Java Core. Продвинутый уровень

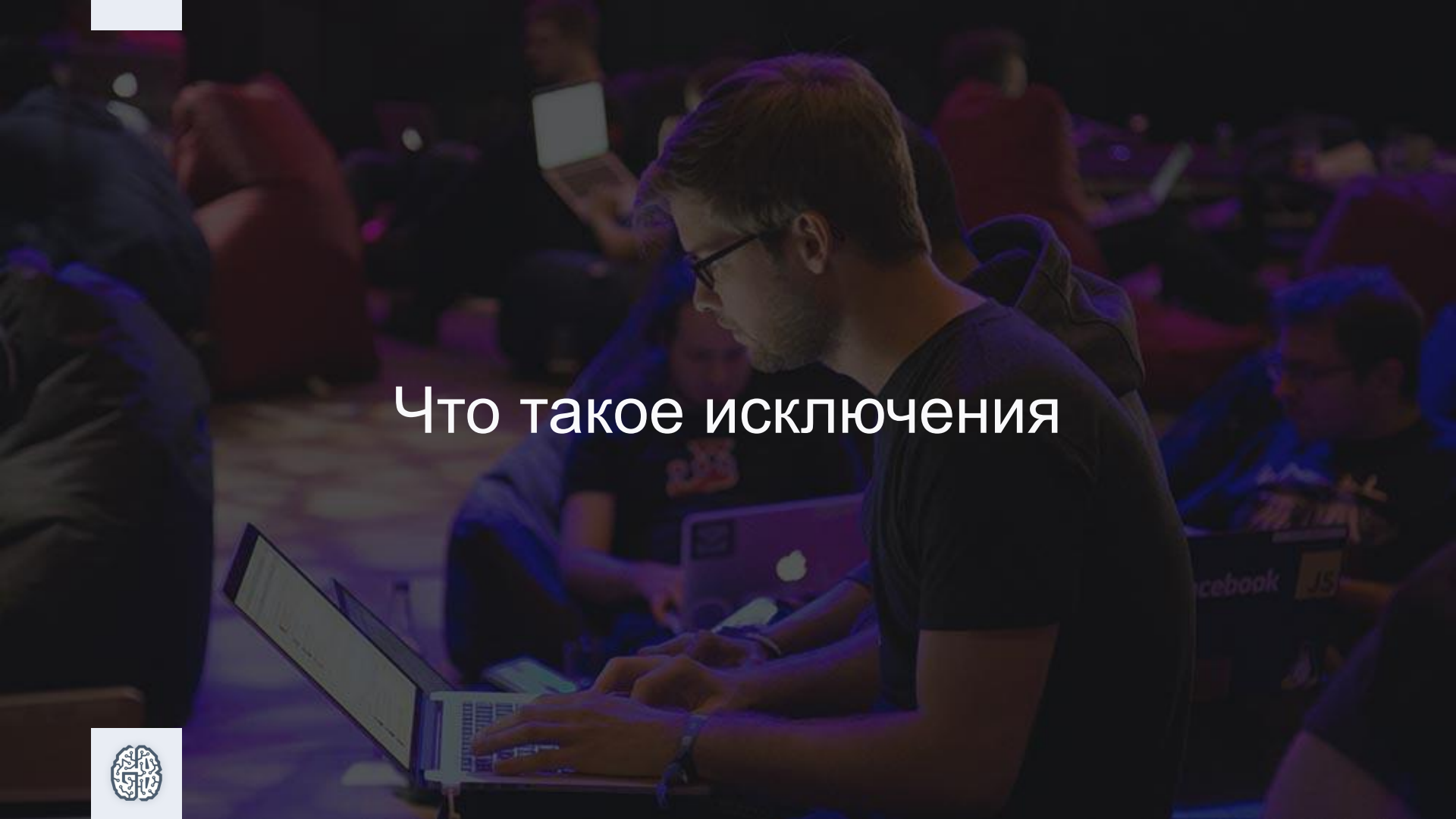
Урок 2

Исключения

Концепция обработки исключений, её сравнение с традиционным механизмом обработки ошибок

План урока

- Что такое исключения?
- Иерархия исключений
- Перехват исключений
- Особенности работы с блоками try-catch
- checked и unchecked исключения
- Ручной выброс исключений
- Проброс исключений
- Оператор finally


Что такое исключения

Что такое исключения

Исключения в Java представляют собой объекты, генерируемые во время появления ошибочных ситуаций и содержащие информацию о них. В зависимости от типа ошибки, будет меняться и класс генерируемого исключения.

