**Класове и обекти**

**Декларация** – това е редът, на който декларираме класа с неговото име и модификатор за достъп. Името на класа по стандарт е с главна буква.  Например:

*public class Person{*

*.............*

*}*

**Модификаторите за достъп** са 3 ключови думи в Java, които се поставят пред променливи, класове и методи по време на декларация, с цел да зададат видимост на дадения деклариран тип. Те са: *private, protected и public*. Първият, поставен пред дадения тип, означава видимост само в рамките на класа, вторият – в класа, в наследниците му и в пакета на класа му и трятият – навсякъде: в класа, в пакета, в наследниците, в други пакети, които са в рамките на проекта. Има и четвърти вариант – да не бъде поставен никакъв модификатор, т.нар. package-private – при него видимостта е само в класа и в пакета.

**Тяло на класа**: в тялото се описва класът – вътре са всички променливи и методи. То е заключено межу две къдрави скоби:

*public class Person {*

*.....тяло.....*

*}*

**Конструктор** – псевдометод, с който се създават впоследствие нови обекти от този клас. Изисквания към конструктора:

* Името му съвпада с името на класа;
* Няма тип на връщания резултат;
* Модификатор public или protected;
* Може да приема аргументи, може и да не приема(по подразбиране).

**Важно:** Ако изрично не напишем конструктор Java си създава по подразбиране конструктор за всеки клас – той е без параметри(default constructor). **Ако създадем какъвто и да е друг конструкто, то този по подразбиране вече не съществува!**

Пример:

*public Person() {*

*….. констр. без параметри*

*}*

*public Person(String name){*

*…. //констр. с параметри*

*}*

**Полета :***всички променливи, с помощта на които се описва даденият клас. Те отразяват състоянието на обекта или помагат на опредлени методи в класа. Всеки обект си има своя стойност, записана в тези полета- един обект не може да споделя с друг обща стойност на тези полета,****освен ако те не са статични****. Статичните полета (декларират се с кл. дума static пред себе си) са такива полета, които са общи за всички инстанции на класа.*

Пример:

*private String name;*

*protected int yearOfBirth;*

*private static int numb = 5;*

***Методи:****методите са начин, за да се опише поведението, което е присъщо на представителите на дадения клас. Метод се декларира така:*

public void speаk() {

System.out.println("Hi, my name is " + name);

}

**Get and Set методи**– използват се от гледна точка на скриването на данни, за да осигури регламентиран и контролиран достъп до полетата на класа. В литературата се наричат още аксесори и мутатори. Това са методи, чрез които се достъпва даденото поле за четене или за модификация. По конвенция започват винаги с get или със set, следвани от името на полето. Например

public String getName() {

return this.name;

}

public void setName(String name) {

this.name = name;

}

Ключовата дума **this**, поставена пред името на полето, означава достъп до **нестатичен** елемент(поле, метод) на класа, в който се намираме.

*this.name = name* ; -> Тук първото name е полето на класа, а второто – параметъра, предаден през метода setName.

От гледна точна на т.нар. **капсулация на данни** стратегията е следната: всяко поле се декларира с модификатор за достъпп *private* и за всяко се пишат get и set методи с модификатор public. По този начин винаги, когато искаме да модифицираме стойността на дадено поле, ще трябва да го правим през метод. Ако искаме да направим валидация преди промяна на стойността му, например полето може да заема стойности само положителни числа от 1 до 10, то можем да направим провека преди да зададем (set) новата стойност на полето, предадена като аргумент на set метода.

1. Пример:
2. package javaExamples;
3. public class Person {
4. private String name;
5. private int yearOfBirth;
6. public Person() {
7. this.name = "";
8. this.yearOfBirth = 0;
9. }
10. public Person(String name, int yearOfbirth) {
11. this.name = name;
12. this.yearOfBirth = yearOfbirth;
13. }
14. public String getName() {
15. return name;
16. }
17. public void setName(String name) {
18. this.name = name;
19. }
20. public int getYearOfBirth() {
21. return yearOfBirth;
22. }
23. public void setYearOfBirth(int yearOfBirth) {
24. this.yearOfBirth = yearOfBirth;
25. }
26. public void speak() {
27. System.out.println("Hi, my name is " + getName());
28. }
29. }

**Пример:**

Дефинирайте клас Student, който съдържа следната информация за студентите: трите имена, курс, специалност, университет, електронна поща и телефонен номер. Декларирайте няколко конструктора за класа Student, които имат различни списъци с параметри (за цялостната информация за даден студент или част от нея). Данните, за които няма входна информация да се инициализират съответно с null или 0. Добавете статично поле в класа Student, в което се съхранява броя на създадените обекти от този клас. Добавете метод в класа Student, който извежда пълна информация за студента



**Задача 1**

Създайте клас Animal, съдържащ следната информация за животните: вид, начин на хранене, възраст, големина. Декларирайте няколко конструктора за класа Animal, които имат различни списъци с параметри (за цялостната информация за дадeно животно или част от нея). Данните, за които няма входна информация да се инициализират съответно с null или 0. Добавете статично поле в класа Animal, в което се съхранява броя на създадените обекти от този клас. Добавете метод в класа Animal, който извежда пълна информация за животното.