



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения

**ОТЧЁТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1
по дисциплине
«Программирование на языке Java»**

Тема: Классы, как новые типы данных. Поля данных и методы

Выполнил студент группы ИКБО-16-20

Пак С.А.

Принял ассистент кафедры ИиППО

Русляков А.А.

Практические работы выполнены «_____» 2021г.

«Зачтено» «_____» 2021г

Москва 2021

СОДЕРЖАНИЕ

ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ.....	3
1. Понятие класса.....	3
2. Определение класса.....	3
2.1. Конвенция кода для класса.....	3
3. Создание экземпляров класса.....	3
4. Операция получения доступа к компонентам класса.....	4
5. Переменные — поля данных класса.....	4
6. Методы класса.....	4
7. Конструкторы.....	5
8. Перегрузка методов.....	5
9. Модификаторы контроля доступа public, private.....	5
10. Информация по сокрытию реализации и инкапсуляции.....	5
РЕШЕНИЕ ЗАДАЧИ.....	7
1. Постановка задачи.....	7
2. Программный код.....	7
3. Вывод программы.....	15
ВЫВОД.....	17

ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ

1. Понятие класса

В Java, класс является определением объектов одного и того же вида. Другими словами, класс — это тип данных, создаваемый программистом для решения задач. Он представляет из себя шаблон, или прототип, который определяет и описывает статические свойства и динамическое поведение, общие для всех объектов одного и того же вида.

Экземпляр класса - реализация конкретного объекта типа класс. Другими словами, экземпляр экземпляра класса. Все экземпляры класса имеют аналогичные свойства, как задано в определении класса.

Класс имеет: имя, переменные, методы.

Имя (или сущность): определяет класс.

Переменные (или атрибуты, состояние, поля данных класса): содержит статические атрибуты класса, или описывают свойства класса (сущности предметной области).

Методы (или поведение, функции, работа с данными): описывают динамическое поведение класса. Другими словами, класс инкапсулирует статические свойства (данные) и динамические модели поведения (операции, которые работают с данными) в одном месте (“контейнере” или “боксе”).

2. Определение класса

В Java, мы используем слово `class` как ключевое или служебное слово, например, чтобы задать определение класса.

```
[AccessControlModifier] class ClassName {  
    // тело класса  
}
```

2.1. Конвенция кода для класса

В соответствии с конвенцией кода на Java имя класса должно быть всегда существительным или словосочетанием из нескольких слов. Все слова должны с прописной буквы (так, называемая верблюжья нотация или *camel notation*).

3. Создание экземпляров класса

Чтобы создать экземпляр класса, вы должны выполнить следующие действия:

- объявить идентификатор экземпляра (имя экземпляра) конкретного класса;
- сконструировать экземпляр класса (то есть выделить память для экземпляра и инициализировать его) с помощью оператора `"new"`.

4. Операция получения доступа к компонентам класса

Доступ к компонентам класса осуществляется с помощью операции получения доступа, а именно операции точка «.»

Переменные и методы, входящие в состав класса, формально называется переменные-поля данных класса и методы класса и являются компонентами класса. Для ссылки на переменную-поле данных класса или метод, вы должны:

- сначала создать экземпляр класса, который вам нужен;
- затем, использовать оператор точка «.» чтобы сослаться на элемент класса (переменную-поле данных или метод класса).

5. Переменные — поля данных класса

Переменная-поле данных имеет имя (идентификатор) и тип, а также может иметь значение определенного типа, например базового или типа определенного программистом ранее. Переменная-поле данных может также быть экземпляром определенного класса (которые будут обсуждаться позже).

Конвенция об именах переменных гласит: имя переменной должно быть существительным или словосочетанием из нескольких слов. Первое слово в нижнем регистре, а остальные слова пишутся с прописной буквы (двугорбая нотация или camel notation).

Формальный синтаксис для определения переменной в Java:

```
[модификатор_доступа] тип имя_перем [= иниц_знач];  
[модификатор_доступа] тип имя_пер-1 [=иниц_знач-1], тип имя_пер-2  
[=иниц_знач-2]  
...;
```

6. Методы класса

Метод (способ как описано в предыдущем разделе):

- принимает параметры из вызова (как в функции);
- выполняет операции, описанные в теле метода, и;
- возвращает часть результата (или void) в точку вызова.

Формальный синтаксис объявления метода в Java:

```
[AccessControlModifier] returnType methodName([argumentList]) {  
    // реализация  
}
```

Конвенция кода о правилах записи имен методов гласит следующее: имя метода должно быть глаголом или начинаться глаголом в виде фразы из нескольких слов, первое слово записывается в нижнем регистре, а остальные слова начинаются с прописной буквы (двугорбая запись).