Тип исключения	Описание
ArithmeticException	Арифметическая ошибка
ArrayIndexOutOfBoundsException	Выход индекса за пределы массива
ClassCastException	Неверное приведение типов
IllegalArgumentException	Употребление недопустимого аргумента при вызове метода
IndexOutOfBoundsException	Выход индекса некоторого типа за допустимые пределы
NegativeArraySizeException	Создание массива отрицательного размера
NullPointerException	Неверное использование пустой ссылки
NumberFormatException	Неверное преобразование символьной строки в числовой формат

Пример

При выполнении кода, представленного ниже, будет сгенерировано исключение ArithmeticException

```
public class ExceptionApp {
    public static void main(String[] args) {
        int a = 0;
        int b = 10 / a;
    }
}
```

Вывод в консоль

```
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at MainClass.main(MainClass.java:4)
```

Что означает вывод в консоль

```
1                                public class ExceptionApp {
2                                    public static void justMethod() {
3                                        int a = 0;
4                                        int b = 10 / a;
5                                    }
6
7                                    public static void main(String[] args) {
8                                        justMethod();
9                                    }
10                               }
```

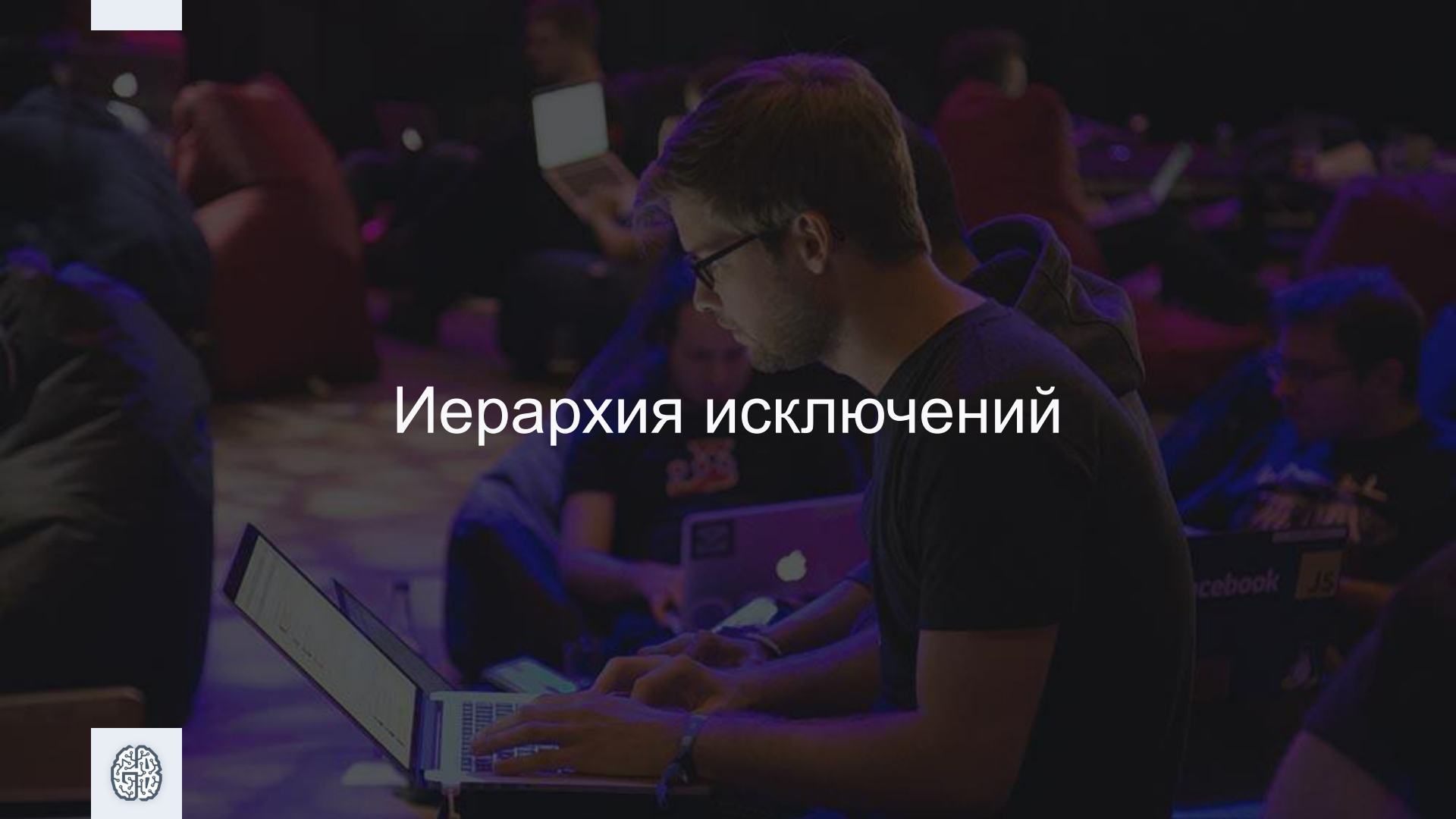
Имя потока, в котором возникло исключение

```
Exception in thread "main" java.lang.ArithmeticException: / by zero
    at ExceptionApp.justMethod(MainClass.java:4)
    at ExceptionApp.main(MainClass.java:8)
```

