Contents

1	□ Конспекты курса: "Продвинутая Java Платформа"	2
	1.1 Лектор: Александр Маторин	2
	1.2 🛘 Описание курса	2
	1.3 🛘 Оглавление	3
	1.3.1 🛮 Лекции	3
	1.4 🛮 Как использовать	3
	1.5 [] Сотрудничество	3
	1.6] Лицензия	3
4	□ Лекция 1 — Основы синтаксиса Java + Настройка окружения и работа в предоставляющим пред	
	Intellij IDEA	3
	2.1 □ Основные темы	3 4
	2.2.1 Установка JDK	4
	2.2.2 Переменная окружения РАТН	4
	2.2.3 CLASSPATH	4
	2.3. Педота с Intellij IDEA	4
	2.3.1 Почему IntelliJ IDEA?	4
	2.3.1 Почему Intellij IDEA:	5
	2.4 Полезные Live Templates (шаблоны кода)	5
	2.6 Рефакторинги в IntelliJ IDEA	5
	2.6.1 ☐ Extract Method (Вынести метод)	5
	2.6.2 ☐ Inline Method (Встроить метод)	6
	2.7 ПОсновы синтаксиса Java	6
	2.7.1 Структура программы	6
	2.7.2 Переменные и типы	6
	2.7.3 Управляющие конструкции	6
	2.8 Советы для новичков	7
	2.9 Полезные ссылки	7
	2.10 Передача параметров по значению	9
	2.10.1Пример:	9
	2.10.2Ho:	9
	2.11∏ Пакеты Java	10
	2.11.1Основные пакеты:	10
	2.11.2Импорт:	10
	2.12 Класс Object — корень иерархии	10
	2.12.1Основные методы:	10
	2.131.1.1 equals	10
	2.13.1 Поведение по умолчанию	11
	2.13.2 Переопределение	11
	2.13.3 Koнтракт equals()	11
	2.141.1.2 hashCode	11
	2.14.1 Kонтракт hashCode()	12
	2.14.2 Реализация по умолчанию	12
	2.14.3 Почему важно переопределять hashCode() вместе с equals()	12

	2.15 Правильное переопределение	12
	2.15.1Способ 1: Вручную	12
	2.15.2Способ 2: Через Lombok (если используется)	13
	$2.15.3$ Cποcoδ 3: B IntelliJ IDEA \rightarrow Alt + Insert \rightarrow equals() and hashCode()	13
	2.16 Частые ошибки	13
	2.17 Советы	14
	2.18 [Koнтракт equals() и hashCode()	14
	2.18.1Почему это важно?	14
	2.18.2Антипаттерн:	14
	2.19 ☐ Частые ошибки	14
	2.20 Переполнение (Overflow)	15
	2.20.1Пример c int:	15
	2.20.2Как избежать?	15
	2.21 BigDecimal — для точных вычислений (деньги!)	15
	2.21.1Пример проблемы:	15
	2.21.2Решение — BigDecimal:	15
	2.21.3Операции:	16
	2.22 StringBuilder — эффективная работа со строками	16
	2.22.1Проблема:	16
	2.22.2Решение — StringBuilder:	16
	2.22.3Основные методы:	16
	2.23 Советы	17
	2.24 Полезные ссылки	17
		,,
L	□ Конспекты курса: "Продвинутая Java Платформа	a"
	1 Harris A Manager	

1.1 Лектор: Александр Маторин

Кафедра БИТ, МФТИ

Ведётся студентом: Андрей Бодакин

1.2 🛘 Описание курса

Курс посвящён глубокому изучению языка **Java**, начиная с основ синтаксиса и заканчивая продвинутыми темами: многопоточность, JVM, коллекции, работа с памятью и многое другое.

Лектор: **Александр Маторин** — практик, эксперт в Java-экосистеме.

Цель репозитория — систематизировать знания, вести конспекты лекций, делиться материалами и примерами кода.

