**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА   
И ГОСУДАРСТВЕННОЙ СЛУЖБЫ**

**ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**НИЖЕГОРОДСКИЙ ИНСТИТУТ УПРАВЛЕНИЯ – филиал РАНХиГС**

Отчет по программированию – Самоучитель 8 глава

Выполнил: студент группы Иб-321

Наумов Александр Евгеньевич

Оглавление

[Наследование 3](#_Toc133163521)

[Наследование – это … 3](#_Toc133163522)

[Основы наследования 3](#_Toc133163523)

[Конструктор подкласса 3](#_Toc133163524)

[Переопределение методов 3](#_Toc133163525)

[Абстрактные классы и интерфейсы 4](#_Toc133163526)

[Интерфейсная переменная 5](#_Toc133163527)

[Пакеты и уровни доступа 5](#_Toc133163528)

[Правила работы с пакетами: 5](#_Toc133163529)

[Права доступа 6](#_Toc133163530)

# Наследование

## Наследование – это …

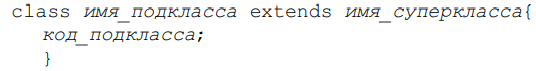
Механизм наследования состоит в том, что на основе уже существующих классов можно создавать новые классы, которые получают, или наследуют, свойства исходных классов.

## Основы наследования

Класс, на основе которого создается новый класс, называется родительским классом, или суперклассом.

Новый класс, который создается на основе суперкласса, называется подклассом.

***Синтаксис*** создания подкласса:



***Примечание:***

В Java запрещено многократное наследование, то есть создание нового класса одновременно с использованием нескольких уже существующих. Вместе с тем, многоуровневое наследование разрешено.

## Конструктор подкласса

Если конструктор суперкласса должен получать аргумент или аргументы и без этого не обойтись никак, то тут есть о чем призадуматься. Выход из ситуации такой, в код конструктора подкласса включается инструкция super с круглыми скобками, в которых перечисляются аргументы, которые передаются конструктору суперкласса. Инструкция super должна быть первой командой в коде конструктора подкласса.

## Переопределение методов

Важное место в наследовании занимает переопределение методов. Суть переопределения методов связана с тем, что программный код унаследованного в подклассе метода может быть переопределен. В результате подкласс имеет такой же метод (с таким же названием), что и суперкласс, но выполняются они по-разному. Для переопределения метода в подклассе необходимо заново описать унаследованный метод в подклассе.

***Примечание:***

Не путайте переопределение метода и перегрузку метода. При перегрузке метода в классе создается несколько вариантов одного метода: название одно и то же, но сигнатуры должны отличаться. При переопределении нужно, чтобы был суперкласс и подкласс. Переопределенный в подклассе имеет такую же сигнатуру, что и метод в суперклассе.

***Примечание:***

Ключевое слово super может использоваться не только для вызова конструктора суперкласса, но и для ссылки на методы (и поля!) суперкласса (используется в случае переопределения методов или перекрытия наследуемых полей). Перекрытие наследуемых полей – это ситуация, когда в подклассе описано поле с таким же именем, что и наследуемое поле из супер класса. Эту ситуацию рассмотрим позже.

***Примечание:***

Названия полей подкласса могут дублировать названия полей из суперкласса. Например, если в суперклассе A описано поле int number, а в подклассе B, который создается на основе класса A, также явно описано поле int number, то по умолчанию, если в классе B выполнить ссылку number, получим доступ к тому полю, что описано в этом подклассе. Чтобы получить доступ к полю, описанному в суперклассе, используем ссылку super.number.

# Абстрактные классы и интерфейсы

Абстрактный метод, в отличие от обычного, имеет сигнатуру, но не содержит блока с программным кодом. Абстрактный метод объявляется с ключевым словом abstract. После сигнатуры абстрактного метода ставится точка с запятой. Класс, содержащий хотя бы один абстрактный метод, также называется абстрактным. Описывается такой класс с использованием все того же ключевого слова abstract.

***Примечание:***

Для абстрактного класса не может быть создан экземпляр класса, то есть объект. Причина очевидна и связана с тем, что абстрактный класс содержит абстрактные методы, которые невозможно выполнять, ведь они не содержат тела метода с программным кодом, а только сигнатуру.

Продолжением идеи абстрактных классов является концепция *интерфейсов*, широко используемая в Java. В отличие от абстрактного класса, который кроме абстрактных методов, может еще содержать обычные методы и поля, в интерфейсе все методы абстрактные. Точнее, в интерфейсе методы только описываются, но не определяются (при этом в описании методов ключевое слово abstract указывать не нужно). Поля также могут присутствовать в интерфейсе, но им сразу нужно присваивать значения, и такие поля являются по умолчанию статическими и неизменными (то есть такими, как если бы они были описаны с идентификаторами final и static). Ну и разумеется, никаких конструкторов в интерфейсе! Им там просто нечего делать.

Интерфейсная переменная — переменная, в качестве типа которой указано имя интерфейса. Такая переменная может ссылаться на объект класса, реализующего интерфейс. Доступ возможен только к методам класса, объявленным в интерфейсе.

# Пакеты и уровни доступа

Пакеты удобно представлять как своеобразные контейнеры, в которых хранятся классы, интерфейсы и другие пакеты (подпакеты). В пределах пакета имена классов, интерфейсов и подпакетов уникальны. Таким образом, через пакеты формируются пространства имен. Для того, чтобы создать пакет, необходимо в начале файла с программным кодом указать ключевое слово package и имя создаваемого пакета (в конце ставится точка с запятой). Если создается пакет подпакета, то имя подпакета указывается вместе с именем пакета (имя пакета, точка и имя подпакета).

***Синтаксис:***

Например, чтобы создать пакет с названием mypack, первой командой в файле с программным кодом указываем такую инструкцию: package mypack; Для создания подпакета subpack, который размещается в пакете mypack, используем такую инструкцию: package mypack.subpack; Все классы и интерфейсы, описанные в файле с package-инструкцией, записываются в соответствующий пакет.

## Правила работы с пакетами:

1. В файле может быть только один открытый класс, то есть класс, описанный с ключевым словом public.

2. Если в файле есть открытый класс, то имя файла должно совпадать с именем этого класса.

3. Если в файле есть package-инструкция, то она указывается первой строкой кода.

4. В файле может быть несколько import-инструкций. Все они размещаются в начале файла, но после package-инструкции (если такая есть).

## Права доступа

