Zadanie:
Stwórz hierarchię klas reprezentującą różne typy pojazdów w wypożyczalni. Zaimplementuj klasę bazową Pojazd oraz klasy pochodne **Samochod** i **Motocykl**. Każdy pojazd powinien mieć metodę **obliczKosztWypozyczenia()**, która będzie zwracała koszt wypożyczenia pojazdu na jeden dzień. Koszt ten powinien być obliczany w następujący sposób:

Dla samochodów: podstawowa stawka dzienna + dodatkowa opłata za każde miejsce siedzące
Dla motocykli: podstawowa stawka dzienna + dodatkowa opłata za każdy kilowat mocy

Zaimplementuj również metodę **wyswietlInformacje()**, która będzie wyświetlała podstawowe informacje o pojeździe.
A teraz przykładowy kod, który jest analogiczny do rozwiązania, ale nie stanowi bezpośredniego rozwiązania zadania:

```
abstract class Owoc {  2 usages  2 inheritors
    protected String nazwa;  2 usages
    protected double cenaPodstawowa;  3 usages
    public Owoc(String nazwa, double cenaPodstawowa) {  2 usages
        this.nazwa = nazwa;
        this.cenaPodstawowa = cenaPodstawowa;
    }
    public abstract double obliczCeneSprzedazy();  1 usage  2 implementations
    public void wyswietlInformacje() {  2 usages  2 overrides
        System.out.println("Nazwa: " + nazwa);
        System.out.println("Cena sprzedaży: " + obliczCeneSprzedazy() + " zł");
    } }
class Jablko extends Owoc {  no usages
    private String odmiana;  3 usages
    public Jablko(String nazwa, double cenaPodstawowa, String odmiana) {  no usages
        super(nazwa, cenaPodstawowa);
        this.odmiana = odmiana;
    }
    @Override  1 usage
    public double obliczCeneSprzedazy() {
        return cenaPodstawowa + (odmiana.equals("Golden") ? 0.5 : 0.3);
    }
    @Override  2 usages
    public void wyswietlInformacje() {
        super.wyswietlInformacje();
        System.out.println("Odmiana: " + odmiana);
    } }
class Banan extends Owoc {  no usages
    private double waga;  3 usages
    public Banan(String nazwa, double cenaPodstawowa, double waga) {  no usages
        super(nazwa, cenaPodstawowa);
        this.waga = waga;
    }
    @Override  1 usage
    public double obliczCeneSprzedazy() {
        return cenaPodstawowa + (waga * 0.2);
    }
    @Override  2 usages
    public void wyswietlInformacje() {
        super.wyswietlInformacje();
        System.out.println("Waga: " + waga + " kg");
    } }
```

Ten przykładowy kod przedstawia hierarchię klas dla owoców w sklepie. Jest on analogiczny do zadania z pojazdami, ale używa innego kontekstu. Klasa bazowa Owoc odpowiada klasie Pojazd w oryginalnym zadaniu, a klasy Jablko i Banan są odpowiednikami klas Samochod i Motocykl.
Główne podobieństwa:
Abstrakcyjna klasa bazowa z metodami obliczCeneSprzedazy() (analogiczna do obliczKosztWypozyczenia()) i wyswietlInformacje(). Klasy pochodne implementujące własne wersje tych metod. Różne sposoby obliczania ceny dla różnych typów obiektów.

Różnice od oryginalnego zadania:

- Inny kontekst (owoce zamiast pojazdów).
- Różne specyficzne atrybuty dla klas pochodnych.
- Nieco inne zasady obliczania ceny.

Ten przykład powinien pomóc w zrozumieniu koncepcji dziedziczenia i polimorfizmu,