1.3 Поплавление
1.3.1 🛘 Лекции
 Лекция 1 — Основы синтаксиса Java + Настройка окружения и работа в IntelliJ IDEA Лекция 2 — Примитивные типы, классы-обёртки, Пакеты Java, Object, equals/hashCode Следующие лекции будут добавляться по мере прохождения курса
1.4 🛘 Как использовать
 Все конспекты в формате Markdown — легко читать на GitHub. Примеры кода — в папке code/ (если есть).
• Pull Request'ы и Issues приветствуются — если нашли ошибку или хотите дополнить материал.
Если ты тоже учишься на курсе — присылай свои конспекты, дополнения, примеры
кода!
Открыт для совместного ведения и улучшения материалов.
1.6 🛘 Лицензия
Этот репозиторий распространяется под лицензией MIT — используйте свободно для обучения и распространения знаний.
2 🛘 Лекция 1 — Основы синтаксиса Java + Настройка
окружения и работа в IntelliJ IDEA
2.1 🛘 Основные темы
• Установка и настройка JDK
• Компиляция и запуск через javac и java • Переменные среды: PATH, CLASSPATH
• Выбор и настройка IDE (IntelliJ IDEA)
• Горячие клавиши и рефакторинги в IntelliJ IDEA
• Основы синтаксиса: классы, методы, переменные, управляющие конструкции

2.2 **Пастройка Java-окружения**

2.2.1 Установка JDK

$ ightarrow$ Скачать можно c: - Oracle JDK - OpenJDK (Adoptium / Temurin) \leftarrow рекомендуется Проверка установки:
\ExtensionTok{java} \AttributeTok{{-}version} \ExtensionTok{javac} \AttributeTok{{-}version}
→ Должны вывести версию Java и компилятора.
2.2.2 Переменная окружения РАТН
→ РАТН — список директорий, где система ищет исполняемые файлы. □ Добавь путь к bin JDK в РАТН: Linux/macOS (в ~/.bashrc или ~/.zshrc):
lem:lem:lem:lem:lem:lem:lem:lem:lem:lem:
Windows : - Панель управления → Система → Дополнительные параметры → Переменные среды → РАТН → Добавить путь, например: C:\Program Files\Java\jdk-21\bin
2.2.3 CLASSPATH
→ Указывает JVM, где искать .class-файлы и библиотеки. □ Обычно не нужно настраивать вручную при работе с IDE или Maven/Gradle. → Если компилируешь вручную:
lem:lem:lem:lem:lem:lem:lem:lem:lem:lem:

2.3 🛘 Работа с IntelliJ IDEA

□. — текущая директория.

2.3.1 Почему IntelliJ IDEA?

• Самая популярная и мощная IDE для Java.

 \square ; — разделитель в Windows, : — в Linux/macOS.

- Умное автодополнение, рефакторинги, отладка, интеграция с Maven/Gradle, Git.
- Community Edition бесплатна и достаточно для обучения.

2.4 **Ш** Горячие клавиши IntelliJ IDEA (must-have)

Действие	Windows/Linux	macOS
Автодополнение	Ctrl + Space	Cmd + Space
Быстрое исправление / подсказки	Alt + Enter	Option + Enter
Запуск программы	Shift + F10	Ctrl + R
Отладка	Shift + F9	Ctrl + D
Поиск по проекту	Ctrl + Shift + F	Cmd + Shift + F
Поиск класса	Ctrl + N	Cmd + O
Поиск файла	Ctrl + Shift + N	Cmd + Shift + O
Переход к определению	Ctrl + B	Cmd + B
Рефакторинг: переименование	Shift + F6	Shift + F6
Закомментировать строку	Ctrl + /	Cmd + /
Форматирование кода	Ctrl + Alt + L	Cmd + Option + L
Открыть структуру класса	Ctrl + F12	Cmd + F12

2.5 🛘 Полезные Live Templates (шаблоны кода)

Шаблон	Результат	Описание
sout	<pre>System.out.println();</pre>	Быстрый вывод в консоль
iter	<pre>for (Type item : collection) { }</pre>	Цикл for-each
psvm	<pre>public static void main(String[] args) { }</pre>	Главный метод
itar	for (int i = 0; i < arr.length; i++) { }	Цикл по индексу
ifn	if (var == null) { }	Проверка на null
inn	if (var != null) { }	Проверка на не-null

[→] Просто введи шаблон и нажми Таь.

