Подготовка к сдаче лабы

# Исключения

Исключение – возникновение ошибки или непредвиденной ситуации. Для того, чтобы программа знала, как поступать придумали обработку исключений.

Блок **try –** определяет блок, в котором может произойти исключение (возникнуть ошибка)

Блок **catch –** определяет блок кода, в котором происходит отлов определённого исключения (их может быть несколько)

Блок **finaly** **–** определяет блок кода, который однозначно должен быть выполнен (является необязательным, то есть может существовать только конструкция try-catch)

**Throw** – используется для вброса исключений

**Throws –** предупреждает о том, что метод может (должен) выбросить исключение (прописывается в сигнатуре)

# Типы исключений

# 

Throwable – самый базовый класс всех исключений. В этом классе содержится код, который записывает текущий стрек-трейс вызовов функции в массив. Очевидно, что в оператор throw можно передать только наследника класса Throwable. Теоретически можно написать throw new Throwable(), но так никто не делает, ибо зачем оно как бы надо.

**Трек-трейс** – Например, в любой момент работы программы можно узнать класс и имя метода, который сейчас выполняется. И даже не одного метода, а получить информацию о всей цепочке вызовов методов от текущего метода до метода main().

Класс **Error** – прямой наследник класса Throwable. Объекты типа Error (и его классов наследников) создаются в случае каких-то серьёзных проблем непосредственно в java-машине. Например, сбой в работе или нехватка памяти. Зачастую программист ничего не может сделать если случилась такая ошибка.

Класс **Exception** (и **RuntimeException**) – это обычные ошибки, которые возникают во время работы многих методов. Цель каждого выброшенного исключения – быть захваченным блоком catch.  
Другими словами, если какая-то переменная оказалась равна null, метод выкинет **NullPointerException**, если в метод передали неверные аргументы — выкинет **InvalidArgumentException**, если в методе случайно было деление на ноль — **ArithmeticException**.

Класс **RuntimeException –** это разновидность (подмножество) исключений **Exception**. Можно даже сказать, что этот класс это версия обычных исключений: на эти исключения накладывается меньше требований и ограничений.

Непроверяемые исключения (unchecked) наследуются от классов **Error и RuntimeException**, все остальные исключения являются проверяемыми (checked).

# Требования для выброса исключений

1. Если метод выбрасывает проверяемое исключение, то в своей сигнатуре он должен содержать информацию об этом

public PhoneNumber(String phoneNumber) throws InvalidPhoneNumberException

Непроверяемые исключения же могут работать без такого объявления.

1. Если был вызван метод, которые содержит проверяемое исключение в своей сигнатуре, то это нельзя оставить просто так, компилятор просто не даст вам этого сделать. Вы должны проверить данное исключение блоком try-catch или try-catch-finaly. Но это не всегда нужно делать. Можно пробросить исключение выше, по методам. Можно обработать только одно исключение, а второе пробросить дальше. Ну или попросту перехватить все исключения.