7. Конструкторы

Конструктор – это специальный метод класса, который имеет то же имя, что используется в качестве имени класса. Конструктор используется для создания и инициализации всех переменных-полей данных класса. Чтобы создать новый экземпляр класса, вы должны использовать специальный оператор "new" с последующим вызовом одного из конструкторов.

Конструктор отличается от обычного метода следующим:

- название метода-конструктора совпадает с именем класса, а имя класса по конвенции, начинается с заглавной буквы;
- конструктор не имеет возвращаемого значения типа (или неявно не возвращает), таким образом, нет объявления типа возвращаемого значения при объявлении;
- конструктор может быть вызван только через оператор «new», он может быть использован только один раз, чтобы инициализировать построенный экземпляр;
- вы не можете впоследствии вызывать конструктор в теле программы подобно обычным методам (функциям);
- конструкторы не наследуются.

8. Перегрузка методов

Перегрузка методов означает, что несколько методов могут иметь то же самое имя метод, но сами методы могут иметь различные реализации (версии). Тем не менее, различные реализации должны быть различимы по списку их аргументов (либо количество аргументов, или типа аргументов, или их порядок).

9. Модификаторы контроля доступа public, private

public: класс / переменная / метод доступным и для всех других объектов в системе.

private: класс / переменная / метод доступным и в пределах только этого класса.

10. Информация по сокрытию реализации и инкапсуляции

Переменные-поля данных класса, как правило, скрыты от внешнего слоя (то есть, недоступны другим классам), для этого осуществляется разграничение доступа или контроль доступа с использованием модификатора доступа - private. Доступ к закрытым переменным - полям класса предоставляются через методы, объявленные с модификатором public.

Использование такого подход соответствует принципу сокрытия информации – принципу инкапсуляции. То есть, объекты могут общаться друг с другом, только через хорошо определенные интерфейсы (публичные методы).

Объектам не позволено знать детали реализации других объектов. Детали реализации всегда скрыты извне или инкапсулированы внутри класса. Скрытие информации облегчает повторное использование класса.

Основное правило при создании определения классов: не объявляйте переменные общедоступными – с модификатором доступа `public`, если у вас на то нет веских оснований, не нарушайте принцип инкапсуляции.

РЕШЕНИЕ ЗАДАЧИ

1. Постановка задачи

Задания:

Необходимо реализовать простейший класс на языке программирования Java. Не забудьте добавить метод toString() к вашему классу. Так-же в программе необходимо предусмотреть класс-тестер для тестирования класса и вывода информации об объекте.

1. Реализуйте простейший класс «Собака»;
2. Реализуйте простейший класс «Мяч»;
3. Реализуйте простейший класс «Книга».

2. Программный код

Файл Dog.java:

```
package ru.mirea.classes;

public class Dog {
    private String name;
    private String owner;
    private int age;

    /**
     * Конструктор по умолчанию
     */
    public Dog() {
        this.name = "Unknown";
        this.owner = "Unknown";
        this.age = 0;
    }

    /**
     * Конструктор, создающий полноценный объект класса Dog
     * @param name        имя собаки
     * @param owner       имя хозяина собаки
     * @param age         возраст собаки
     */
    public Dog(String name, String owner, int age) {
        this.name = name;
        this.owner = owner;
        this.age = age;
    }
}
```

```

/**
 * Геттер для поля name
 * @return      имя собаки
 */
public String getName() {
    return this.name;
}

/**
 * Сеттер для поля name
 * @param newName      новое имя собаки
 */
public void setName(String newName) {
    this.name = newName;
}

/**
 * Геттер для поля owner
 * @return      имя хозяина собаки
 */
public String getOwner() {
    return this.owner;
}

/**
 * Сеттер для поля owner
 * @param newOwner      имя нового хозяина собаки
 */
public void setOwner(String newOwner) {
    this.owner = newOwner;
}

/**
 * Геттер для поля age
 * @return      возраст собаки
 */
public int getAge() {
    return this.age;
}

/**
 * Сеттер для поля age
 * @param newAge      новый возраст собаки
 */
public void setAge(int newAge) {
    this.age = newAge;
}

```



```

/**
 * Конвертирует возраст собаки в человеческий возраст
 * @return          возраст собаки по человеческим меркам
 */
public int toHumanAge() {
    return 7 * this.age;
}

/**
 * Объединяет всю информацию об объекте в одну строку
 * @return          строка с информацией об объекте
 */
public String toString() {
    return this.name + " : " + this.owner + " : " + this.age;
}
}