Распечатка стека, чтобы мы знали что привело нас к появлению исключения

Класс, метод и номер строки, где возникло исключение

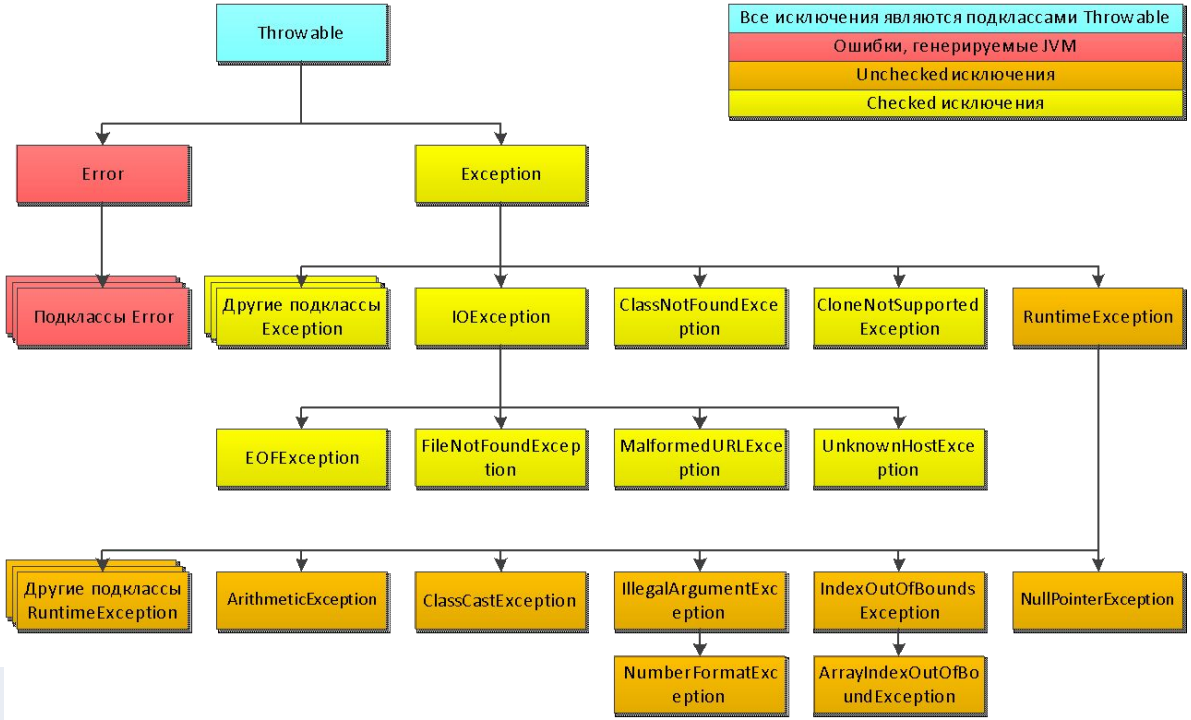
Класс исключения (ArithmeticException), сообщающий о том, что возникла ошибка при выполнении арифметической операции. После двоеточия идет сообщение, уточняющее, что именно произошло

