Problem Set

1. Create an abstract class Animal with an abstract method sound(). Create two subclasses Dog and Cat that extend Animal and provide implementations for the sound() method. In the main() method, create objects of Dog and Cat and call the sound() method.
2. Define an abstract class Shape with an abstract method area() and a concrete method description() that prints "This is a shape". Create two subclasses Circle and Rectangle that provide specific implementations of area(). Call both description() and area() methods from objects of Circle and Rectangle.
3. Create an abstract class Vehicle with a constructor that takes a String parameter type and initializes it. Create two subclasses Car and Bike that inherit from Vehicle and provide their own method getFuelType(). Demonstrate the use of the constructor in the abstract class.
4. Create an abstract class Employee with an abstract method getSalary() that returns a double. Create two subclasses FullTimeEmployee and PartTimeEmployee, each providing its own implementation of getSalary(). In the main() method, create instances of both subclasses and display their salaries.
5. Create an abstract class Instrument with an abstract method play(). Create two subclasses Piano and Guitar, both implementing play(). In the main() method, create an array of Instrument references and assign objects of Piano and Guitar to it, then call play() on each.
6. Define an abstract class Appliance with an abstract method turnOn() and a concrete method powerInfo() that prints "This appliance uses electricity." Create two subclasses WashingMachine and Microwave that implement turnOn(). Call both turnOn() and powerInfo() in the main() method.
7. Create an abstract class Person with an abstract method getDetails() and a concrete method sayHello() that prints "Hello!". Create two subclasses Teacher and Student, both providing implementations of getDetails() and overriding sayHello() to print customized greetings. Demonstrate both method overriding and abstract class implementation.
8. Create an abstract class Shape with an abstract method area(). Create an interface Drawable with a method draw(). Create a class Circle that extends Shape and implements Drawable, providing specific implementations for both area() and draw(). Demonstrate the use of both abstract class and interface in one class.
9. Create an abstract class Machine with a constructor that takes an int parameter for id. Create two subclasses Computer and Printer that use the constructor of Machine via super(). In both subclasses, override a method start(). Demonstrate constructor chaining in the main() method.

10. Create an abstract class Device with a final method deviceInfo() that prints "This is a device." Add an abstract method functionality() in the class. Create two subclasses Phone and Tablet that override the functionality() method. In the main() method, create instances of Phone and Tablet and call both deviceInfo() and functionality().
11. Create an abstract class LivingBeing with an abstract method breathe(). Create a subclass Animal that extends LivingBeing and adds another abstract method move(). Create a subclass Bird that implements both breathe() and move() methods. Demonstrate the use of multiple abstract classes in an inheritance chain.
12. Create an abstract class BankAccount with an abstract method calculateInterest(). Create two subclasses SavingsAccount and CurrentAccount, each implementing calculateInterest() differently. In the main() method, use polymorphism to call calculateInterest() on both types of accounts using a BankAccount reference.