```

Файл Ball.java:

```
package ru.mirea.classes;
```

```

public class Ball {
    private double radius;
    private String brand;
    private String sport;

    /**
     * Конструктор по умолчанию
     */
    public Ball() {
        this.radius = 0.0;
        this.brand = "None";
        this.sport = "None";
    }

    /**
     * Конструктор, создающий полноценный объект класса Ball
     * @param radius      радиус мяча
     * @param brand       фирма, производившая мяч
     * @param sport       вид спорта, для которого используется
мяч
     */
    public Ball(double radius, String brand, String sport) {
        this.radius = radius;
        this.brand = brand;
        this.sport = sport;
    }
}

```

```

}

/**
 * Геттер для поля radius
 * @return      радиус мяча
 */
public double getRadius() {
    return radius;
}

/**
 * Сеттер для поля radius
 * @param radius      новое значение радиуса мяча
 */
public void setRadius(double radius) {
    this.radius = radius;
}

/**
 * Геттер для поля brand
 * @return      фирма, производившая мяч
 */
public String getBrand() {
    return brand;
}

/**
 * Сеттер для поля brand
 * @param brand      другая фирма, производившая мяч
 */
public void setBrand(String brand) {
    this.brand = brand;
}

/**
 * Геттер для поля sport
 * @return      вид спорта, для которого используется мяч
 */
public String getSport() {
    return sport;
}

/**
 * Сеттер для поля sport
 * @param sport      другой вид спорта, для которого
используется мяч
 */

```

```

public void setSport(String sport) {
    this.sport = sport;
}

/**
 * Объединяет всю информацию об объекте в одну строку
 * @return строка с информацией об объекте
 */
public String toString() {
    return "\tРадиус мяча: " + this.radius + "\n\tФирма: " +
this.brand + "\n\tВид спорта: " + this.sport;
}
}

```

Файл Book.java:

```

package ru.mirea.classes;

```

```

public class Book {
    private String title;
    private String author;
    private int pages;

    /**
     * Конструктор по умолчанию
     */
    public Book() {
        this.title = "Unknown";
        this.author = "Anonymous";
        this.pages = 0;
    }

    /**
     * Конструктор, создающий книгу, имеющую только название
     * @param title название книги
     */
    public Book(String title) {
        this.title = title;
        this.author = "Anonymous";
        this.pages = 0;
    }

    /**
     * Конструктор, создающий книгу, имеющую только название и имя
    автора
     * @param title название книги
     * @param author автор книги

```

```

*/
public Book(String title, String author) {
    this.title = title;
    this.author = author;
}

/**
 * Конструктор, создающий полноценный объект класса Book
 * @param title        название книги
 * @param author       автор книги
 * @param pages        количество страниц в книге
 */
public Book(String title, String author, int pages) {
    this.title = title;
    this.author = author;
    this.pages = pages;
}

/**
 * Геттер для поля title
 * @return            название книги
 */
public String getTitle() {
    return title;
}

/**
 * Сеттер для поля title
 * @param title        новое название книги
 */
public void setTitle(String title) {
    this.title = title;
}

/**
 * Геттер для поля author
 * @return            имя автора книги
 */
public String getAuthor() {
    return author;
}

/**
 * Сеттер для поля author
 * @param author       новый автор книги
 */
public void setAuthor(String author) {

```

```

        this.author = author;
    }

    /**
     * Геттер для поля pages
     * @return количество страниц в книге
     */
    public int getPages() {
        return pages;
    }

    /**
     * Сеттер для поля pages
     * @param pages другое количество страниц в книге
     */
    public void setPages(int pages) {
        this.pages = pages;
    }

    /**
     * Объединяет всю информацию об объекте в одну строку
     * @return строка с информацией об объекте
     */
    public String toString() {
        return "\tНазвание книги: " + this.title
            + "\n\tАвтор книги: " + this.author
            + "\n\tКоличество страниц: " + this.pages;
    }
}

```

Файл DogTest.java:

```

package ru.mirea;

import java.util.Scanner;
import ru.mirea.classes.Dog;

public class DogTest {
    private static final Scanner IN = new Scanner(System.in);

    public static void main(String[] args) {
        System.out.print("Введите имя собаки: ");
        String dogName = IN.nextLine();

        System.out.print("Введите имя хозяина собаки: ");
        String dogOwner = IN.nextLine();
    }
}

```

```

        System.out.print("Введите возраст собаки: ");
        int dogAge = IN.nextInt();

        Dog dog = new Dog(dogName, dogOwner, dogAge);

        System.out.println();
        System.out.println("Информация о собаке по имени " +
dog.getName() + ": " + dog);
        System.out.println("Возраст " + dog.getName() + " по
человеческим меркам: " + dog.toHumanAge());
    }
}

