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: Electronics and Communication Engineering **Duration: 3 hrs.**

Course Code: 23EC5PE1OP / 22EC5PE1OP **Max Marks: 100**

Course: Object Oriented Programming using C++

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)	Differentiate Procedural oriented approach and object-oriented approach with an example for each.	*CO 1*	*PO 1*	**8**

	b)	#include <iostream>
		using namespace std;
		
		void update1(int x) {
		    x = 20; }
		
		void update2(int &x) {
		    x = 30; }
		int main() {
		    int a = 10, b = 15;
		    update1(a);
		    update2(b);
		    cout << "a: " << a << ", b: " << b << endl;
		    return 0; }
		Refer the given code to answer the following Questions.
		i. Explain the difference between update1(a) and update2(b).
		ii. What will be output of the program and why?
			CO 1	*PO 1*	**6**

	c)	List the macros in C++. How do they help?	*CO1*	*PO 1*	**6**
		OR			
2	a)	Define a structure **Product** with fields **productName, price, and quantity.** Define an enum Category with values **Electronics, Clothing, and Groceries.** Create an array of **Product structures** and categorize each product using the Category **enum.** Display the product information along with its category.	*CO 3*	*PO 3*	**10**

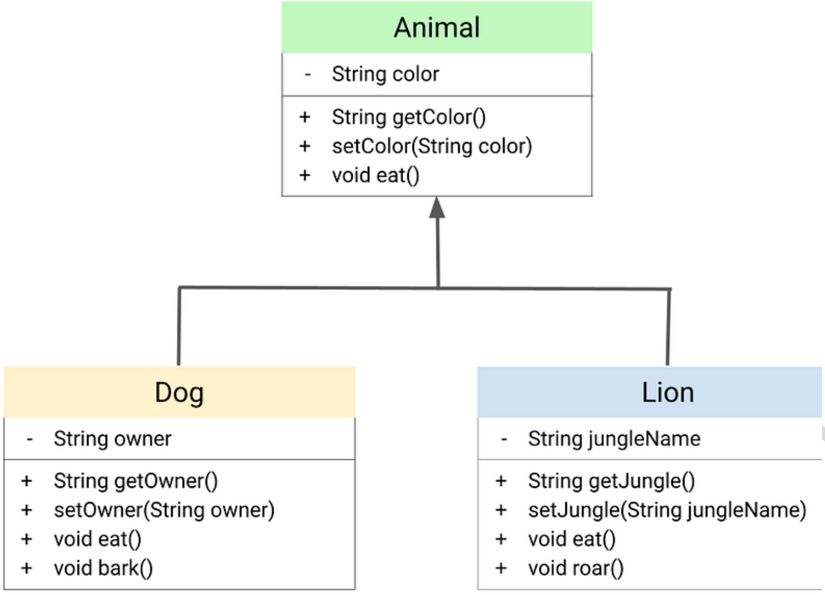
	b)	Consider the following code and write the output:
		int n1,n2; // n1 is stored at memory 0x1002 and n2 at 0x2002
		int *p,*q;
		p=&n1; q=&n2;
			CO 1	*PO 1*	**10**

			int& R=a1;
			cout<< n1 <<n2 << p <<q<<R;
			p++; q++; n1++; n2++; R++;
			cout<< n1 <<n2 << p <<q<<R;
			cout<<*p++; cout<< (*q)++;			

			UNIT - II			

	3	a)	Justify the need of constructors also complete the following code with expected output.
			#include <iostream>
			using namespace std;
			class Rectangle {
			private:
			    int length, width;
			public:
			    Rectangle(int l, int w) {
			        // Missing Line
			        // Missing Line
			    }
			    int calculateArea() {
			        return _____________________;
			    }
			    void display() {
			        cout << "Length: " << length << ", Width: " << width << ", Area: " << calculateArea() << endl;
			    }};
			int main() {
			    Rectangle rect(10, 5);
			    // Missing Line :Display rectangle details
			    return 0;
			}
				CO1	*PO1*	**8**