2.6 🛘 Рефакторинги в IntelliJ IDEA

2.6.1 [Extract Method (Вынести метод)

Выдели код \rightarrow Ctrl + Alt + M \rightarrow дай имя методу \rightarrow готово! **Было:**

```
\KeywordTok{public} \DataTypeTok{void} \FunctionTok{process}\OperatorTok{()} \OperatorTok\\{}
  \DataTypeTok{int}\NormalTok{ a }\OperatorTok{=} \DecValTok{5}\OperatorTok{;}
  \DataTypeTok{int}\NormalTok{ b }\OperatorTok{=} \DecValTok{10}\OperatorTok{;}
  \DataTypeTok{int}\NormalTok{ sum }\OperatorTok{=}\NormalTok{ a }\OperatorTok{+}\NormalTok{ b}\OperatorTok{;}
  \BuiltInTok{System}\OperatorTok{.}\FunctionTok{out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\Stringle)\OperatorTok{\}}
```

	T	9	П	n	
•	1	u	., .	w	٠

2.6.2 ☐ Inline Method (Встроить метод)

Если метод слишком простой — можно "встроить" его обратно: Ctrl + Alt + N

→ Полезно при оптимизации или упрощении.

2.7 🛘 Основы синтаксиса Java

2.7.1 Структура программы

\KeywordTok{public} \KeywordTok{class}\NormalTok{ HelloWorld }\OperatorTok{\{} \KeywordTok{public} \DataTypeTok{static} \DataTypeTok{void} \FunctionTok{main}\OperatorTok{(}\BuiltInTok{StaticFunctionTok{out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\\OperatorTok{\}} \OperatorTok{\}} \OperatorTok{\}} \OperatorTok{\}}

→ Каждая программа начинается с main.

2.7.2 Переменные и типы

\DataTypeTok{int}\NormalTok{ age }\OperatorTok{=} \DecValTok{25}\OperatorTok{;}
\DataTypeTok{double}\NormalTok{ price }\OperatorTok{=} \FloatTok{19.99}\OperatorTok{;}
\DataTypeTok{boolean}\NormalTok{ isActive }\OperatorTok{=} \KeywordTok{true}\OperatorTok{;}
\BuiltInTok{String}\NormalTok{ name }\OperatorTok{=} \StringTok{"Alice"}\OperatorTok{;}

2.7.3 Управляющие конструкции

```
\ControlFlowTok{if} \OperatorTok{(}\NormalTok{age }\OperatorTok{\textgreater{}=} \DecValTok{18}\OperatorTok{)}
\BuiltInTok{System}\OperatorTok{.}\FunctionTok{out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\Stri
\OperatorTok{\}} \ControlFlowTok{else} \OperatorTok{\}\
\BuiltInTok{System}\OperatorTok{.}\FunctionTok{out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\Stri
\OperatorTok{\}}
\ControlFlowTok{for} \OperatorTok{(}\DataTypeTok{int}\NormalTok{ i }\OperatorTok{=} \DecValTok{0}\OperatorTok{;
\BuiltInTok{System}\OperatorTok{.}\FunctionTok{out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\Norm\OperatorTok{\}}
\ControlFlowTok{\}}
\ControlFlowTok{\}}
\ControlFlowTok{\}\}
\ControlFlowTok{\}}
\ControlFlowTok{\}}
\OperatorTok{\}}
\OperatorTok
```

2.8 🛘 Советы для новичков

- Всегда проверяй, что java -version работает в терминале.
- Не бойся использовать Alt + Enter IntelliJ IDEA часто знает, как исправить ошибку.
- Учись пользоваться рефакторингами они экономят кучу времени.

