Que1) Create a base class BankAccount with methods like deposit() and withdraw(). Derive a class SavingsAccount that overrides the withdraw() method to impose a limit on the withdrawal amount. Write a program that demonstrates the use of overridden methods and proper access modifiers & return the details.

```
class BankAccount {
    private double balance;

    public BankAccount(double balance) {
        this.balance = balance;
    }

    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: " + amount + ", New balance: " + balance);
    }

    public void withdraw(double amount) {
        balance -= amount;
        System.out.println("Withdrew: " + amount + ", New balance: " + balance);
    }

    public double getBalance() {
        return balance;
    }
}

class SavingsAccount extends BankAccount {
    private double withdrawalLimit;
```

```
    public SavingsAccount(double balance, double withdrawalLimit) {
        super(balance);
        this.withdrawalLimit = withdrawalLimit;
    }

    @Override
    public void withdraw(double amount) {
        if (amount <= withdrawalLimit) {
            super.withdraw(amount);
        } else {
            System.out.println("Withdrawal amount exceeds the limit of: " + withdrawalLimit);
        }
    }
}

public class BankDemo {
    public static void main(String[] args) {
        SavingsAccount sa = new SavingsAccount(1000, 500);
        sa.deposit(200);
        sa.withdraw(300);
        sa.withdraw(600);
    }
}
```

Que2) Create a base class Vehicle with attributes like make and year. Provide a constructor in Vehicle to initialize these attributes. Derive a class Car that has an additional attribute model and write a constructor that initializes make, year, and model. Write a program to create a Car object and display its details.

```
// Base class Vehicle
```

```
class Vehicle {
    private String make;
    private int year;

    // Constructor to initialize make and year
    public Vehicle(String make, int year) {
        this.make = make;
        this.year = year;
    }

    // Getter for make
    public String getMake() {
        return make;
    }

    // Getter for year
    public int getYear() {
        return year;
    }
}

// Derived class Car
class Car extends Vehicle {
    private String model;

    // Constructor to initialize make, year, and model
    public Car(String make, int year, String model) {
        super(make, year);  // Call the base class constructor
        this.model = model;
    }
```

```
    // Getter for model
    public String getModel() {
        return model;
    }

    // Method to display car details
    public void displayDetails() {
        System.out.println("Make: " + getMake());
        System.out.println("Year: " + getYear());
        System.out.println("Model: " + getModel());
    }
}

// Main class to test the program
public class Main {
    public static void main(String[] args) {
        // Creating a Car object
        Car car = new Car("Toyota", 2020, "Camry");
car.displayDetails(); } }

        //
```

Que3) Create a base class Animal with attributes like name, and methods like eat() and sleep(). Create a subclass Dog that inherits from Animal and has an additional method bark(). Write a program to demonstrate the use of inheritance by creating objects of Animal and Dog and calling their methods.

```
// Base class Animal
class Animal {
    protected String name;
```

```
    // Constructor to initialize the name
    public Animal(String name) {
        this.name = name;
    }

    // Method to simulate eating
    public void eat() {
        System.out.println(name + " is eating.");
    }

    // Method to simulate sleeping
    public void sleep() {
        System.out.println(name + " is sleeping.");
    }
}

// Subclass Dog that inherits from Animal
class Dog extends Animal {

    // Constructor to initialize the name using the Animal constructor
    public Dog(String name) {
        super(name);
    }

    // Additional method for Dog to bark
    public void bark() {
        System.out.println(name + " is barking.");
    }
}
```

```
// Main class to demonstrate the use of inheritance
public class Main {
    public static void main(String[] args) {
        // Creating an Animal object
        Animal animal = new Animal("Generic Animal");
        animal.eat();
        animal.sleep();

        // Creating a Dog object
        Dog dog = new Dog("Buddy");
        dog.eat();     // Inherited from Animal
        dog.sleep();   // Inherited from Animal
        dog.bark();    // Dog's specific method
    }
}
```

Que 4) Build a class Student which contains details about the Student and compile and run its instance.

```
// Student class
class Student {
    // Attributes of the Student
    private String name;
    private int age;
    private String studentId;

    // Constructor to initialize the attributes
    public Student(String name, int age, String studentId) {
        this.name = name;
        this.age = age;
        this.studentId = studentId;
```

```
    }

    // Method to display student details
    public void displayDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Student ID: " + studentId);
    }
}

// Main class to run the program
public class Main {
    public static void main(String[] args) {
        // Creating an instance of Student
        Student student = new Student("Alice", 20, "S12345");

        // Displaying the student's details
        student.displayDetails();
    }
}
```

Que5) Write a Java program to create a base class Vehicle with methods startEngine() and stopEngine(). Create two subclasses Car and Motorcycle. Override the startEngine() and stopEngine() methods in each subclass to start and stop the engines differently.

```
// Base class Vehicle
class Vehicle {

    // Method to start engine
    public void startEngine() {
        System.out.println("Vehicle engine is starting...");
    }

    // Method to stop engine
    public void stopEngine() {
        System.out.println("Vehicle engine is stopping...");
    }
}
```

```
// Subclass Car that overrides the startEngine and stopEngine methods
class Car extends Vehicle {

    @Override
    public void startEngine() {
        System.out.println("Car engine is starting with a key ignition...");
    }

    @Override
    public void stopEngine() {
        System.out.println("Car engine is stopping by turning off the key...");
    }
}

// Subclass Motorcycle that overrides the startEngine and stopEngine methods
class Motorcycle extends Vehicle {

    @Override
    public void startEngine() {
        System.out.println("Motorcycle engine is starting with a button press...");
    }

    @Override
    public void stopEngine() {
        System.out.println("Motorcycle engine is stopping by turning off the button...");
    }
}

// Main class to demonstrate method overriding
public class Main {
    public static void main(String[] args) {
        // Creating a Car object
        Vehicle car = new Car();
        car.startEngine();
        car.stopEngine();

        // Creating a Motorcycle object
        Vehicle motorcycle = new Motorcycle();
        motorcycle.startEngine();
        motorcycle.stopEngine();
    }
}
```