Перечисляемые типы

- Enum - ограниченных набор именованных значений, работать с которыми можно как с обычными компонентами программы
- Метод values - список констант перечисления, следующий в порядке объявления -> можно foreach
- При создании компилятор генерирует соответствующий класс, который наследует от класса Enum полезные возможности базового класса
- Ordinal() - возвращает значение int определяющее порядок объявления экземпляров перечисления
- Реализует Comparable -> есть метод compareTo()
- Реализует интерфейс Serializable
- Вызывая getDeclaringClass()
- Name – имя в таком виде, в каком оно было оъявлено
- ValueOf() - экземпляр перечисления, соответсвующий переданому имени
- Для использование констант необходимо статистически экспортировать перечисления
- Перечисление не может использоваться при наследовании -> можно добавлять новые методы
- Перечисление может содержать метод main

- Для сохранение расширенной информации - конструктор на уровне пакета
- Конструктор мало на что влияет, тк может создавать экземпляры, обявленные в определении перечисления
- Переопределение не отличается от переопр в обычном классе
- switch работает с перечислениями
- При отсутсвии возвр значений компилятор не жалуется на отсутствие default
- Все классы перечислений создаются за вас и расширяют класс Enum
- Метода values нет в основном классе enum
- values - статический метод, добавляемый компилятором
- Имеется статич. секция инициализации, которая, может переопределяться
- Из-за стирания декомпилятор не располагает полной инф об enum -> в качестве базового класса
- Explore указывается обобщение Enum
- values - статический метод, вставленный в определение перечисления компилятором.
- Перечисления наследуют enum, множественного наследования нет -> перечисление не может бвть создано наследованием
- Но можно реализовать интерфейсы
- Для вызова метода необходимо иметь экземпляр enum

- Случайный выбор
- Во многих примерах этой главы используется случайный выбор из экземпляров перчисления
- Задачу можно перевести на более общий уровень с использованием обобщений
- Категории можно реализовать группировкой элементов внутри интерфейса и созданием
- перчисления но основе этого интерфейса.
- Если вы хотите создать перчисления
- Категории реализовываются группировкой элементов внутри интерфейса и созлланием перечисление на основе интерфейса

- Единственной формой создание подтипов для перечислений - реализацияя вложенного перечисления
- Если вы хотите создать перечисление перечислений, создайте окружающее перечисление
- Вложение перечислений в перечисления для создание подкатегорий
- Результат - реорганизация кода

- EnumSet вместо флагов
- Set - разновидность коллекции, в которой запрещены повтор объекты
- Перечисление требует, чтобе все его члены были уникальны - на первый взгляд множество
- Объединение в виде EnumSet, замена битовых флагов на базе int
- Наборы флагов используются для передачи информации с состояниями вкл.выкл
- Приходится работать с физическими битами
- Главная цель была высокая скорость
- В реализации - одно значение Long, интерпретируемый как битовый вектор
- Все операции исключительно быстро и эффективно
- Один из признаков ориентированности EnumSet на производительность - пергрузка метода of()
- Команда - начинается с интерфейса содержащего один метод - создает несколько реализаций с разным поведением этого метода
- EnumMap могут использоваться для диспетчеризации в ситуациях с несколькими перечислениями

- Методы Констант
- С разными классами - разное поведение
- Но экземпляр перечисления - не может быть использован как тип класса
- Переопределение методов констант вместо реализации абстрактного метода?

- Цепочка обязанностей
- Разработчик создает несколько способов решения нескольких задач и связывает их в цепочку
- Конечный автоматы
- Возможны переходные состояния - автомат выходит после завершения их задач
- С двумя Input связана сумма
- Выбор между экземплярами перечисления реализуется switch
- Множественная диспетчеризация
- Используется одиночная диспетчеризация
- Item - интейрфейс для типов, которые будут задействованы во множественной диспетчеризации
- Диспетчеризация с использованием перечислений не работают - не могут

- Экземпляры перечисления не могут использоваться в качестве типов аргументов в сигнатурах методов
- Лучшее - использовать switch
- Вторая диспетчериация выполняется compete() с двумя аргументами
- EnumMap
- Двойную диспетчеризацию (переключение типов) можно EnumMap
- EnumMap инициализируется в секции static
- Табличная структура на примере вызова initRow()
- compete() - обе диспетчеризации в одной команде

- Использование двумерного массива
- Каждый экземпляр перечисления - фиксированное значение может быть получено вызовом ordinal()
- Массив отображающий Competitor на Outcome - самое компактное и прямолинейное решение