2.9 Полезные ссылки

- Скачать IntelliJ IDEA Community
- OpenJDK (Adoptium)
- Горячие клавиши IntelliJ IDEA (официальная шпаргалка)
- Теория по Java

```
- `StringBuilder` -
##
 Java 8
| `byte` | 8
                   | 1
                              | -128
                                     127
                                                        1 `0`
                                                        1 '0'
| `short` | 16
                   1 2
                              | -32768
                                       32767
                  14
                                      2^31-1
| `int` | 32
                              | -2^31
                                                        1 `0`
| `long`
                                                        | `OL`
        | 64
                  | 8
                              | -2^63
                                      2^63-1
                  | 4
| `float` | 32
                              | ±3.4e38 (7 )
                                                      | `0.0f`
| `double` | 64
                  8 |
                              | ±1.7e308 (15 )
                                                      | `0.0d`
| `char` | 16
                   | 2
                              * | `true` / `false`
                                                    | `false`
| `boolean`|
               * |
                              JVM -
                                                       `int` (32 )
         `boolean`
    - (Wrapper Classes)
##
1 - 1
|-----|
| `int` | `Integer`
| `long` | `Long`
| `double` | `Double`
| `boolean`| `Boolean`
| `char` | `Character`
       1 ...
###
             (`List<Integer>`, `List<int>`).
         `null`.
         : `Integer.parseInt()`, `Character.isDigit()` . .
```

 $\label{linTok} $$ \BuiltInTok{Integer}\NormalTok{ x }\Omega_{=} \KeywordTok{null}\Omega_{;;} $$ \Delta Tok{int}\NormalTok{ y }\Omega_{=}\NormalTok{ x}\Omega_{;;} \CommentTok{// } \AllPointerExcept $$$

→ Всегда проверяй на null перед распаковкой.

2.10 Передача параметров по значению

□ В Java всё передаётся по значению — даже объекты!

→ При передаче объекта — копируется **ссылка на объект**, а не сам объект.

2.10.1 Пример:

→ Объект изменился, потому что мы изменяли данные по скопированной ссылке.

2.10.2 Ho:

\OperatorTok{\}}

\DataTypeTok{static} \DataTypeTok{void} \FunctionTok{reassign}\OperatorTok{(}\NormalTok{Person person}\OperatorTok{\NormalTok{ person }\OperatorTok{=} \KeywordTok{new} \FunctionTok{Person}\OperatorTok{(}\StringTok{"Charlie" \OperatorTok{\}}

→ После вызова reassign(p) — p.name всё ещё "Alice".

2.11 Пакеты Java

Пакеты — это пространства имён для классов.

2.11.1 Основные пакеты:

- java.lang автоматически импортируется (String, Object, System, Math).
- java.util коллекции, дата/время, Scanner, Random.
- java.io ввод/вывод, файлы.
- java.time современные даты (Java 8+).

2.11.2 Импорт:

```
\KeywordTok{import} \ImportTok{java}\OperatorTok{.}\ImportTok{util}\OperatorTok{.}\ImportTok{List}\OperatorTok{\KeywordTok{import} \ImportTok{java}\OperatorTok{.}\ImportTok{util}\OperatorTok{.}\ImportTok{ArrayList}\OperatorTok{\KeywordTok{import} \ImportTok{java}\OperatorTok{.}\ImportTok{time}\OperatorTok{.}\ImportTok{LocalDate}\OperatorTok{\CommentTok{// :}
\CommentTok{// :}
\KeywordTok{import} \ImportTok{java}\OperatorTok{.}\ImportTok{util}\OperatorTok{.*;}
```

 $\square *$ — не нагружает приложение, только упрощает написание кода.

2.12 🛘 Класс Object — корень иерархии

- → Корневой класс всей иерархии в Java.
- → Любой класс наследник Object (явно или неявно).

2.12.1 Основные методы:

- toString() строковое представление объекта.
- equals(Object obj) сравнение объектов.
- hashCode() хеш-код для использования в хеш-таблицах.
- getClass() получить класс объекта.
- clone() неглубокое клонирование (осторожно!).
- finalize() освобождение ресурсов перед удалением, deprecated (Java 9+).

2.13 1.1.1 equals

\KeywordTok{public} \DataTypeTok{boolean} \FunctionTok{equals}\OperatorTok{(}\AttributeTok{@Nullable} \BuiltInT

| Назначение: проверяет, равны ли два объекта логически (по содержимому),

а не физически (по ссылке).