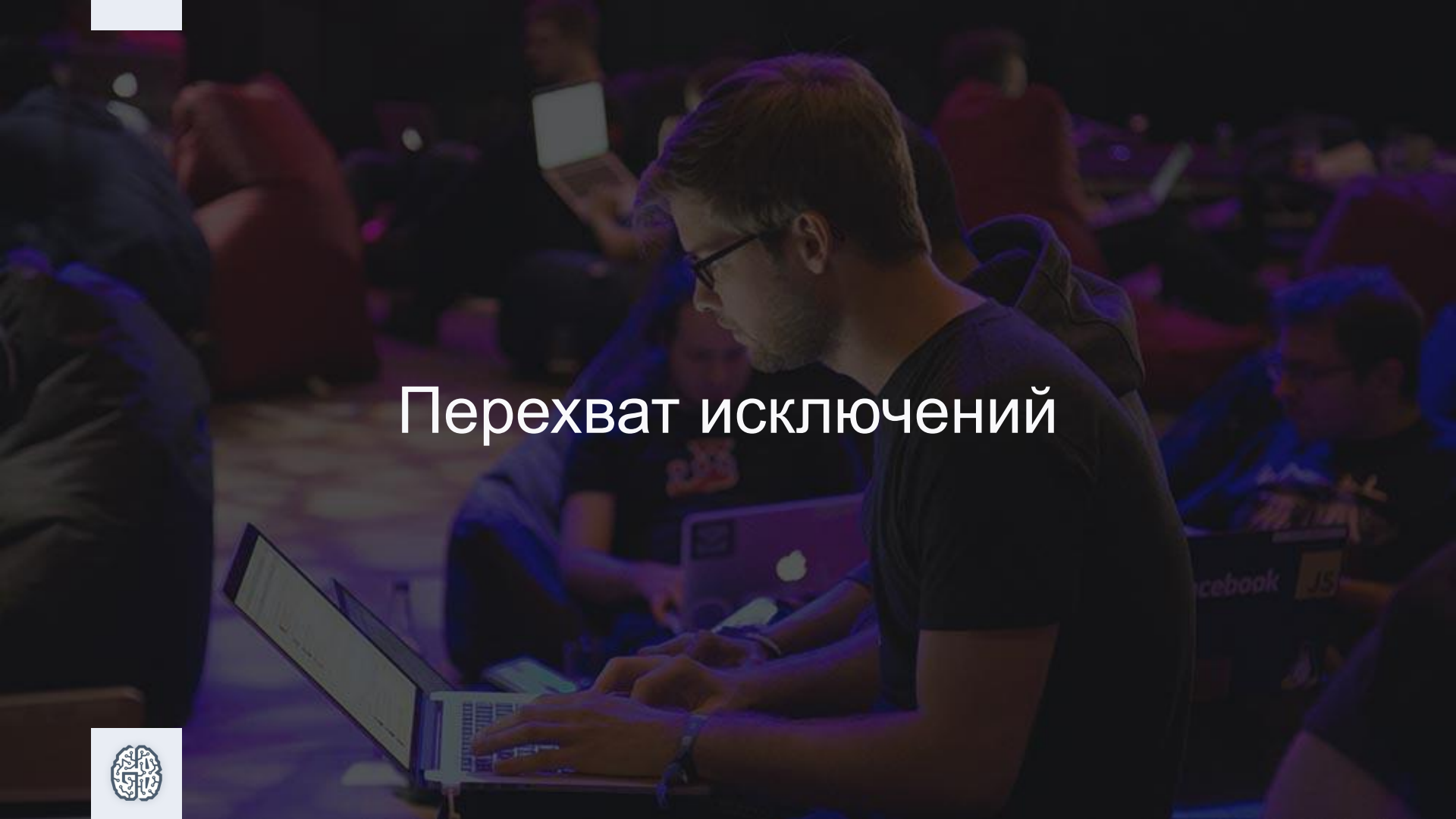

Иерархия исключений

Все исключения можно разделить на три группы:

1. Класс **Exception** и его подклассы
2. Класс **RuntimeException** и его подклассы
3. Класс **Error** и его подклассы

Иерархия исключений





Перехват исключений

Как и зачем перехватывать исключения

```
public static void main(String[] args) {
    int a, b;
    a = 0;
    b = 10 / a;
    System.out.println("Это сообщение не будет выведено в консоль");
}
```

При делении на ноль, Java сгенерирует и бросит ArithmeticException. Если мы не перехватим его, оно долетит то стандартного обработчика исключение Java, и работа нашего приложения завершится, а в консоль будет выведена информация об исключении.

```
public static void main(String[] args) {
    int a, b;
    try {
        a = 0;
        b = 10 / a;
        System.out.println("Это сообщение не будет выведено в консоль");
    } catch (ArithmeticException e) {
        System.out.println("Деление на ноль");
    }
    System.out.println("Завершение работы");
}
```

Результат:

```
Деление на ноль
Завершение работы
```

ArithmeticException e и есть сгенерированное Java и перехваченное нами исключение, то есть исключение это самый обычный объект.

