**1. WAP to print “Hello World” using C++**

**2. What is OOP? List OOP concepts**

* Object-Oriented Programming (OOP) is a programming paradigm that revolves around the concept of "objects," which are instances of classes. It is designed to organize and structure code in a way that models real-world entities and their interactions. OOP is based on four main principles: encapsulation, inheritance, polymorphism, and abstraction. Here are the key concepts of Object-Oriented Programming:

1. Class
2. Object
3. Encapsulation
4. Inheritance
5. Polymorphism
6. Abstraction
7. Message Passing
8. Association
9. Aggregation
10. Composition

These OOP concepts collectively provide a powerful and flexible framework for designing and organizing code in a modular and reusable manner.

**3. What is the difference between OOP and POP?**

Object-Oriented Programming (OOP) and Procedural Programming (POP) are two different programming paradigms that govern how programs are structured, organized, and designed. Here are the key differences between OOP and POP:

**Paradigm:**

**OOP:** Focuses on organizing code into objects, which encapsulate data and behavior.

**POP:** Focuses on procedures or routines that operate on data.

**Data and Behavior:**

**OOP:** Encapsulates data and behavior into objects. Objects have attributes (data members) and methods (functions) that operate on the data.

**POP:** Separates data and procedures. Data is typically structured in records or structures, and functions operate on this data.

**Code Organization:**

**OOP:** Organizes code around classes and objects. Emphasizes modularity, encapsulation, and abstraction.

**POP:** Organizes code around procedures or functions. Emphasizes a step-by-step execution of instructions.

**Encapsulation:**

**OOP:** Achieves encapsulation by bundling data and methods that operate on the data within a class

**POP:** Encapsulation is limited to using procedures to operate on data, but it may not be as strongly enforced as in OOP.

**Inheritance:**

**OOP:** Supports inheritance, allowing one class to inherit properties and behaviors from another class.

**POP:** Typically does not support inheritance in the same way OOP does.

**Polymorphism:**

**OOP:** Supports polymorphism, allowing objects of different types to be treated as objects of a common base type.

**POP:** Polymorphism is achieved through functions that can operate on different types of data, but it may not be as flexible as OOP polymorphism.

**Flexibility and Reusability:**

**OOP:** Emphasizes the reuse of code through the creation of classes and objects, promoting modularity and flexibility.

**POP:** Code reuse is achieved through functions, but it may not be as modular or flexible as OOP.

**Real-World Modeling:**

**OOP:** Often used for modeling real-world entities and their interactions in a natural and intuitive way.

**POP:** Focuses on solving problems through a sequence of procedural steps without necessarily modeling real-world entities as directly.

**Examples:**

**OOP:** Languages like Java, Python, and C++ are commonly associated with OOP.

**POP:** Languages like C, Pascal, and Fortran are often associated with POP.

It's important to note that the choice between OOP and POP depends on the nature of the problem being solved and the preferences of the programmer or development team. Many modern programming languages support both paradigms, allowing developers to choose the approach that best fits the requirements of their projects.

**4. WAP to create simple calculator using class**

**5. Define a class to represent a bank account. Include the following members: Data Member:**

**-Name of the depositor**

**-Account Number**

**-Type of Account**

**-Balance amount in the account**

**3. Member Functions**

**-To assign values**

**-To deposited an amount**

**-To withdraw an amount after checking balance**

**-To display name and balance**

**6. Write a C++ program to implement a class called Circle that has private member variables for radius. Include member functions to calculate the circle's area and circumference.**

**7. Write a C++ program to create a class called Rectangle that has private member variables for length and width. Implement member functions to calculate the rectangle's area and perimeter.**

**8. Write a C++ program to create a class called Person that has private member variables for name, age and country. Implement member functions to set and get the values of these variables.**

**9. Write a program to find the multiplication values and the cubic values using inline function**

**10. Write a program of Addition, Subtraction, Division, Multiplication using constructor.**

**11. Write a C++ program to create a class called Car that has private member variables for company, model, and year. Implement member functions to get and set these variables.**

**12. Write a C++ program to implement a class called Bank Account that has private member variables for account number and balance. Include member functions to deposit and withdraw money from the account.**

**13. Write a C++ program to create a class called Triangle that has private member variables for the lengths of its three sides. Implement member functions to determine if the triangle is equilateral, isosceles, or scalene.**

**14. Write a C++ program to implement a class called Employee that has private member variables for name, employee ID, and salary. Include member functions to calculate and set salary based on employee performance. Using of constructor**

**15. 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.**

**16. Write a C++ program to implement a class called Student that has private member variables for name, class, roll number, and marks. Include member functions to calculate the grade based on the marks and display the student's information. Accept address from each student implement using of aggregation Topics Covered Constructor Destructor Encapsulation Abstraction**

**17. Assume a class cricketer is declared. Declare a derived class batsman from cricketer. Data member of batsman. Total runs, Average runs and best performance. Member functions input data, calculate average runs, Display data. (Single Inheritance)**

**18. Write a C++ Program to find Area of Rectangle using inheritance**

**19. Create a class person having members name and age. Derive a class student having member percentage. Derive another class teacher having member salary. Write necessary member function to initialize, read and write data. Write also Main function (Multiple Inheritance)**

**20. Write a C++ Program display Student Mark sheet using Multiple inheritance**

**21. Assume that the test results of a batch of students are stored in three different classes. Class Students are storing the roll number. Class Test stores the marks obtained in two subjects and class result contains the total marks obtained in the test. The class result can inherit the details of the marks obtained in the test and roll number of students. (Multilevel Inheritance)**

**22. Write a C++ Program to show access to Private Public and Protected using Inheritance**

**23. Write a C++ Program to illustrates the use of Constructors in multilevel inheritance**

**24. Write a program to Mathematic operation like Addition, Subtraction, Multiplication, Division Of two number using different parameters and Function Overloading**

**25. Write a Program of Two 1D Matrix Addition using Operator Overloading**

**26.Write a program to concatenate the two strings using Operator Overloading**

**27.Write a program to calculate the area of circle, rectangle and triangle using Function Overloading Rectangle: Area \* breadth Triangle: ½ \*Area\* breadth Circle: Pi \* Area \*Area Topics Covered Inheritance Polymorphism**

**28.Write a program to swap the two numbers using friend function without using third variable**

**29.Write a program to find the max number from given two numbers using friend function**

**30. Write a program of to swap the two values using template**

**31. Write a program of to sort the array using templates**