	2.13.1	□ Поведение по умолчанию
--	--------	--------------------------

ightarrow В классе Object метод equals() сравнивает **ссылки**:

\BuiltInTok{Object}\NormalTok{ a }\OperatorTok{=} \KeywordTok{new} \BuiltInTok{Object}\OperatorTok{();} \BuiltInTok{Object}\NormalTok{ b }\OperatorTok{=} \KeywordTok{new} \BuiltInTok{Object}\OperatorTok{();} \BuiltInTok{System}\OperatorTok{.}\FunctionTok{out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{()\NormalTok{Out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{()\NormalTok{Out}\OperatorTok{Out}\Opera

→ Это эквивалентно a == b.

2.13.2 Переопределение

→ В подклассах equals() **часто переопределяют**, чтобы сравнивать объекты по полям:

\AttributeTok{@Override}
\KeywordTok{public} \DataTypeTok{boolean} \FunctionTok{equals}\OperatorTok{(}\BuiltInTok{Object}\NormalTok{ o}\\
\ControlFlowTok{if} \OperatorTok{(}\KeywordTok{this} \OperatorTok{==}\NormalTok{ o}\OperatorTok{)} \ControlFlowTok{if} \OperatorTok{(}\NormalTok{o}\OperatorTok{==} \KeywordTok{null} \OperatorTok{||} \FunctionTok{\OperatorTok{||} \FunctionTok{\OperatorTok{o}\

2.13.3 [] Kohtpakt equals()

Для **ненулевых объектов** метод equals() должен задавать **отношение эквивалентности**:

- 1. **Рефлексивность**: $x.equals(x) \rightarrow true$
- 2. **Симметричность**: если x.equals(y) == true, то y.equals(x) == true
- 3. **Транзитивность**: если x.equals(y) == true и y.equals(z) == true, то x.equals(z) == true
- 4. **Консистентность**: если данные объекта не менялись, то x.equals(y) должен возвращать одно и то же значение при повторных вызовах.
- 5. Для любого ненулевого x: x.equals(null) == false

 \square Нарушение контракта \rightarrow непредсказуемое поведение в коллекциях (HashSet, HashMap и др.).

2.14 1.1.2 hashCode

□ **Назначение**: возвращает целочисленный хеш-код объекта. Используется в хеш-структурах: HashMap, HashSet, HashTable и др.

2.14.1	☐ Kohtpakt hashCode()
1. Ko	нсистентность : если дан

- 1. **Консистентность**: если данные объекта не менялись, hashCode() должен возвращать **одно и то же значение** при каждом вызове.
- 2. **Согласованность с** equals(): если x.equals(y) == true, To x.hashCode() == y.hashCode() **обязательно**.
- 3. **Необязательное условие**: если x.equals(y) == false, то x.hashCode() и y.hashCode() **могут совпадать** это называется **коллизия хешей** (нормально для хеш-таблиц).

2.14.2 🏻 Реализация по умолчанию

- → В ранних версиях JVM hashCode() возвращал **адрес объекта в памяти**.
- → Сейчас используется псевдослучайное число, которое: Генерируется при первом вызове hashCode(). Записывается в заголовок объекта (object header). Не меняется в течение жизни объекта, даже если объект перемещается сборщиком мусора.
 - □ Почему изменили?
 - При маленьком heap-е адреса были близки → хеши были не равномерны → **ухудшалась производительность хеш-таблиц**.
 - Новая реализация даёт лучшее распределение хешей → меньше коллизий
 - \rightarrow быстрее работа HashMap.

