Podstawy i języki programowania

Laboratorium 9 - wstęp do obiektowości

mgr inż. Krzysztof Szwarc

krzysztof@szwarc.net.pl

Sosnowiec, 18 grudnia 2017

Zadania

1. Napisz program pobierający od użytkownika informacje o dwóch pracownikach (ich imię, nazwisko oraz pensję). Oblicz średnią wartość pensji i wypisz dane obu pracowników w formie: nazwisko imię zarabia (pensja-średnia) ponad średnią.
2. Przerób napisany program tak, aby użytkownik mógł wprowadzić dane trzech pracowników.
3. Przerób program tak, aby pracownik był opisany również wiekiem (przy wyświetlaniu informacji o zarobkach niech będzie wypisany również wiek).

Klasy

Struktura klasy

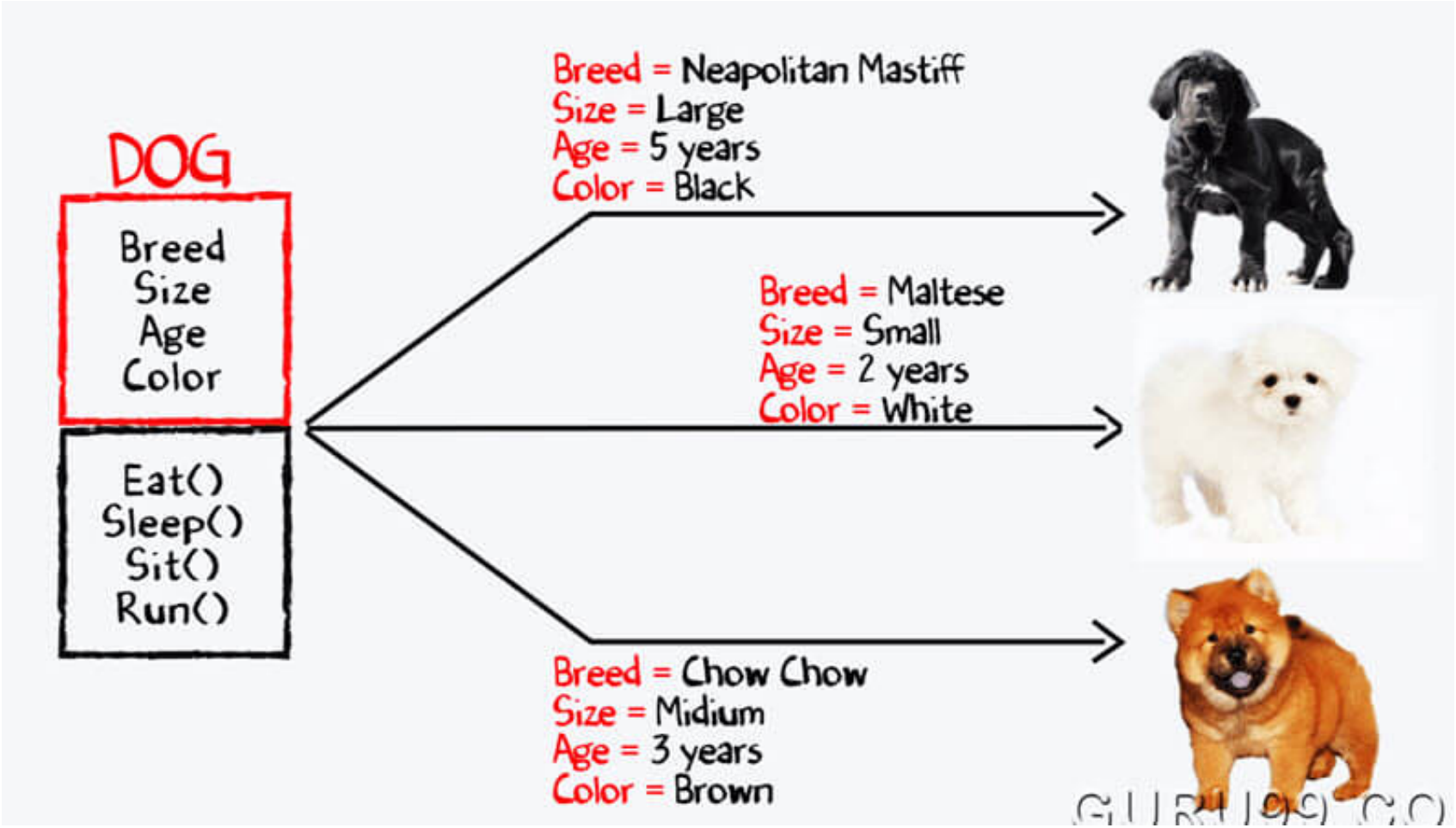
Klasa składa się z pól (zmiennych) oraz metod (funkcji).
Ogólna struktura klasy:

```
modyfikatorDostepu class NazwaKlasy
{
    modyfikatorDostepu typPola nazwaPola;

    modyfikatorDostepu zwracanyTyp
       nazwaMetody(parametry)
    { return zmiennaOZwracanymTypie; }
}
```

