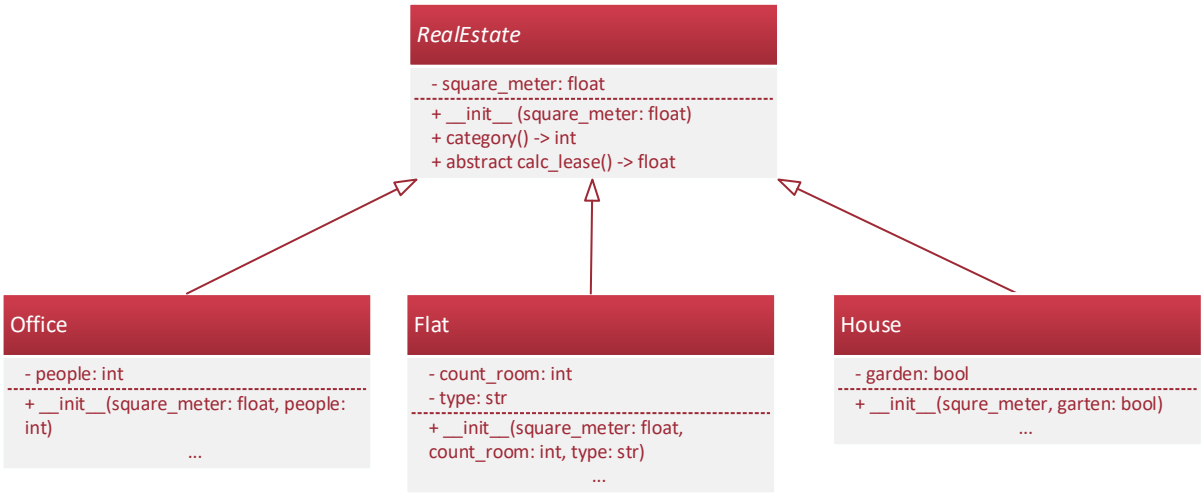
Übungsbeispiel 8: Real Estate

Orientieren Sie sich an folgendem Diagramm und Beschreibung zur Abbildung von Immobilien und Berechnung der Miete.



Die abstrakte Basisklasse ***RealEstate*** nimmt die Quadratmeter der Immobilie auf und stellt die Methode `calc_lease()-> float` zur Berechnung der Miete zur Verfügung. Stellen Sie sicher, dass diese Methode in allen Ableitungen implementiert werden muss. square_meter soll private und als get/set Property umgesetzt werden. category als get Property, das die Quadratmeter durch 10 dividiert (als int) zurückliefert.

Implementieren Sie in der Klasse RealEstate, sowie in allen abgeleiteten Klassen die repr Methoden.

Implementieren Sie die Klasse ***Office*** und leiten Sie von RealEstate ab. Diese Klasse übernimmt zusätzlich zu den Parametern der super-Klasse, die Anzahl der erlaubten Personen im Büro auf. Stellen Sie folgende Mietenberechnung, abhängig von den Personen, zur Verfügung:

	Miete
Personen < 50	Quadratmeter * 8
Personen >= 50 und Personen < 100	Quadratmeter * 8.2 + 90
Personen >= 100	Quadratmeter * 8.5 + Personen

Implementieren Sie die Klasse ***Flat*** und leiten Sie von RealEstate ab. Im init wird zusätzlich die Anzahl der Räume und ein Wohnungstyp (High, Standard, Low als String) übernommen und in Attributen gespeichert. Die Mietsberechnung ist abhängig vom Wohnungstyp:

type	**Berechnung**
Low	Quadratmeter * 7
Standard	Quadratmeter * 7.5 + Anzahl-Räume * 10
High	Quadratmeter * 8 + Anzahl-Räume * 12
Andere Werte	-1

Implementieren Sie die Klasse ***House*** und leiten Sie von RealEstate ab. Im init wird noch zusätzlich übergeben, ob die Immobilie einen Garten hat.

Berücksichtigen Sie folgende Punkte bei der Berechnung der Miete:

- Miete (wenn Garten vorhanden) = Quadratmeter * 10 + 200
- Miete (ohne Garten) = Quadratmeter * 15

UND

- Liegt die Miete unter 1.000 Euro, so wird mindestens 1.000 Euro verlangt.

Erstellen Sie eine Klasse **Accounting**. Verwalten Sie in einer privaten Liste die gekauften RealEstates (Name real_estates) und instanziieren Sie diese leere Liste im init.

Erstellen Sie eine Methode `add(re: RealEstate)`, welche weitere RealEstates in die Liste aufnimmt.

Erstellen Sie eine Methode `print_all()`, welche alle RealEstates samt deren Quadratmeter und Miete ausgibt.

Implementieren Sie eine Methode `get_overall_lease() -> float`, welche die Gesamtmiete über alle gekauften RealEstates hinweg berechnet.

Implementieren Sie eine Methode `get_average_lease() -> RealEstate`, welche den Durchschnitt aller Mieten berechnet. Der Durchschnitt berechnet sich aus der Summe aller Mieten, dividiert durch die Anzahl der Immobilien.

Implementieren Sie eine Methode `get_real_estate_in_category() -> Dict[int, int]`, welche über alle gekauften RealEstates hinweg die Anzahl der RealEstate je category (verwenden Sie hierzu das category Property) ermittelt. Im ersten Schlüsselwert des Dict wird das Ergebnis von category gespeichert, im zweiten Wert wieder wird die Anzahl der Immobilien dieser Kategorie erfasst.

Ausprobieren
Testen Sie Ihre Implementationen in einer main Funktion. Zur Überprüfung können Sie auch _repr_ Methoden umsetzten bzw. die Unit Tests verwenden.