$2.14.3\ \square$ Почему важно переопределять hashCode() вместе c equals()

 \rightarrow Представь, что ты положил объект в HashMap, а потом изменил поле, участвующее в equals(), но не обновил hashCode():

\NormalTok{Person p }\OperatorTok{=} \KeywordTok{new} \FunctionTok{Person}\OperatorTok{(}\StringTok{"Alice"}\OperatorTok{map}\OperatorTok{.}\FunctionTok{put}\OperatorTok{(}\NormalTok{p}\OperatorTok{,} \StringTok{"value"}\\NormalTok{p}\OperatorTok{.}\FunctionTok{setName}\OperatorTok{(}\StringTok{"Bob"}\OperatorTok{);} \CommentTok{/\dots null operatorTok{.}\FunctionTok{get}\OperatorTok{(}\NormalTok{p}\OperatorTok{);} \CommentTok{// \rightarrow null operatorTok{.}\FunctionTok{get}\OperatorTok{(}\NormalTok{p}\OperatorTok{);} \CommentTok{// \rightarrow null operatorTok{.}\FunctionTok{get}\OperatorTok{(}\NormalTok{p}\OperatorTok{);} \CommentTok{// \rightarrow null operatorTok{.}\FunctionTok{get}\OperatorTok{.}\FunctionTok{.}\FunctionTok{get}\OperatorTok{.}\FunctionT

 \rightarrow Решение: поля, используемые в equals() и hashCode(), должны быть неизменяемыми (immutable).

2.15 Правильное переопределение

2.15.1 Способ 1: Вручную

```
\AttributeTok{@Override}
\KeywordTok{public} \DataTypeTok{boolean} \FunctionTok{equals}\OperatorTok{(}\BuiltInTok{Object}\NormalTok{ o}\
   \ControlFlowTok{if} \OperatorTok{(}\KeywordTok{this} \OperatorTok{==}\NormalTok{ o}\OperatorTok{)} \Control
   \ControlFlowTok{if} \OperatorTok{(}\NormalTok{o} \OperatorTok{==} \KeywordTok{null} \OperatorTok{||} \Funct
\NormalTok{    Person person }\OperatorTok{=} \OperatorTok{(}\NormalTok{Person}\OperatorTok{(}\NormalTok{ o}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0}\OperatorTok{0
```

2.15.2 Способ 2: Через Lombok (если используется)

```
\AttributeTok{@EqualsAndHashCode}
\KeywordTok{public} \KeywordTok{class}\NormalTok{ Person }\OperatorTok{\{}
\KeywordTok{private} \BuiltInTok{String}\NormalTok{ name}\OperatorTok{;}
\KeywordTok{private} \BuiltInTok{Integer}\NormalTok{ age}\OperatorTok{;}
\OperatorTok{\}}
```

2.15.3 Cπoco6 3: B IntelliJ IDEA → Alt + Insert → equals() and hashCode()

→ IDE сгенерирует корректный код.

```
\AttributeTok{@Override}
\KeywordTok{public} \DataTypeTok{boolean} \FunctionTok{equals}\OperatorTok{(}\BuiltInTok{Object}\NormalTok{ o}\\
   \ControlFlowTok{iff} \OperatorTok{(}\KeywordTok{this} \OperatorTok{==}\NormalTok{ o}\OperatorTok{)} \Control
   \ControlFlowTok{iff} \OperatorTok{(}\NormalTok{ o}\OperatorTok{==}\NormalTok{ o}\OperatorTok{||} \FunctionTok{||} \FunctionTok{==}\KeywordTok{null} \OperatorTok{||} \FunctionTok{||} \FunctionTok{\OperatorTok{||}} \FunctionTok{\OperatorTo
```

2.16 П Частые ошибки

на основе переданных полей.

- Сравнение через == для объектов → сравниваются ссылки, а не содержимое.
- Забыли @Override → можно случайно создать перегрузку, а не переопределение.

- Не переопределили hashCode() → проблемы с HashMap.
- Использовали изменяемые поля в hashCode() → объект "сломается" в HashSet, если поле изменится.
- Использовали float/double в hashCode() без округления → нестабильность из-за точности.

2.17 □ Советы

- Всегда переопределяй equals() и hashCode() вместе.
- Используй java.util.Objects безопасно и читаемо.
- B IntelliJ IDEA: Alt + Insert → генерация методов экономит время.
- Тестируй поведение в НаѕһМар это частый вопрос на собеседованиях.
- Поля в hashCode() и equals() лучше делать final.

