1. Exercise: You are given a partially completed code in the editor. Modify the code so that the code prints the following text:

**Hello I am a motorcycle, I am a cycle with an engine.**

**My ancestor is a cycle who is a vehicle with pedals.**

class Cycle{

    String define\_me(){

        return "a vehicle with pedals.";

    }

}

class Bike extends Cycle{

    String define\_me(){

        return "a cycle with an engine.";

    }

    Bike(){

        System.out.println("Hello I am a Bike I am "+ define\_me());

        String temp=define\_me();

        System.out.println("My ancestor is a cycle who is "+ temp );

    }

}

class InheritenceExample{

    public static void main(String []args){

        Bike M=new Bike();

    }

}

1. Exercise:

* Consider the below code and you must add a ‘**bark** method to the Dog class, then modify the main method accordingly so that the code prints the following lines:

I am walking

I am eating

I am barking

class Animal{

void walk(){

System.out.println("I am walking");

}

}

class Dog extends Animal{

void eat(){

System.out.println("I am eating");

}

}

public class InheritenceExample{

   public static void main(String[] args){

      Dog dog = new Dog();

      dog.walk();

      dog.eat();

   }

}

1. Exercise: Create class named as ‘A’ and create a sub class ‘B’. Which is extends from class ‘A’. Add some methods and properties to A and to B. Then create a third test class (that has a public static void main method) to experiment and verify that inheritance is taking place.
2. Exercise: Create an Animal superclass and a Dog subclass. Add a public property - furColor - and method - makeNoise() - to Animal. Then test that you can create an instance of Dog and access the inherited members. Now change the methods and variables created to private. Test if you are still seeing the inherited members in the subclass. Now try adding a public getter/setter for the private property in Animal. Test whether you can access the inherited getter/setter on the Dog instance.
3. (overriding) Override the inherited makeNoise() method in Dog. For the overridden version, take in a String argument for the message you want to display in the method when invoked. Use a covariant return type in the method signature.
4. Do any hands-on exercises and quizzes in Chapter 7 ("Inheritance and Polymorphism") in Head First Java: (https://learning.oreilly.com/library/view/head-first-java/0596009208/ch07.html)
5. Start working through the “Aspire Journeys: Java Novice to Javanista” videos on Percipio (https://hclcontent.percipio.com/track/afc2d17e-927f-4d0a-9f7a-0fe98ef9ae17). These have many quizzes sprinkled throughout.