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: III**

Branch: Information Science and Engineering **Duration: 3 hrs.**

Course Code: 23IS3PCOOP/22IS3PCOOP **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)	Explain the features of Object Oriented Programming.			**05**
	b)	Determine the final **return value** and **p value** of the function **update(p, p),** if the value of p is initialized to **5** before the call in the calling function? **int update(int &x, int c) {** **c = c - 1;** **if (c == 0) return 1;** **x = x + 1;** **return f(x, c) + x;** **}**	CO2	PO2	**05**
	c)	Write a C++ program to implement a class called **Circle** that has private member variables for **radius**. Include member functions to **read, display and calculate** the circle's **area and circumference**. Member functions should be defined outside the class. Write mainfunction to create **15** circle objects and calculate area and circumference of those objects.	CO1	PO1	**10**
		OR			
2	a)	Distinguish between procedure-oriented programming and object oriented programming.	CO1	PO1	**05**
	b)	Predict the output of the following **#include<iostream>** **using namespace std;** **int x = 100;** **void Test_scope()** **{** **int x = 2;** **{** **int x = 1;** **cout<<"The value of x in Function is " << x << endl;** **cout<<"the value of ::x in Function is " << ::x << endl;** **}** **cout<<"the value of ::x in outer block is " << ::x << endl;** **}** **int main()** **{** **int x=50;** **cout<<"Value of x is "<<x<<endl;** **fun();** **cout<<"Value of x is "<<::x;** **return 0;** **}**	CO3	PO2	**05**

		c)	Write a C++ program to implement a class called **Date** that has private member variables for **day, month, and year**. Include member functions to **set and get** these variables, as well as to **validate** if the date is valid. Member functions should be defined outside the class. In the main function create **15** Date objects and validate the date of all objects.	*CO1*	*PO1*	**10**
			UNIT - II			
	3	a)	Elucidate inline functions. What are the advantages of inline functions? **void Test_inline( ){ i1 …… i15}** **int main( )** **{** **Test_inline ( );** **}** If Test_inline( ) has 15 instructions. Can it be an inline function? Justify your answer if yes or if not.	*CO1*	*PO1*	**05**
		b)	Write a C++ program to find minimum of Class A and Class B members, and the members are private in their classes. Identify suitable OOP concept and use in the program.	*CO1*	*PO1*	**07**
		c)	Consider the skeleton program given below: **class complex** **{** **int r,i;** **….** **}** **int main()** **{** **Complex c1,c2,c3;** **….** **c3.add(c1,c2);** **add(c1,c2,&c3);** **....** **}** Complete the program with functions to **read, display and add** two complex objects that supports the function call in the main( ).	*CO2*	*PO2*	**8**
			OR			
	4	a)	Elucidate dynamic constructors with an example.			**05**

		b)	Complete the code by writing function definitions for the function calls in the main function.
			class demo {
			Write your code here
			}
			int main()
			{
			demo object;
			object.display();
			object.display("Hello");
			object.display(10);
			object.display(5.5,6.5F);
			}
				CO2	*PO2*	**05**

		c)	You are building a university system to manage student records like name, age and department. The student class should allow the creation of student records in the following ways: 1. A default student record with default information. 2. A student record initialising both the student's name and age. 3. A student record initialising only the name (assumes a default age). 4. A new student record created by copying an existing student's information. Identify appropriate OOP concept to create such records and write a C++ program.	*CO3*	*PO3*	**10**
			UNIT - III			
	5	a)	Describe any three rules for overloading operators? List the operators restricted to be overloaded?			**05**
		b)	Design a **Vehicle Management System** using **multilevel inheritance** in C++. • **Base Class**: Create a class named **Vehicle** that includes common properties of all vehicles such as brand, model, and year. • **Derived Class**: Create a class named **Car** that inherits from Vehicle and adds attributes specific to cars, such as numberOfDoors and fuelType. • **Sub-derived Class**: Create a class named **ElectricCar** that inherits from Car and includes properties unique to electric cars, such as batteryCapacity (in kWh) and rangePerCharge (in miles). The program should include the following: 1. Constructors for all classes to initialize their attributes. 2. A displayDetails function in each class to output the details of the object, ensuring that it displays all relevant information, including inherited and unique attributes. 3. Demonstrate the functionality by creating an object of the ElectricCar class in the main function and displaying its details.	*CO1*	*PO1*	**08**
		c)	Write a C++ program to illustrate overloading pre-increment and post-increment operators in a class called **OpOverload** with two member variables.	*CO1*	*PO1*	**07**
			OR			
	6	a)	Write a C++program to create a class called "**Building**" with attributes for address, number of floors, and total area. Create subclasses "**ResidentialBuilding**" and "**CommercialBuilding**" that add specific attributes like number of apartments for residential and office space for commercial buildings. Define the function to calculate the total rent for each subclass and driver function.	*CO3*	*PO3*	**07**
		b)	Write a C++ program to input an complex number object and display the same by overloading << and >> operators.	*CO3*	*PO3*	**08**

		c)	Consider the program below:
			#include <iostream>
			using namespace std;
			class A {
			public**:** **void display()** {
			cout << "Class A Display" ;
			}
			};
			class B : public A {
			public: **void display()** {
			cout << "Class B Display";
			}
			};
			class C : public A {
			public: **void display()** {
			cout << "Class C Display";
			}
			};
			class D : public B, public C {
			public:
			void display() {
			cout << "Class D Display" << endl;
			}
			};
			int main( )
			{
			D obj;
			obj.display();
			}
			1. Predict the output of the above program with explanation.
			2. Explain what happens if D tries to call A::display() using obj.A::display().
			3. Suggest a solution to resolve ambiguity in accessing the A class members through the diamond inheritance structure.
				CO2	*PO2*	**05**

			UNIT - IV			
	7	a)	Write a pure virtual function, output_Virtual () in class A which acts as base class for Common and Separate. Override the function to print different messages in each sub class. Write a relevant main function to overidden functions.	*CO1*	*PO1*	**07**
		b)	Write a C++ program to illustrate base class pointer to access derived class methods.	*CO1*	*PO1*	**08**
		c)	Explain I/O stream in C++.	*CO1*	*PO1*	**05**
			OR			
	8	a)	Write a C++ program to read the content of the files.			**10**
		b)	Predict the expected outputs of the following instructions: **inf.seekg(20, std::ios::cur);** **inf.seekg(-25, std::ios::cur);** **inf.seekg(24, std::ios::beg);** **inf.seekg(26);** **inf.seekg(-28, std::ios::end);**	*CO2*	*PO2*	**05**
		c)	Can we declare a static function as virtual? Justify your answer.			**05**
			UNIT - V			
	9	a)	Describe the three keywords of exceptions with an example .			**05**
		b)	Write a program to perform the following operations on vector using STL. i. Add elements	*CO3*	*PO3*	**08**

			ii. Access elements iii. Change elements iv. Remove elements			
		c)	Write a C++ function to illustrate templates for finding minimum of elements contained in an array demonstrate for two different data types.	*CO1*	*PO1*	**07**
			OR			
	10	a)	Elucidate the components of STL.			**05**
		b)	Write a C++ program to illustrate class template that demonstrates two generic data types being passed for the class template.	*CO1*	*PO1*	**07**
		c)	Assume an application that has a class **emp** with id, name and salary as member variable. Write read method that accepts values, throw user defined exceptions when salary entered is less than 0.	*CO1*	*PO1*	**08**

****