2.18 [] Koнтракт equals() и hashCode()

Если два объекта равны по equals() — их hashCode() должен быть одинаковым.

2.18.1 Почему это важно?

→ HashMap, HashSet, HashTable используют hashCode() для определения "корзины", а equals() — для точного сравнения.

2.18.2 Антипаттерн:

```
\AttributeTok{@Override}
\KeywordTok{public} \DataTypeTok{boolean} \FunctionTok{equals}\OperatorTok{(}\BuiltInTok{Object}\NormalTok{ o}\
\CommentTok{// ... }
\OperatorTok{\}}
\CommentTok{// hashCode()}
```

→ Объекты могут "потеряться" в HashMap.

2.19 | Частые ошибки

- Сравнение через == для объектов → сравниваются ссылки, а не содержимое.
- Забыли @Override → можно случайно создать перегрузку, а не переопределение.
- Не переопределили hashCode() → проблемы с HashMap.
- Использовали изменяемые поля в hashCode() → объект "сломается" в HashSet, если поле изменится.

2.20 Переполнение (Overflow)

→ Происходит, когда результат операции выходит за пределы диапазона типа.

2.20.1 Пример с int:

\DataTypeTok{int}\NormalTok{ max }\OperatorTok{=} \BuiltInTok{Integer}\OperatorTok{.}\FunctionTok{MAX_VALUE}\C \DataTypeTok{int}\NormalTok{ overflow }\OperatorTok{=}\NormalTok{ max }\OperatorTok{+} \DecValTok{1}\OperatorTok{BuiltInTok{System}\OperatorTok{.}\FunctionTok{out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\NormalTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{.}\Fu

→ Никакого исключения — просто "заворачивается" (как одометр в машине).

2.20.2 Как избежать?

\OperatorTok{\}}

- Используй long для больших чисел.
- Используй Math.addExact(), Math.multiplyExact() бросают ArithmeticException при переполнении.

\ControlFlowTok{try} \OperatorTok{\{}
\DataTypeTok{int}\NormalTok{ result }\OperatorTok{=} \BuiltInTok{Math}\OperatorTok{.}\FunctionTok{addExact}
\OperatorTok{\}} \ControlFlowTok{catch} \OperatorTok{(}\BuiltInTok{ArithmeticException}\NormalTok{ e}\OperatorTok{\BuiltInTok{System}\OperatorTok{.}\FunctionTok{Out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\Stri

• Для критических вычислений — используй BigInteger.

2.21 🛘 BigDecimal — для точных вычислений (деньги!)

ightarrow float и double **не подходят** для финансовых расчётов — из-за ошибок округления.

2.21.1 Пример проблемы:

```
\DataTypeTok{double}\NormalTok{ a }\OperatorTok{=} \FloatTok{0.1}\OperatorTok{;}
\DataTypeTok{double}\NormalTok{ b }\OperatorTok{=} \FloatTok{0.2}\OperatorTok{;}
\BuiltInTok{System}\OperatorTok{.}\FunctionTok{out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\NormalTok{Out}\OperatorTok{.}\FunctionTok{println}\OperatorTok{(}\NormalTok{Out}\OperatorTok{.}\FunctionTok{Out}\OperatorTok{.}\FunctionTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\OperatorTok{Out}\Oper
```

2.21.2 Решение — BigDecimal:

\BuiltInTok{BigDecimal}\NormalTok{ a }\OperatorTok{=} \KeywordTok{new} \BuiltInTok{BigDecimal}\OperatorTok{()\S\BuiltInTok{BigDecimal}\NormalTok{ b }\OperatorTok{=} \KeywordTok{new} \BuiltInTok{BigDecimal}\OperatorTok{()\S\BuiltInTok{BigDecimal}\NormalTok{ sum }\OperatorTok{=}\NormalTok{ a}\OperatorTok{.}\FunctionTok{add}\OperatorTok{BuiltInTok{System}\OperatorTok{.}\FunctionTok{out}\OperatorTok{.}\FunctionTok{()\NormalTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{.}\FunctionTok{OperatorTok{.}\FunctionTok{.}\Fu

□ Всегда создавай BigDecimal из String, а не из double!