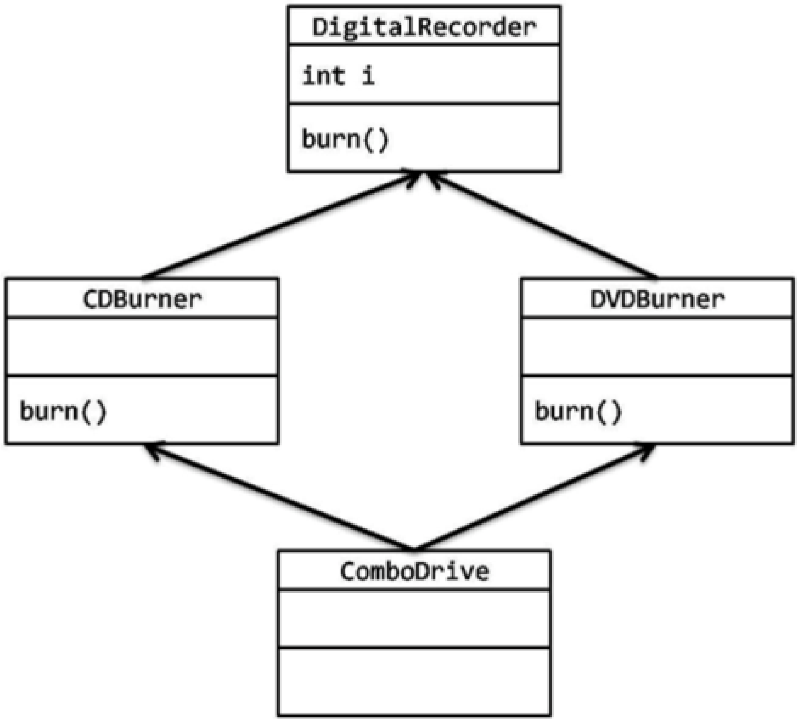
		b)	How is dynamic memory managed in C++ using new and delete? Provide an example.	*CO 1*	*PO 1*	**6**
		c)	Explain the significance of access specifiers in C++. How do they ensure data encapsulation?	*CO 1*	*PO1*	**6**
			OR			
	4	a)	Design a class BankAccount with attributes accountNumber, accountHolderName, and balance. Include contructors to initialize values and member functions to deposit and withdraw money. Ensure that withdrawal fails if the balance is insufficient.	*CO3*	*PO3*	**8**
		b)	Create a class Calculator with overloaded functions add() to handle addition of integers, floating-point numbers, and arrays.	*CO 3*	*PO 3*	**7**
		c)	Summarize the basic principles of object-oriented programming.	*CO 1*	*PO 1*	**5**

			UNIT - III			
	5	a)	Develop codes to mimic the given scenario:  Animal - String color + String getColor() + setColor(String color) + void eat() Dog - String owner + String getOwner() + setOwner(String owner) + void eat() + void bark() Lion - String jungleName + String getJungle() + setJungle(String jungleName) + void eat() + void roar()	*CO3*	*PO3*	**10**

		b)	Predict what will happen if:
			the commented line (this->increment();) is uncommented. Justify your answer.
			#include <iostream>
			class Counter {
			private:
			    int count;
			public:
			    Counter() : count(0) {}
			    void increment() {
			        count++;
			        // this->increment();
			    }
			    int getCount() const {
			        return count;
			    }};
			int main() {
			    Counter c;
			    c.increment();
			    std::cout << "Count: " << c.getCount() << std::endl;
			    return 0;}
				CO1	*PO1*	**4**

		c)	Illustrate the different types of compile time and runtime polymorphism in C++.	*CO 1*	*PO 1*	**6**
			OR			
	6	a)	Complete the C++ class definition for a Time class. Overload the addition operator (+) using a friend function to add two matrices. int main() {	*CO 3*	*PO 3*	**8**

		Time T1,T2(0), T3; // T1,T2 to have sec and min as data members. T2 members should have 0 value.
		T1.settime(); // include codes to read the value from user
		T3 = T1 + T2; // add the sec and minutes of both objects and store in T3
		
		return 0;
		}			

	b)	Identify the type and issue in the given inheritance.  DigitalRecorder int i burn() CDBurner burn() DVDBurner burn() ComboDrive	*CO 3*	*PO 3*	**4**
	c)	Mimic the given scenario in C++. Write the expected output. - **Base class Person**: Contains attributes like name and age. - **Derived class Student** (using **public inheritance**): Access to name and age. - **Derived class Professor** (using **protected inheritance**): Access to name but not age. **Develop classes and functions.**	*CO 3*	*PO3*	**8**
		UNIT - IV			
7	a)	Create a program that defines two custom exceptions (InvalidAgeException and InvalidNameException). Write a function validatePerson that takes an age and name as parameters. If the age is less than 18, throw an InvalidAgeException. If the name is an empty string, throw an InvalidNameException. In the main function, catch both exceptions and display a message for each exception type.	*CO3*	*PO3*	**10**
	b)	Develop a generic function to add 3 numbers (of any data type).	*CO 3*	*PO3*	**10**
		OR			

	8	a)	Write a C++ program that demonstrates the use of templates to create a Swap function that can swap two values of any data type (e.g., int, float, char). Use the template function to swap different types of variables in the main function.	*CO 3*	*PO3*	**10**
		b)	How does C++ handle multiple exceptions using multiple catch blocks? Explain with an example.	*CO 1*	*PO 1*	**10**
			UNIT - V			
	9	a)	Assess the usage of iostream in handling console I/O operations with an example for each function.	*CO 1*	*PO 1*	**10**
		b)	Create a program that: • Reads data from a file line by line and displays each line on the console until the end of the file is reached. • Use a loop to detect when the end of the file is reached and stop reading further. • Demonstrate the use of the eof() function to detect the end of file.	*CO3*	*PO3*	**10**
			OR			
	10	a)	Outline the use of file read and file write functions that are used with files. Use examples.	*CO1*	*PO 1*	**10**
		b)	Discuss the importance of file modes and how they influence the behavior of file operations. Explore any 5 file modes available in C++ and their specific use cases.	*CO 1*	*PO1*	**10**

******