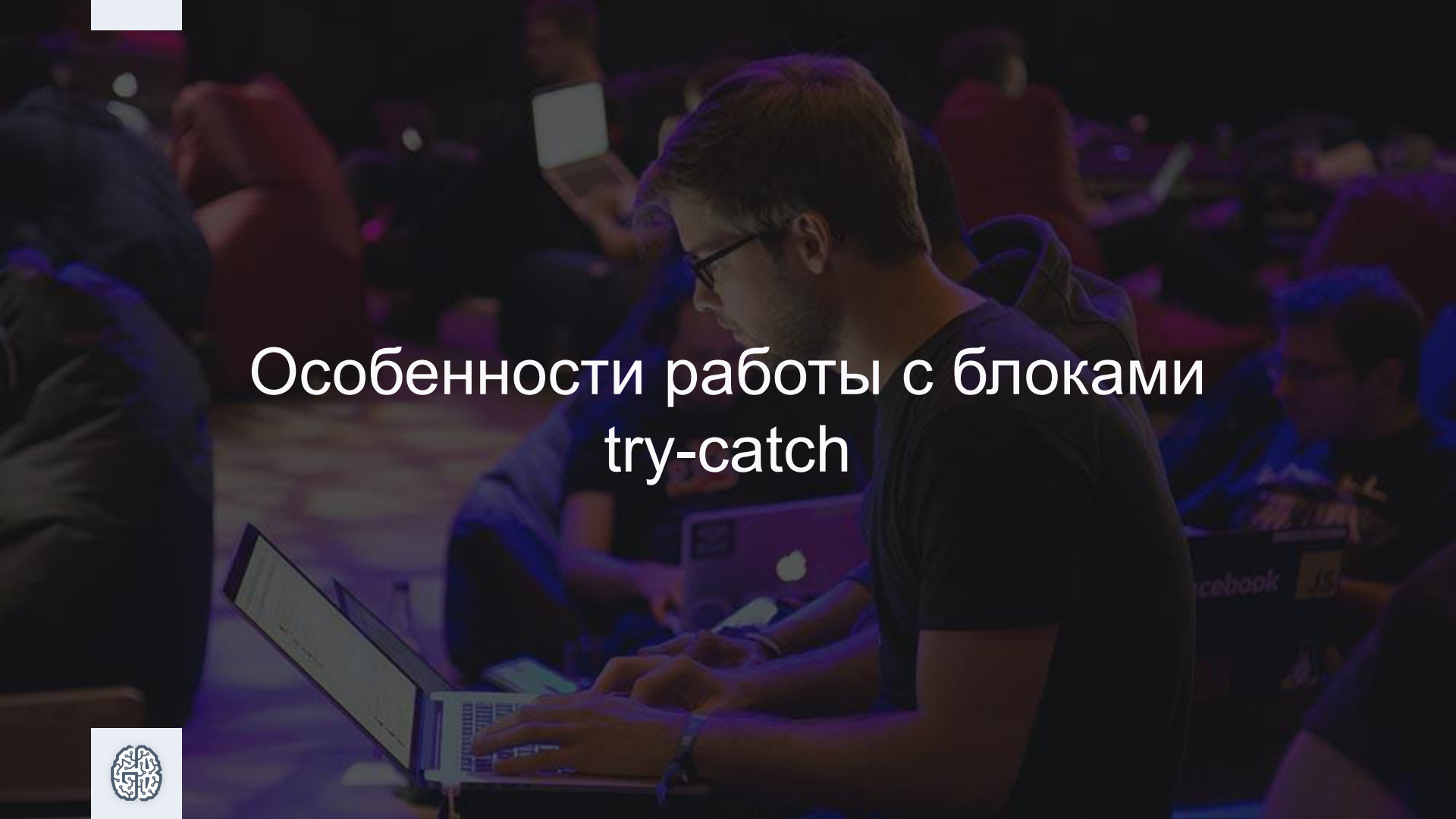
В блоке try, в строке b = 10 / a; будет сгенерировано и брошено исключение, но мы перехватим его блоком catch, отпечатаем в консоль “Деление на ноль”, и программа будет выполняться дальше

```
public static void main(String args[]) {
        System.out.println("Начало");
        try {
            int a = 0;
            int b = 42 / a;
        } catch (ArithmeticException e) {
            e.printStackTrace();
        }
        System.out.println("Конец");
}
```

Результат:

```
Начало
java.lang.ArithmeticException: / by zero
      at MainClass.main(MainClass.java:7)
Конец
```

Если есть необходимость выводить информацию об исключении в случае его перехвата, то можно воспользоваться методом e.printStackTrace()


Особенности работы с блоками
try-catch

Несколько блоков catch для одного try

```
public static void main(String args[]) {
    try {
        int a = 10;
        a -= 10;
        int b = 42 / a;
        int[] c = {1, 2, 3};
        c[42] = 99;
    } catch (ArithmeticException e) {
        System.out.println("Деление на ноль: " + e);
    } catch (ArrayIndexOutOfBoundsException e){
        System.out.println("Ошибка индексации массива: " + e);
    }
    System.out.println("После блока операторов try/catch");
}
```

Сгенерированное исключение будет последовательно проверяться блоками catch сверху-вниз. Если один из catch сработал, то следующие блоки catch не задействуются.

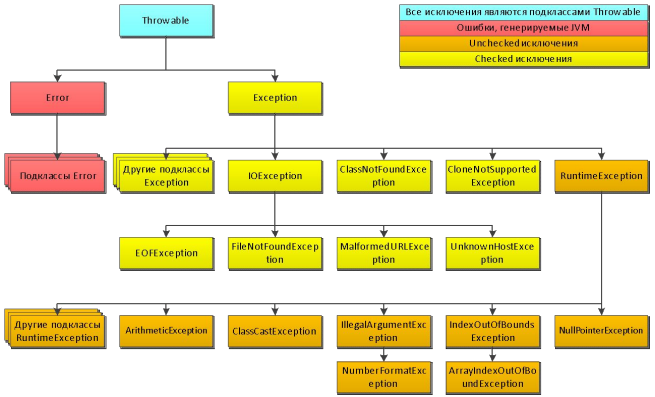
Порядок блоков catch

```
public static void main(String args[]) {
    try {
        int a = 0;
        int b = 42 / a;
    } catch (Exception e) {
        System.out.println("Exception");
    } catch (ArithmeticException e) { // ОШИБКА : недостижимый код !
        System.out.println("Этот код недостижим");
    }
}
```