```

Файл BallTest.java:

```

package ru.mirea;

import java.util.Scanner;
import ru.mirea.classes.Ball;

public class BallTest {
    private static final Scanner IN = new Scanner(System.in);

    public static void main(String[] args) {
        System.out.print("Введите радиус мяча: ");
        double radius = IN.nextDouble();

        IN.nextLine();

        System.out.print("Введите фирму, которая произвела мяч: ");
        String brand = IN.nextLine();

        System.out.print("Введите спорт, в котором используется мяч:
");
        String sport = IN.next();

        Ball ball = new Ball(radius, brand, sport);

        System.out.println();
        System.out.println("Информация о мяче:");
        System.out.println(ball);
    }
}

```

Файл BookTest.java:

```

package ru.mirea;

import java.util.Scanner;

```

```

import ru.mirea.classes.Book;

public class BookTest {
    private static final Scanner IN = new Scanner(System.in);

    public static void main(String[] args) {
        System.out.print("Введите название книги: ");
        String title = IN.nextLine();

        System.out.print("Введите имя автора книги: ");
        String author = IN.nextLine();

        System.out.print("Введите количество страниц: ");
        int pages = IN.nextInt();

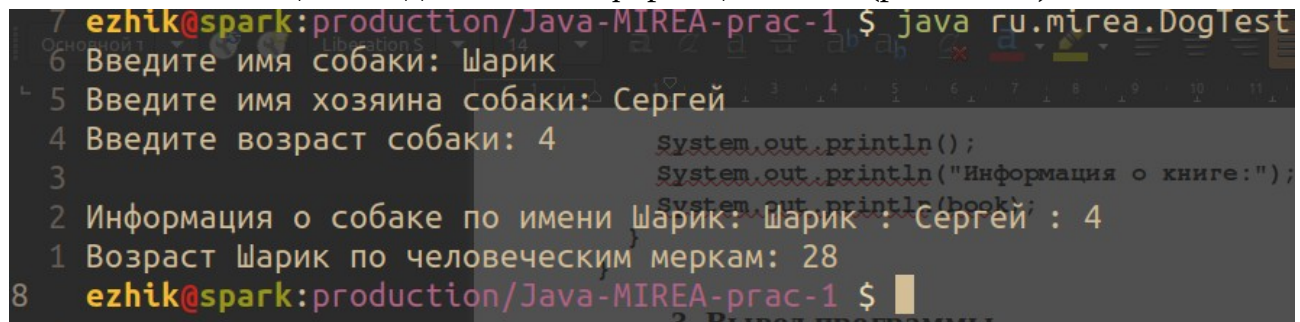
        Book book = new Book(title, author, pages);

        System.out.println();
        System.out.println("Информация о книге:");
        System.out.println(book);
    }
}

```

3. Вывод программы

Во всех программах пользователь поочерёдно вводит значения для полей объектов, и в конце выводится вся информация о них (рис.1, 2, 3).



```

7 ezhik@spark:production/Java-MIREA-prac-1 $ java ru.mirea.DogTest
6 Введите имя собаки: Шарик
5 Введите имя хозяина собаки: Сергей
4 Введите возраст собаки: 4
3
2 Информация о собаке по имени Шарик: Шарик : Сергей : 4
1 Возраст Шарик по человеческим меркам: 28
8 ezhik@spark:production/Java-MIREA-prac-1 $

```

Рис.1 DogTest.java

```

9 ezhik@spark:production/Java-MIREA-prac-1 $ java ru.mirea.BallTest
8 Введите радиус мяча: 23
7 Введите фирму, которая произвела мяч: Nike
6 Введите спорт, в котором используется мяч: football
5                                     System.out.println("Информация о книге:");
4 Информация о мяче:                                     System.out.println(book);
3     Радиус мяча: 23.0     }
2     Фирма: Nike
1     Вид спорта: football
10 ezhik@spark:production/Java-MIREA-prac-1 $

```

3. Вывод программы

В всех программах пользователь поочерёдно вводит объекты, и в конце выводится вся информация о них (при

Рис.2 BallTest.java

```

9 ezhik@spark:production/Java-MIREA-prac-1 $ java ru.mirea.BookTest
8 Введите название книги: Alice in Wonderland
7 Введите имя автора книги: Lewis Carroll
6 Введите количество страниц: 188
5                                     System.out.println();
4 Информация о книге:                                     System.out.println("Информация о книге:");
3     Название книги: Alice in Wonderland                                     System.out.println(book);
2     Автор книги: Lewis Carroll
1     Количество страниц: 188
0 ezhik@spark:production/Java-MIREA-prac-1 $

```

3. Вывод программы

В всех программах пользователь поочерёдно вво

Рис.3 BookTest.java

ВЫВОД

В ходе выполнения работы освоил на практике работу с классами на Java.