Encapsulation:

CODE:

public class Main {

public static void main(String[] args) {

Person person = new Person();

person.setName("Alice");

person.setAge(25);

System.out.println("Name: " + person.getName());

System.out.println("Age: " + person.getAge());

}

}

class Person {

private String name;

private int age;

public void setName(String name) {

this.name = name;

}

public String getName() {

return name;

}

public void setAge(int age) {

this.age = age;

}

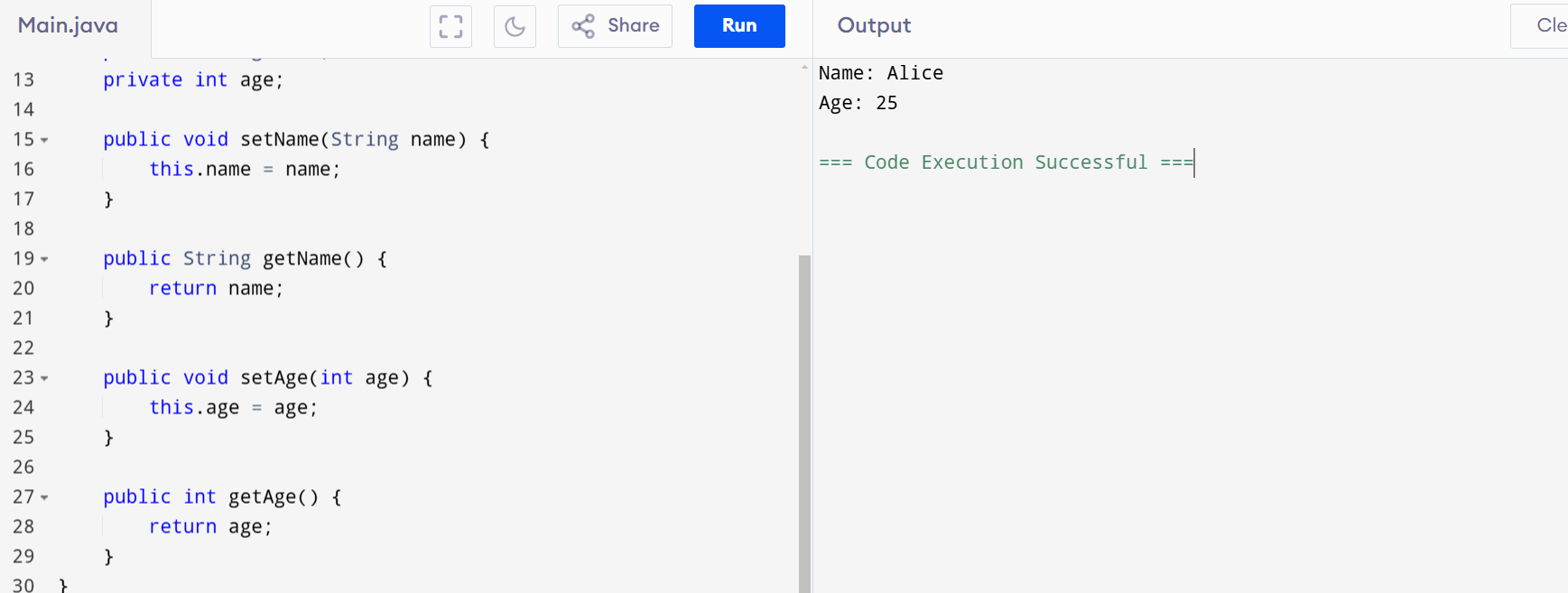
public int getAge() {

return age;

}

}

OUTPUT:



Inheritance:

CODE:

public class Main {

public static void main(String[] args) {

Dog dog = new Dog();

dog.sound();

dog.sleep();

}

}

class Animal {

public void sound() {

System.out.println("Animal makes a sound");

}

public void sleep() {

System.out.println("Animal is sleeping");

}

}

class Dog extends Animal {

public void sound() {

System.out.println("Dog barks");

}

}

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.

Polymorphism:

CODE;

public class Main {

public static void main(String[] args) {

Animal animal = new Animal();

Animal dog = new Dog();

animal.sound();

dog.sound();

}

}

class Animal {

public void sound() {

System.out.println("Animal makes a sound");

}

}

class Dog extends Animal {

public void sound() {

System.out.println("Dog barks");

}

}

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.

Abstraction:

CODE:

public class Main {

public static void main(String[] args) {

Car car = new Car();

car.start();

car.stop();

}

}

abstract class Vehicle {

public abstract void start();

public abstract void stop();

}

class Car extends Vehicle {

public void start() {

System.out.println("Car started");

}

public void stop() {

System.out.println("Car stopped");

}

}

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.