1. Napisz klasę *Cat*, która w konstruktorze przyjmować będzie imię kota. Klasa powinna posiadać metodę *makeSound*, która wypisywać będzie imię kota oraz wydawany przez niego dźwięk.
2. Utwórz tablicę kotów, dodaj do niej parę utworzonych obiektów i dla wszystkich wywołaj metodę *makeSound*.
3. Do klasy *Cat* dodaj metodę *eatMouse*, która będzie zliczała zjedzone przez kota myszy i wypisywała komunikat: „Zjadłem X myszy”.
4. Napisz klasę *Dog*, która w konstruktorze przyjmować będzie imię psa. Klasa powinna posiadać metodę *makeSound*, która wypisywać będzie imię psa oraz wydawany przez niego dźwięk.
5. Utwórz tablicę zwierząt, dodaj do niej parę utworzonych obiektów typu *Cat* oraz *Dog* i dla wszystkich wywołaj metodę *makeSound*.
6. Napisz interfejs *Movable*, który będzie zawierał metodę *move*. Napisz klasę *Car* implementującą interfejs *Movable* – w metodzie *move* ma wypisać komunikat „jadę”.
7. Zmodyfikuj klasę *Cat* tak, żeby implementowała interfejs *Movable* – w metodzie *move* ma wypisać komunikat „idę”.
8. Utwórz klasę *Vet*, która będzie miała metodę *sayHello* przyjmującą jako parametr obiekt klasy *Cat* i wypisującą powitanie dla tego kota, np. „Witaj Mruczek”.
9. W klasie *Vet* napisz metodę *sayHello* przyjmującą jako parametr obiekt klasy *Dog* i wypisującą powitanie dla tego psa, np. „Witaj Burek”.
10. Zamiast dwóch wersji metody sayHello napisz jedną, która będzie potrafiła przyjąć w parametrze obiekty klas *Cat* oraz *Dog*.
11. Napisz klasy: Rectangle, Circle i Triangle. Każda z tych klas powinna posiadać odpowiednie pola i konstruktor oraz metodę getArea - obliczającą pole:
    1. Prostokąta – wysokość * szerokość
    2. Koła – Pi * promień * promień
    3. Trójkąta – ½ * wysokość * podstawa
12. Napisz metodę, która policzy łączne pole powierzchni paru utworzonych figur.
13. Mając podaną powierzchnię X, która może zostać pokryta przez farbę, napisz metodę, która sprawdzi, czy daną ilością farby można zamalować wszystkie podane figury.
14. Napisz klasę *Calculator*, która będzie miała metodę *add*, dodającą dwie liczby i zwracającą ich wynik. Metoda ta powinna umieć dodawać liczby zespolone (klasa *Complex*) oraz liczby naturalne (klasa *MyNumber*). Jeśli jest taka potrzeba - zmodyfikuj odpowiednio klasy *Complex* i *MyNumber*.
15. Utwórz klasę *Employee* dziedziczącą po klasie *Person*. Dlaczego klasa jest podkreślona na czerwono?
16. Zmodyfikuj klasę *Employee* w taki sposób, żeby przy podaniu roku urodzenia z zakresu innego niż 1900-2020 ustawiało rok urodzenia na 0.
17. Do klasy Employee dodaj pole *salary* oraz metodę *getSalary*. Zrób tak, aby metoda *whoAmI* dla pracownika wyświetlała tekst „Nazywam się Jan Kowalski i zarabiam 1000zł”
18. Utwórz klasę *Manager* dziedziczącą po klasie *Employee*. Dla managera do pensji dodawane jest 10% jako dodatek funkcyjny. Zmodyfikuj odpowiednio metodę *getSalary*. Zrób tak, aby metoda *whoAmI* dla pracownika wyświetlała tekst „Nazywam się manager Jan Kowalski i zarabiam 1000zł”
19. Utwórz klasy *Mammal* (ssak) i *Canidae* (psowate). Zmodyfkuj klasę *Dog* tak, aby dziedziczyła po klasach *Mammal* i *Canidae*
20. Czy któraś z tworzonych przez nas klas mogłaby być oznaczona jako abstrakcyjna?
21. Zastanów się nad modyfikatorami dostępu do klas *Person*, *Employee* oraz *Manager*. Dodaj potrzebne gettery i settery.