При перехвате исключения определенного типа, мы перехватываем и исключения его подтипов. Поэтому блоки catch с суперклассами, должны стоять ниже блоков catch с их подклассами, в противном мы получим ошибку о недостижимом коде

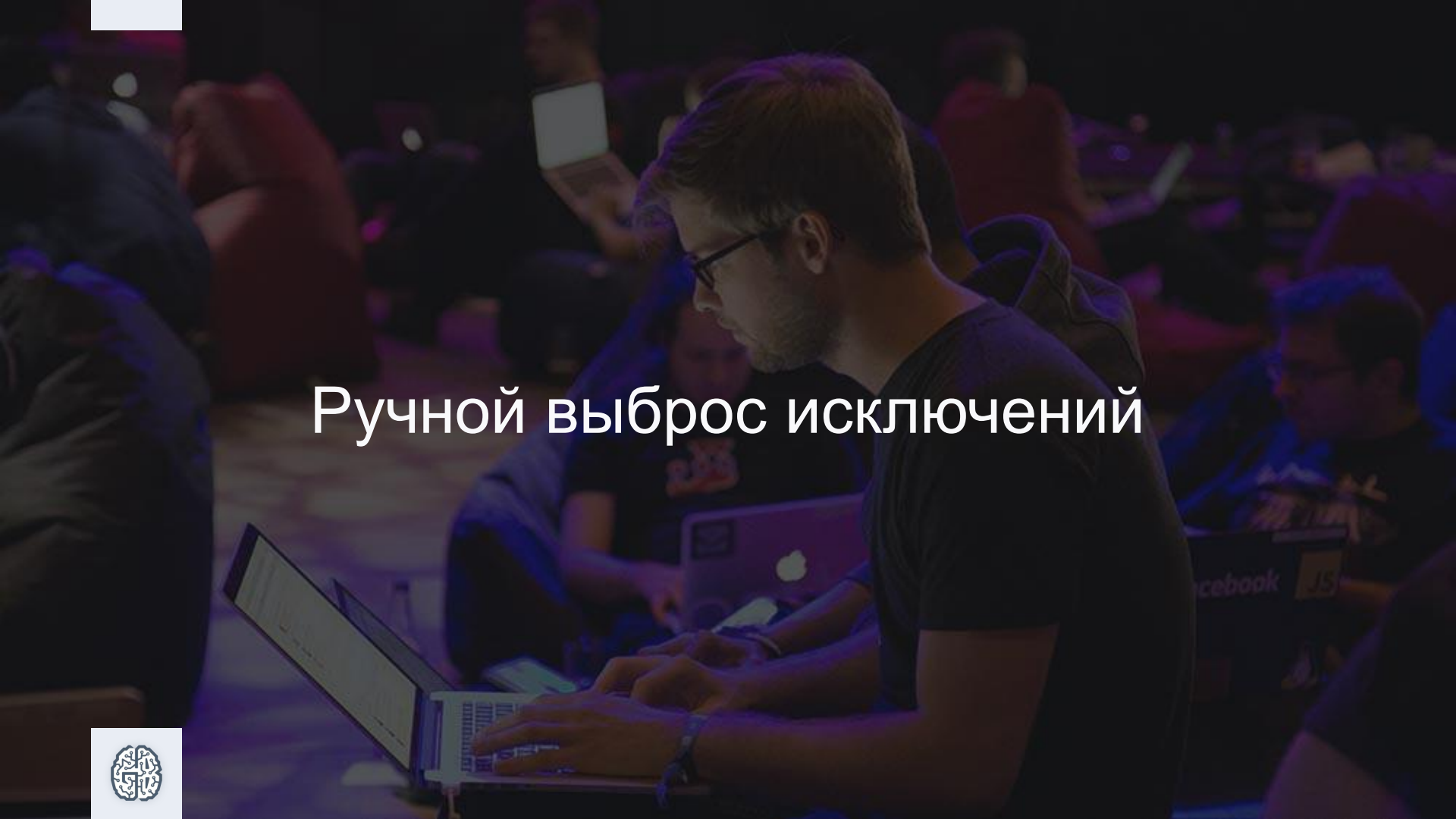
checked и unchecked исключения

checked и unchecked исключения



Все исключения делятся на две группы: checked и unchecked. Checked исключения должны быть обязательно обработаны через блоки try-catch, либо проброшены наверх. Обработку unchecked исключений компилятор не отслеживает

