U.S.N.										

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

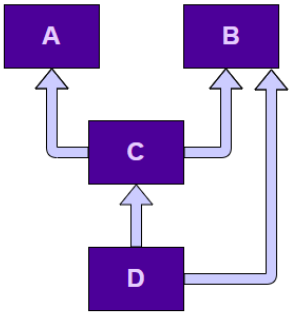
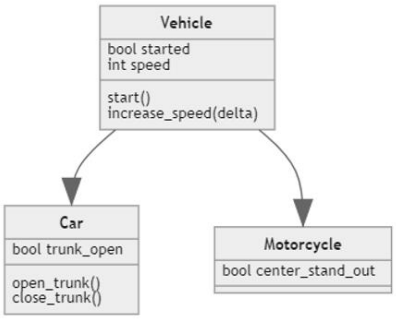
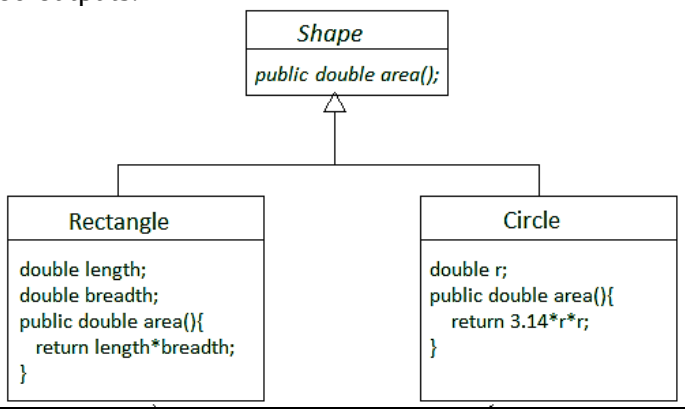
Autonomous Institute Affiliated to VTU

April 2024 Semester End Main Examinations

Programme: B.E. **Semester: III**
Branch: Artificial Intelligence & 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)	Define and illustrate an Abstract Machine.	*CO1*	*PO1*	**06**
	b)	Differentiate Class and Object with a suitable example.	*CO1*	*PO1*	**06**
	c)	Provide a working code that demonstrates usage of parameterized and non-parameterized constructors for the following requirements. i. Employee class maintains name, age and salary related data. ii. Compute annual salary for the same data. iii. Display complete employee information.	*CO1*	*PO3*	**08**
		UNIT - II			
2	a)	Implement a class person, a constructor and a *from_ Birth_Year* method. The constructor takes normal parameters name and age. While, *from_Birth_Year* takes class, name and birthYear. calculates the current age by subtracting it with the current year and returns the class instance.	*CO2*	*PO3*	**08**
	b)	Design a class that handles student age, USN and section_name data. Display the same data with and without creating instances separately.	*CO2*	*PO2*	**07**
	c)	Write an anonymous function that prints even and odd numbers from a list (Assume 10 values in the list).	*CO2*	*PO1*	**05**
		OR			
3	a)	Assume a Banking Class which handles private data like credit card number, account balance of a customer. Provide two different working solutions that handles of an individual customer can only see their respective private information.	*CO2*	*PO2*	**08**
	b)	Write a regular functions wherein multiple integer arguments must be passed (No.of arguments 1, 2, 3).	*CO2*	*PO1*	**06**
	c)	Given the following two data, using anonymous functions get the desired output. *numbers = [1, 2, 3, 4, 5, 6, 7, 8], filter= [3,5,7,8]* ***Expected O/p: [1, 2, 4, 6]***	*CO2*	*PO2*	**06**

			UNIT - III			
	4	a)	Illustrate in what way abstraction is better than encapsulation.	***CO2***	***PO2***	**04**
		b)	Considering the below class design, provide the correct class implementation with a single method each (No implementation required for the method). Also determine the method resolution order. 	***CO2***	***PO3***	**08**
		c)	Convert the following class design diagram into a python code. (Assume valid data for attributes and method output) 	***CO3***	***PO3***	**08**
			OR			
	5	a)	Can multiple inheritance be achieved in python? Provide one possible solution to address the same.	***CO3***	***PO2***	**04**
		b)	Analyze the below scenario and provide necessary object oriented implementation. **Scenario:** RBI plans to implement two types of rate of interest as fixed and floating rates. Assuming banks like SBI and HDFC would need to operate the same to their customers accordingly.	***CO3***	***PO3***	**08**
		c)	Analyze the following class design and answer the following. i) predict the OOP principle that can be employed ii) create appropriate classes and methods as per the design iii) Create instance for the classes required and display the method outputs. 	***CO3***	***PO3***	**08**

			UNIT - IV			
	6	a)	Differentiate bug, error and exception.	***CO3***	***PO1***	**06**

		b)	Fix the following code w.r.t exception handling
			def divide_numbers(a, b):
			  if b == 0:
			  // Fix code
			def main():
			  try:
			  // Fix code
			  except CustomError as e:
			  // Fix code
			main()
			Expected O/p: Error: Division by zero is not allowed. (Line: 5, Method: divide_numbers)
				CO3	***PO2***	**06**

		c)	Illustrate the concept of polymorphism and predict the missing parts of the code w.r.t polymorphism.
			
			from math import pi
			class Shape:
			  def __init__(***//fix code***):
			  self.name = name
			
			  def area(self):
			  pass
			
			  def fact(self):
			  return "I am a two-dimensional shape."
			
			  def __str__(self):
			  return self.name
			
			class Square(Shape):
			  //fix code
			
			class Circle(Shape):
			  //fix code
			
			a = Square(4)
			b = Circle(7)
			print(b)
			print(b.fact())
			//fix code
			print(b.area())
				CO3	***PO2***	**08**

			UNIT - V			
	7	a)	Differentiate Process and Threads	***CO1***	***PO1***	**04**
		b)	Define concurrency and parallelism. Do concurrency and parallelism achieve same results w.r.t. computational efforts on CPU bound tasks or due to I/O bound tasks? Justify	***CO2***	***PO2***	**06**

		c)	Write a python code that mimics both thread and multiprocessing in displaying even and odd numbers with timer of 2ms with an iteration count of 15.	***CO3***	***PO2***	**10**

******