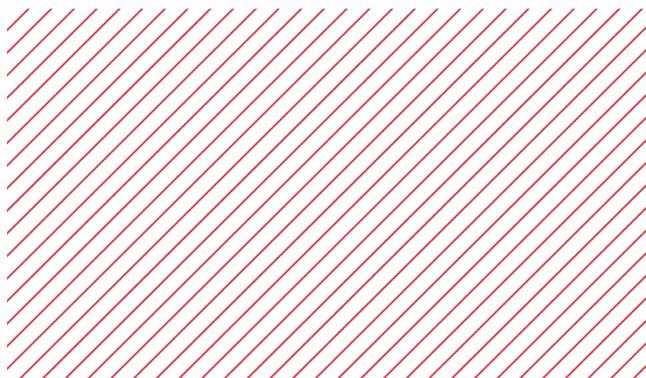


академия
больших
данных



Advanced Scala

Корольков Алексей



Структура курса

1. Введение в Scala. Основные конструкции языка



2. ООП в Scala. Pattern matching. Функциональные конструкции. Adt

3. Библиотека коллекций в Scala



4. Асинхронные операции, обработка исключений, неявные параметры

5. Параметрический полиморфизм. Имплиситы



6. Основы функционального программирования. Часть 1

7. Основы функционального программирования. Часть 2

8. ZIO, TF и основной ФП-стэк в Scala

9. Функциональные стримы на примере fs2, работа с бд

10. Акторы в Scala



traits
classes
objects



Traits

- конкретные и абстрактные методы
- нет параметров конструктора
- не можем создавать экземпляры
- поддерживают множественное подмешивание
- можно подмешивать при создании экземпляра



Classes and Objects

- классы являются шаблонами для создания объектов
- классы могут иметь поля, методы, конструкторы
- классы могут наследовать другие классы
- экземпляры классов называются объектами



case Classes and Objects

- иммутабельная структура данных
- объект companion
- copy
- hashCode
- equals
- toString

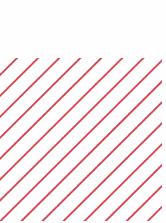


recursion



Recursion

В функциональных языках нет изменяемых переменных,
поэтому вместо традиционных циклов while, for
используется рекурсия



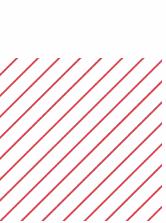
Tail recursion

Хвостовая рекурсия — частный случай рекурсии, при котором любой рекурсивный вызов является последней операцией перед возвратом из функции.

Оптимизация хвостовой рекурсии путём преобразования её в плоскую итерацию реализована во многих оптимизирующих компиляторах. В некоторых функциональных языках программирования спецификация гарантирует обязательную оптимизацию хвостовой рекурсии.



pattern matching



Pattern matching

Pattern matching - aka switch на стероидах

- Типы
- Структурный / структурный вложенный
- Литералы
- Константы
- Гарды
- “As” паттерн
- Альтернативы



algebraic data types



Algebraic data types

- Product types (произведения)
- Sum types (суммы)



Algebraic data types

Произведение типов $A * B$ - это такой тип, который позволит закодировать все возможные комбинации типов A и B

Boolean * Unit

Boolean * Int



Algebraic data types

Сумма типов A + B - это такой тип, который позволит
закодировать все значения типа A и все значения типа B

Boolean + Unit

Boolean + Int



pure functions



Pure functions

Чистая функция определяется исключительно входящими параметрами и должна возвращать только определенный результат без побочных эффектов

Функции возвращающие Unit по определению не могут быть чистыми



Итоги

Спасибо за внимание