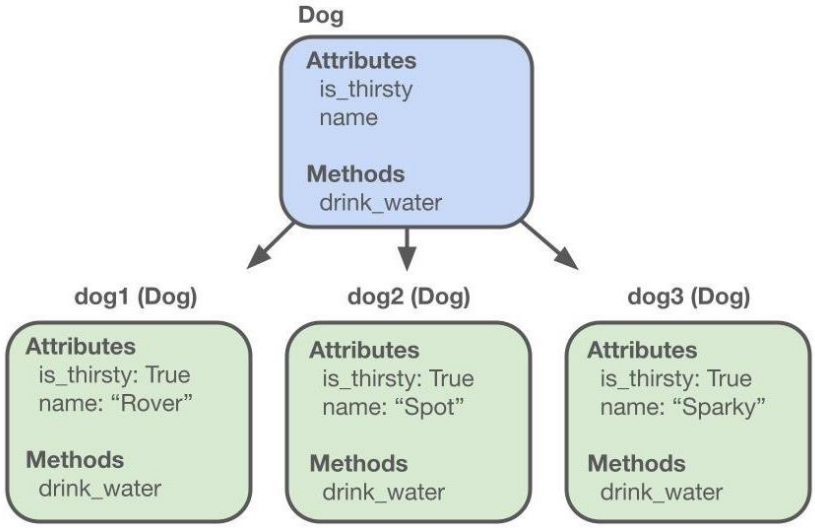
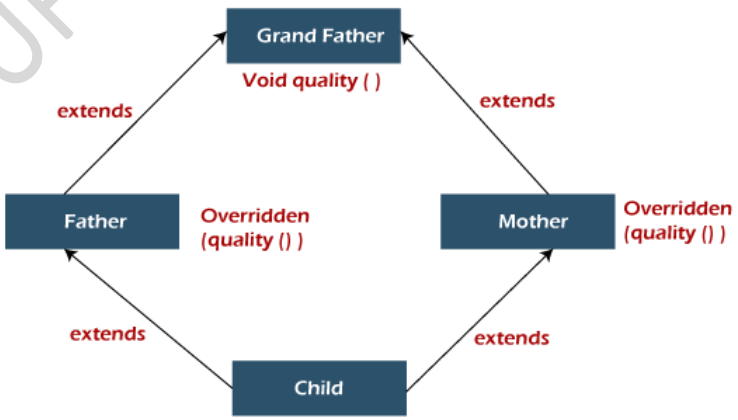
October 2024 Supplementary Examinations

Programme: B.E. **Semester : III**
Branch: Artificial Intelligence and Machine Learning **Duration: 3 hrs.**
Course Code: 23AM3PCOOP **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.

		UNIT - I	CO	PO	Marks
1	a)	Create a Country class with attributes for name, capital, and population. Instantiate two country objects with diverse attributes.	*CO2*	*PO3*	**06**
	b)	Differentiate functional and object-oriented programming languages. Provide relevant code snippets for each.	*CO1*	*PO1*	**08**
	c)	Describe the functioning of the interpreter for a generic abstract machine with a neat diagram.	*CO1*	*PO1*	**06**
		UNIT - II			
2	a)	Design a class named Menu_Item to encapsulate information about items on restaurant menu in python. Include private attributes for the item name, price, and dietary information. Implement methods for setting and getting these attributes. Also, include a method to determine if the item is vegetarian.	*CO3*	*PO3*	**08**
	b)	Differentiate anonymous functions over regular functions. Demonstrate lambda functions to find the largest element in a list.	*CO3*	*PO2*	**05**

	c)	Illustrate different types of methods in python. Complete the following code with relevant type of method.
		Class Rectangle:
		def __init__(self, length, width):
		// fix the code
		
		def calculate_area(self):
		// fix the code
		def calculate_perimeter(self):
		// fix the code
		
		rectangle_instance = Rectangle(5, 8)
		area = rectangle_instance.calculate_area()
		perimeter = rectangle_instance.calculate_perimeter()
			CO2	*PO3*	**07**

			OR			
	3	a)	Develop a python program to mimic the scenario in the following diagram using encapsulation:  	*CO3*	*PO3*	**05**
		b)	Develop an interactive simple calculator using functions which takes user input and performs basic arithmetic operations like addition, subtraction, multiplication, and division based on user's choice.	*CO3*	*PO3*	**10**
		c)	Illustrate the declaration and scope of different access modifiers in python.	*CO2*	*PO2*	**05**
			UNIT - III			
	4	a)	Consider the scenario given below and answer the following questions: i. Identify the type of inheritance in the scenario given below. ii. Assess the efficiency of the execution of the method 'quality ()' in the 'Child' class. Substantiate your response. iii. Illustrate the approach through which Python addresses the issue identified in question (ii).  	*CO2*	*PO1*	**05**
		b)	Design a Python class named Shape that serves as an abstract representation of geometric shapes. Include abstract methods	*CO3*	*PO3*	**10**

			area and perimeter within the class. Derive two classes, Rectangle and Circle, from the Shape class. Implement these classes to calculate and return the area and perimeter for rectangles and circles.			
		c)	How do you access parent class in child class? Provide your answer with an example.	*CO2*	*PO1*	**05**
			OR			
	5	a)	Develop a program to simulate a multimedia player. Create a base class Media with attributes like title and duration. Derive classes Audio and Video from Media. Media Player class inherits from both Audio and Video. Implement methods to play, pause, and stop media.	*CO3*	*PO3*	**10**
		b)	Illustrate 'is-a' relation of OOPs principle with one example.	*CO2*	*PO1*	**05**
		c)	Demonstrate the application of name mangling and the effective use of getters and setters in a Python class.	*CO2*	*PO3*	**05**
			UNIT - IV			
	6	a)	Define polymorphism. Demonstrate the following ways of implementing polymorphism in python with an example for each. i. User defined Method overloading. ii. Overloading in Built in methods.	*CO3*	*PO1*	**08**
		b)	Illustrate the functions of various blocks in exception handling with an example.	*CO2*	*PO2*	**08**
		c)	Illustrate operator overloading with an example program	*CO2*	*PO3*	**04**
			UNIT - V			
	7	a)	Define thread synchronization. Write a python program to create two synchronized threads, where first thread will print odd numbers and second thread will print even numbers between 1 to 20 numbers.	*CO3*	*PO3*	**10**
		b)	Describe life cycle of thread with suitable diagram.	*CO1*	*PO1*	**06**
		c)	Define the following: i. Daemon threads ii. Thread pools iii. Inter-Thread Communication iv. Thread Deadlock	*CO1*	*PO1*	**04**

******