Real-World Inheritance Problems in Python

1. Vehicle Hierarchy

You are tasked with creating a class structure for different types of vehicles. Create a base class `Vehicle` with attributes like `make`, `model`, and `year`, and methods like `start()` and `stop()`. Then, create subclasses such as `Car`, `Truck`, and `Motorcycle` that inherit from `Vehicle` and add specific features such as `number_of_doors` for `Car`, `cargo_capacity` for `Truck`, and `type_of_handlebars` for `Motorcycle`.

Sample Input & Output

Input:

```
car = Car("Toyota", "Camry", 2022, 4)
car.start()
car.number_of_doors
```

Output:

```
Car started.
4
```

---

2. Employee Management System

Design an employee management system where you have a base class `Employee` with attributes like `name`, `id`, and `salary`. Create subclasses `Manager`, `Engineer`, and `Intern` that inherit from `Employee`. Add specific attributes like `team_size` for `Manager`, `skills` for `Engineer`, and `university` for `Intern`.

Sample Input & Output

Input:

```
manager = Manager("Alice", 101, 90000, 10)
manager.team_size
```

Output:

```
10
```

---

3. E-commerce Platform

Build an e-commerce platform where you have a base class `Product` with attributes like `name`, `price`, and `description`. Create subclasses `Clothing`, `Electronics`, and `Furniture` that inherit from `Product` and add specific attributes such as `size` and `material` for `Clothing`, `warranty_period` for `Electronics`, and `dimensions` for `Furniture`.

Sample Input & Output

Input:

```
clothing = Clothing("T-Shirt", 20, "Cotton T-shirt", "L", "Cotton")
clothing.material
```

Output:

```
Cotton
```

---

4. School Management System

Develop a school management system with a base class `Person` that contains attributes like `name`, `age`, and `address`. Create subclasses `Student` and `Teacher` that inherit from `Person`. The `Student` class should have additional attributes like `grade` and `subjects`, while the `Teacher` class should have `specialization` and `salary`.

Sample Input & Output

Input:

```
student = Student("John", 16, "123 Main St", "10th", ["Math", "Science"])
student.subjects
```

Output:

```
['Math', 'Science']
```

---

5. Banking System

Create a banking system with a base class `Account` that has attributes like `account_number`, `holder_name`, and `balance`. Develop subclasses `SavingsAccount` and `CurrentAccount` that inherit from `Account` and add features such as `interest_rate` for `SavingsAccount` and `overdraft_limit` for `CurrentAccount`.

Sample Input & Output

Input:

```
savings = SavingsAccount("123456", "Alice", 1000, 0.05)
savings.interest_rate
```

Output:

```
0.05
```

---

6. Zoo Management System

Create a base class `Animal` with attributes like `name`, `species`, and `diet`. Implement subclasses like `Mammal`, `Bird`, and `Reptile` that inherit from `Animal` and add specific attributes such as `gestation_period` for `Mammal`, `wing_span` for `Bird`, and `cold_blooded` for `Reptile`.

Sample Input & Output

Input:

```
bird = Bird("Parrot", "Psittacine", "Herbivore", 0.5)
bird.wing_span
```

Output:

```
0.5
```

---

7. Online Learning Platform

Build a class structure for an online learning platform. Create a base class `Course` with attributes like `title`, `description`, and `duration`. Subclasses such as `VideoCourse`, `ArticleCourse`, and `QuizCourse` should inherit from `Course` and have specific attributes like `video_length`, `number_of_pages`, and `number_of_questions`.

Sample Input & Output

Input:

```
video_course = VideoCourse("Python Basics", "Learn Python", 5, 120)
video_course.video_length
```

Output:

```
120
```

---

8. Library Management System

Create a base class `Item` with attributes such as `title`, `author`, and `publication_year`. Subclasses such as `Book`, `Magazine`, and `DVD` should inherit from `Item` and add specific features like `genre` for `Book`, `issue_number` for `Magazine`, and `duration` for `DVD`.

Sample Input & Output

Input:

```
book = Book("1984", "George Orwell", 1949, "Dystopian")
book.genre
```

Output:

```
Dystopian
```

---

9. Gaming System

Develop a gaming system where you have a base class `Character` with attributes like `name`, `health`, and `level`. Subclasses like `Warrior`, `Mage`, and `Archer` should inherit from `Character` and have unique abilities like `weapon_type` for `Warrior`, `magic_type` for `Mage`, and `range` for `Archer`.

Sample Input & Output

Input:

```
warrior = Warrior("Thor", 100, 5, "Hammer")
warrior.weapon_type
```

Output:

```
Hammer
```

---

10. Transport Booking System

Design a transport booking system with a base class `Transport` that has attributes like `source`, `destination`, and `cost`. Subclasses `Flight`, `Train`, and `Bus` should inherit from `Transport` and add specific attributes such as `airline` for `Flight`, `coach_type` for `Train`, and `seat_capacity` for `Bus`.

Sample Input & Output

Input:

```
flight = Flight("New York", "London", 500, "British Airways")
flight.airline
```

Output:

```
British Airways
```
