 Contact: 8087883669

Java object-oriented programming concepts

Classes: A class in Java is a blueprint or a template for creating objects. It defines the properties (attributes) and behaviors (methods) that an object will have. The attributes of a class represent the state of an object, and the methods represent the behavior of the object. In Java, classes are defined using the class keyword, followed by the class name and the class body.

1. **Objects:** An object in Java is an instance of a class. It represents a specific occurrence of a class and has its own state and behavior. Objects are created using the new keyword, which calls the class constructor to initialize the object. Once an object is created, its properties can be accessed and modified using the dot (.) operator.
2. **Encapsulation**: Encapsulation is the process of hiding the internal details of an object and exposing only the necessary information through methods. In Java, encapsulation is achieved through access modifiers such as public, private, and protected. A public method can be accessed from anywhere, a private method can only be accessed within the class, and a protected method can be accessed within the class and its subclasses.
3. **Inheritance**: Inheritance is the process of creating a new class from an existing class. The new class (subclass) inherits the properties and behaviors of the existing class (superclass) and can also add its own properties and behaviors. In Java, inheritance is implemented using the extends keyword. A subclass can access the properties and methods of its superclass using the dot (.) operator.
4. **Polymorphism**: Polymorphism is the ability of an object to take on many forms. In Java, polymorphism is achieved through method overriding and method overloading.
5. **Method Overriding**: Method overriding is the process of providing a new implementation for an existing method in a subclass. The method in the subclass has the same name, return type, and parameter list as the method in the superclass. When the method is called on an object of the subclass, the overridden method in the subclass is executed instead of the method in the superclass.
6. **Method Overloading**: Method overloading is the process of defining two or more methods with the same name but different parameter lists. The methods can have different numbers or types of parameters. When a method is called with arguments, the Java compiler determines which version of the method to call based on the number and types of the arguments.
7. **Abstraction**: Abstraction is the process of hiding the complexity of an object and exposing only the essential features. In Java, abstraction is achieved through abstract classes and interfaces.
8. **Abstract Classes**: An abstract class in Java is a class that cannot be instantiated. It provides a template for its subclasses and can contain abstract methods (methods without implementation) that must be implemented by its subclasses. An abstract class can also

contain concrete methods (methods with implementation) that can be inherited by its subclasses.

9. **Interfaces**: An interface in Java is a collection of abstract methods that define a contract for a class. A class that implements an interface must provide an implementation for all its methods. An interface can also contain constants and default methods (methods with implementation) starting from Java 8. A class can implement multiple interfaces, but can only extend one superclass.

In summary, object-oriented programming is a powerful paradigm for creating modular and reusable software applications. Java provides a rich set of features for implementing object-oriented programming concepts, including classes, objects, encapsulation, inheritance, polymorphism, abstraction, abstract classes, and interfaces. By mastering these concepts, developers can create robust and maintainable software applications that can adapt to changing requirements and business

Sample program:

Adder Test class :

```
import util.Adder;

public class AdderTest {
   public static void main(String[] args) {
      //new operator in java creates an object from class
      Adder adderObj = new Adder();

      adderObj.takeFirstInputFromUser();

      adderObj.takeSecondInputFromUser();

      adderObj.sumAndPrintResult();

      //below is the private method and cant be
      //accessed from object
      //adderObj.performSum();

   }
}
```

Adder class:

```
package util;
import java.util.Scanner;
public class Adder {
   private int input1 = 0;
   private int input2 = 0;
```

 Contact: 8087883669

```
    private Scanner scanner = new Scanner(System.in);
    public void takeFirstInputFromUser(){
        System.out.println("Enter first input");
        input1 = scanner.nextInt();
    }
    public void takeSecondInputFromUser(){
        System.out.println("Enter second input");
        input2 = scanner.nextInt();
    }
    public void sumAndPrintResult(){
        System.out.println("Sum is : "
                +performSum(input1, input2));
    }
    private int performSum(int i1, int i2){
        int sum = input1 + input2;
        return sum;
    }
}
```

Assignment question:

Write a vehicle class with below specification

- It should take vehicle name (car/bus/truck etc)
- It should take number of wheels from user
- It should take number of passengers which vehicle can accommodate
- It should take vehicle type (electric/petrol/diesel etc)
- It should have method which will print all summary of what information its collected from user