Klasa może nie zawierać metod lub pól. Do pól odwołujemy się tak jak do zmiennych lub korzystając ze słowa this w następujący sposób: this.nazwaPola;

Klasa, a obiekt



Źródło: https://cdn.guru99.com/

Przypomnienie modyfikatorów dostępu

Modyfikator dostępu

W Javie wyróżniamy cztery modyfikatory dostępu:

1. public - dostęp do niego ma każdy obiekt.
2. private - dostęp do niego ma wyłącznie właściciel.
3. protected - dostęp do niego ma właściciel, klasy dziedziczące po nim oraz klasy w tym samym pakiecie.
4. default - ustawiany jeżeli nie zadeklarujemy modyfikatora dostępu. Widoczność ograniczona jest do klas znajdujących się w tym samym pakiecie.

Konstruktory

Opis

Każda klasa zawiera konstruktor - specyficzną metodę wywoływaną w momencie tworzenia obiektu (new nazwaKlasy()). Sposób definiowania konstruktora:

```
class NazwaKlasy
{
    NazwaKlasy()
    { }
    NazwaKlasy(String zmienna)
    { }
}
```

Konstruktor, tak jak inne metody można przeciążać (ta sama nazwa metody przyjmująca różne parametry).

Hermetyzacja

Opis

Hermetyzacja (enkapsulacja) polega na odizolowaniu pól i metod od innych klas udostępniając tylko niezbędne elementy. W praktyce polega na stosowaniu modyfikatorów typu private lub protected dla pól i niektórych metod (które nie muszą być używane z zewnątrz). Dostęp do prywatnych pól powinien odbywać się za pomocą metod zwracających i przypisujących odpowiednie wartości (tzw. gettery i settery).

Przykład - hermetyzacja

```
public class Pracownik
{
    private String imie;

    public void ustawImie(String imie)
    {
        this.imie=imie;
    }

    public String zwrocImie()
    {
        return imie;
        // return this.imie;
    }
}
```

Przykład - konstruktor

```
public class Pracownik
{
    private String imie;

    public Pracownik(String imie)
    {
        this.imie=imie;
    }

    public void ustawImie(String imie)
    {
        this.imie=imie;
    }

    public String zwrocImie()
    {
        return imie;
    }
}
```

Przykład - użycie klasy

```
public static void main(String[] args)
{
    Pracownik naszPracownik = new
        Pracownik("Jan");
    System.out.println(naszPracownik.zwrocImie());
    // Jan
    naszPracownik.ustawImie("Andrzej");
    System.out.println(naszPracownik.zwrocImie());
    // Andrzej
}
```

Zadania

1. Napisz klasę Pracownik zawierającą pola reprezentujące imię, nazwisko oraz pensję. Pamiętaj o hermetyzacji.
2. Do napisanej klasy dodaj konstruktor domyślny (bezparametrowy), który będzie ustawiał domyślne dane pracownika.
3. Przerób program z początku zajęć tak, aby korzystał z napisanej klasy.
4. Dodaj pole pracownika płeć i przerób program tak, aby przy wypisywaniu danych pracownika podawał również jego płeć.

Klasy

Modyfikator static

Modyfikator static wtkorzystywany jest do stworzenia pola lub metody dostępnej dla każdej instancji klasy (obiektu). Takie pole/metoda istnieje zawsze (nawet jeżeli nie został utworzony obiekt danej klasy; analogicznie do klasy Math). Pole statyczne może być wykorzystywane jako licznik utworzonych instacji klasy.

Przykład - pole statyczne

```
public class Pracownik
{
    public static int licznik;

    public Pracownik()
    {
        licznik++;
    }
}
```

Przykład - użycie pola statycznego

```
public static void main(String[] args)
{
    Pracownik naszPracownik = new Pracownik();
    naszPracownik = new Pracownik();
    naszPracownik = new Pracownik();
    System.out.println(naszPracownik.licznik);
    // 3
}
```

Zadanie

1. Napisz klasę Student zawierającą następujące pola: imie, nazwisko, numer oraz licznik (statyczny) i bezparametrową metodę przedstawSie (wypisującą tekst: imię nazwisko numer) oraz iluJestStudentow (wypisującą wartość licznika). Pamiętaj o hermetyzacji.

Dziękuję za uwagę