Вид исключения, зависит от вида его родительского класса

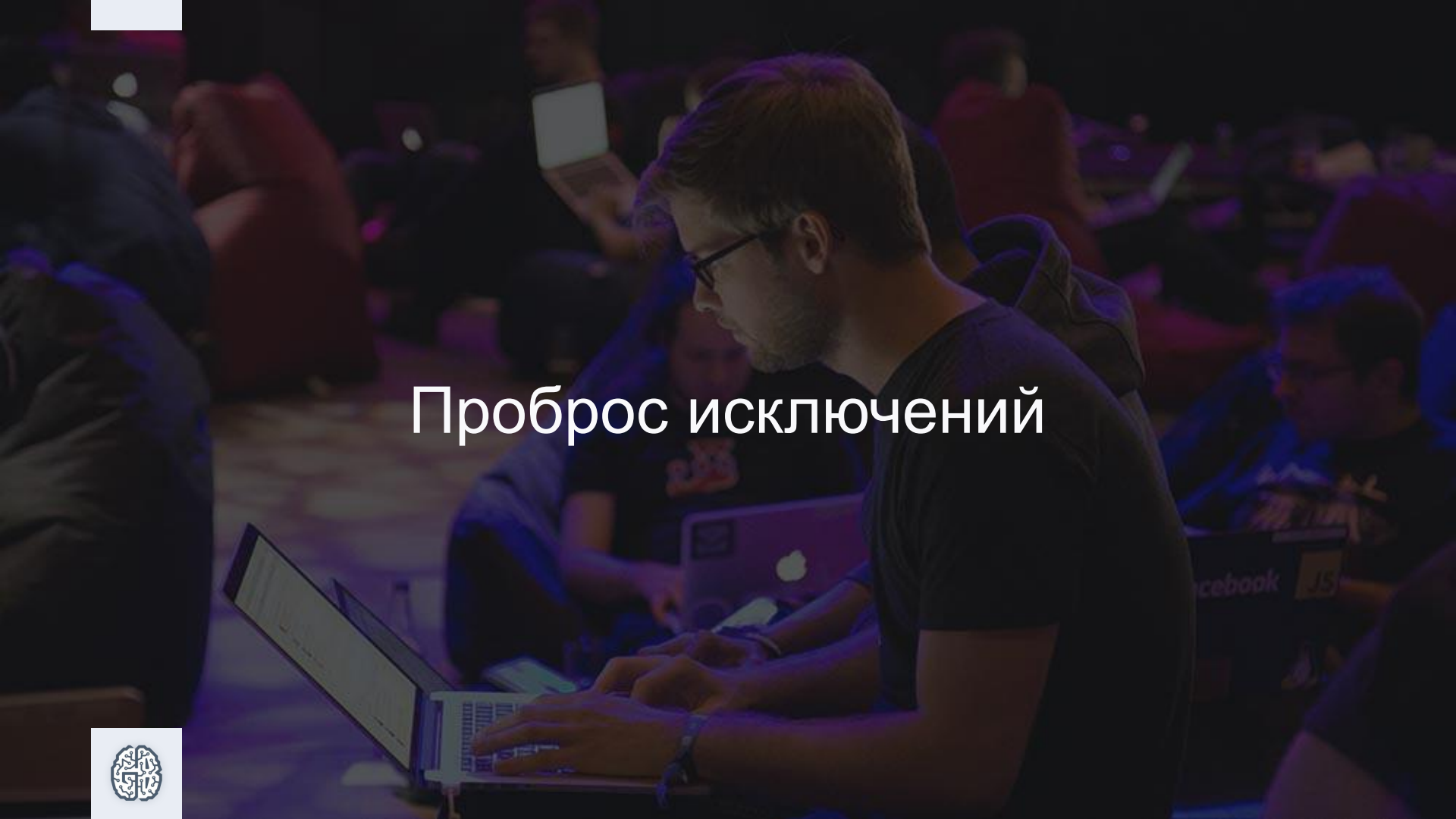


Ручной выброс исключений

Ручной выброс исключений

```
public double sqrt(double in) {
    if (in < 0.0) {
        throw new ArithmeticException("Нельзя вычислить корень квадратный из
отрицательного числа");
    }
    return Math.sqrt(in);
}
```

Если при работе наших методов, есть ситуации, при которых корректное выполнение метода невозможно, то существует возможность вручную сгенерировать и бросить исключение


Проброс исключений

Проброс исключений

```
public static void main(String args[]) {
    try {
        createReport();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

public static void createReport() throws IOException {
    PrintWriter pw = new PrintWriter("report.txt");
    pw.close();
}
```

В некоторых случаях, в методе невозможно корректно обработать то или иное исключение, и тогда его можно пробросить “наверх” с помощью ключевого слова throws

Оператор finally

Оператор finally

```
try {
   // блок кода, в котором отслеживаются исключения
} catch (ТипИсключения1 e1) {
   // обработчик исключения тип_исключения_1
} catch (ТипИсключения2 e2) {
   // обработчик исключения тип_исключения_2
} finally {
   // блок кода, который обязательно выполнится по завершении блока try
}
```

