Abstraction and Interface

1. Create an abstract class **Shape** with abstract methods area() and perimeter().
   Derive classes **Circle**, **Rectangle**, and **Triangle** to implement these methods and calculate the respective values.
2. Design an abstract class **Vehicle** with properties like speed, color, and an abstract method startEngine().
   Create subclasses **Car**, **Bike**, and **Truck** that provide their own implementation of startEngine().
3. Define an interface **PaymentGateway** with methods pay(), refund(), and generateReceipt().
   Implement this interface in classes **CreditCardPayment**, **BkashPayment**, and **PayPalPayment**.
4. Create two interfaces: **Printable** (method print()) and **Scannable** (method scan()).
   Implement both interfaces in a class **MultiFunctionPrinter** that performs both tasks.
5. Create an abstract class **BankAccount** that has a constructor to initialize accountNumber and balance.
   Include an abstract method calculateInterest().
   Extend it with **SavingsAccount** and **CurrentAccount** that implement the method differently.
6. Define an interface **Logger** with an abstract method logMessage(), a **default** method logInfo(), and a **static** method logError().
   Implement it in class **FileLogger** to show their use.
7. Create an abstract class **Employee** with data members name, id, and an abstract method calculateSalary().
   Subclasses **FullTimeEmployee** and **PartTimeEmployee** should implement their own salary calculation.
8. Design an abstract class **Appliance** with abstract methods turnOn() and turnOff().
   Create subclasses **Fan**, **Television**, and **Refrigerator** that provide specific implementations.
9. Define two interfaces: **Readable** and **Writable**.
   Then create another interface **FileOperations** that extends both and adds a method openFile().
   Implement **FileOperations** in class **TextFile**.

10. Create an abstract class **Animal** with abstract method sound().
    Create an interface **Pet** with method play().
    Implement both in classes **Dog** and **Cat**.
11. Model a **transport booking system** using abstraction:

- Abstract class **Transport** with abstract methods bookTicket() and calculateFare().
- Derived classes: **Bus**, **Train**, **Airplane**, each implementing them differently.

12. Interface for Game Design

Create interfaces **Playable**, **Saveable**, and **Controllable**.
Implement them in classes **CarGame**, **ShootingGame**, and **AdventureGame** showing multiple behaviors.