2.21.3 Операции:

\NormalTok{a}\OperatorTok{.}\FunctionTok{add}\OperatorTok{(}\NormalTok{b}\OperatorTok{)}
\NormalTok{a}\OperatorTok{.}\FunctionTok{subtract}\OperatorTok{(}\NormalTok{b}\OperatorTok{)}
\NormalTok{a}\OperatorTok{.}\FunctionTok{multiply}\OperatorTok{(}\NormalTok{b}\OperatorTok{)}
\NormalTok{a}\OperatorTok{.}\FunctionTok{divide}\OperatorTok{(}\NormalTok{b}\OperatorTok{,} \DecValTok{2}\OperatorTok{.

→ Используй BigDecimal для: - Денег - Процентов - Точных научных расчётов

2.22 🛘 StringBuilder — эффективная работа со строками

→ String в Java **неизменяем (immutable)** → каждая операция "a" + "b" создаёт новый объект.

2.22.1 Проблема:

```
\label{thm:control} $$ \BuiltInTok{String}\NormalTok{ result }\OmegaeratorTok{=} \StringTok{""}\OperatorTok{;} $$ \ControlFlowTok{for} \OmegaeratorTok{(}\DataTypeTok{int}\NormalTok{ i }\OmegaeratorTok{=} \OmegaecValTok{0}\OperatorTok{;} \NormalTok{ result }\OmegaeratorTok{+=} \StringTok{"a"}\OmegaeratorTok{;} \CommentTok{// $\leftarrow$ 1000 ! \OperatorTok{\}} $$
```

→ Медленно и расходует память.

2.22.2 Решение — StringBuilder:

\BuiltInTok{StringBuilder}\NormalTok{ sb }\OperatorTok{=} \KeywordTok{new} \BuiltInTok{StringBuilder}\OperatorToK\ControlFlowTok{for} \OperatorTok{(}\DataTypeTok{int}\NormalTok{ i }\OperatorTok{=} \DecValTok{0}\OperatorTok{;} \NormalTok{ sb}\OperatorTok{.}\FunctionTok{append}\OperatorTok{(}\StringTok{"a"}\OperatorTok{);} \CommentTok\OperatorTok{\}}
\BuiltInTok{String}\NormalTok{ result }\OperatorTok{=}\NormalTok{ sb}\OperatorTok{.}\FunctionTok{toString}\OperatorTok{\}}

→ B **100+ раз быстрее** для больших объёмов.

2.22.3 Основные методы:

```
\NormalTok{sb}\OperatorTok{.}\FunctionTok{append}\OperatorTok{(}\StringTok{"text"}\OperatorTok{)}
\NormalTok{sb}\OperatorTok{.}\FunctionTok{insert}\OperatorTok{(}\DecValTok{0}\OperatorTok{,} \StringTok{"prefix
\NormalTok{sb}\OperatorTok{.}\FunctionTok{delete}\OperatorTok{(}\DecValTok{0}\OperatorTok{,} \DecValTok{5}\Oper
\NormalTok{sb}\OperatorTok{.}\FunctionTok{reverse}\OperatorTok{()}
\NormalTok{sb}\OperatorTok{.}\FunctionTok{toString}\OperatorTok{()}
```

 $\ \square$ Если нужна потокобезопасность — используй StringBuffer (но он медленнее из-за синхронизации).

2.23 □ Советы

- Всегда переопределяй equals() и hashCode() вместе.
- Используй java.util.Objects безопасно и читаемо.
- B IntelliJ IDEA: Alt + Insert → генерация методов экономит время.
- Для денег только BigDecimal.
- Для конкатенации строк в цикле только StringBuilder.
- Проверяй переполнение в критических местах используй Math.*Exact().

2.24 Полезные ссылки

• Oracle: Primitive Data Types

Oracle: BigDecimalOracle: StringBuilder

Хабр: Контракт equals/hashCodeBaeldung: Guide to hashCode()

• Baeldung: BigDecimal

"