Блок finally срабатывает всегда, вне зависимости от того, было ли сгенерировано или перехвачено исключение, или нет

Практическое задание

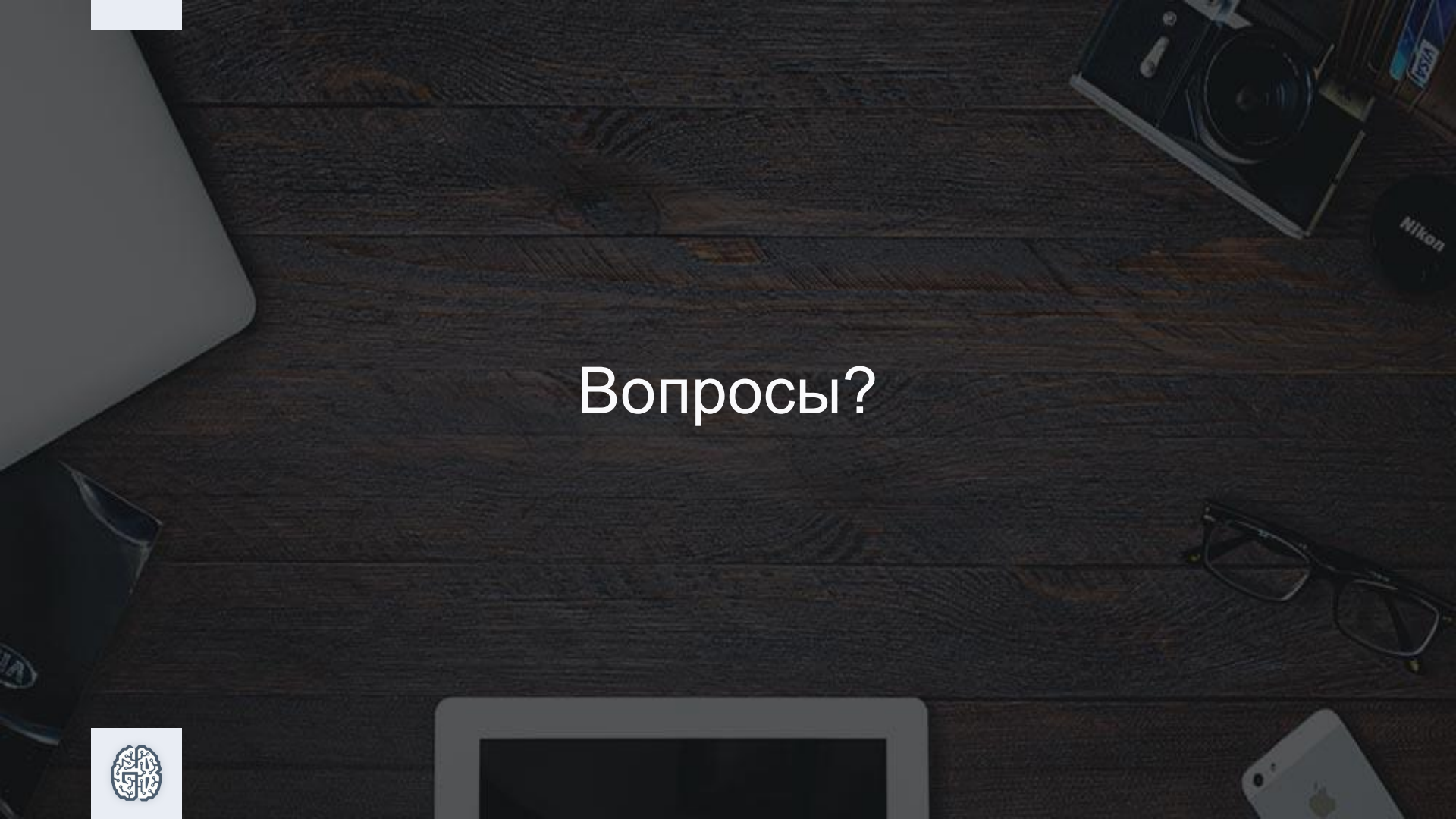
1. Напишите метод, на вход которого подаётся двумерный строковый массив размером 4x4. При подаче массива другого размера необходимо бросить исключение MyArraySizeException.

Практическое задание

2. Далее метод должен пройтись по всем элементам массива, преобразовать в int и просуммировать. Если в каком-то элементе массива преобразование не удалось (например, в ячейке лежит символ или текст вместо числа), должно быть брошено исключение MyArrayDataException с детализацией, в какой именно ячейке лежат неверные данные.

Практическое задание

3. В методе main() вызвать полученный метод, обработать возможные исключения MySizeArrayException и MyArrayDataException и вывести результат расчета.


Вопросы?