**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Информационные технологии»

**Отчет по лабораторной работе №4**

по дисциплине «Информационные технологии и программирование» на тему:

Работа с исключениями в Java

Выполнил:

студент группы БВТ2302

Маслов Максим Станиславович

Проверил:

Харрасов К. Р.

Москва 2024

**Цель работы**

Ознакомиться с теорией, касающейся исключений в java, и выполнить задания.

**Теория**

Исключения в Java являются механизмом обработки ошибок и нестандартных ситуаций во время выполнения программы. Они позволяют изолировать ошибки и обрабатывать их в специальном блоке кода, что делает программу более надежной и устойчивой к ошибкам.

В Java есть множество типов исключений, которые могут возникнуть в различных ситуациях. Например, классы IOException и FileNotFoundException используются для обработки ошибок вводавывода, а классы NullPointerException и ArrayIndexOutOfBoundsException используются для обработки ошибок, связанных с работой с объектами и массивами.

Кроме того, в Java можно создавать свои собственные исключения, наследуясь от класса Exception или его подклассов. Это позволяет создавать более гибкую и точечную обработку ошибок в своих программах.

При работе с исключениями важно учитывать, что блок try-catch не должен заменять проверку условий и предотвращение возможных ошибок на этапе написания кода. Он должен использоваться только для обработки неожиданных ошибок, которые могут возникнуть в процессе выполнения программы. Код, который может вызвать исключение, помещается в блок try. Если исключение возникает внутри блока try, то управление передается в блок catch, который обрабатывает исключение. Блок catch может содержать несколько разделов, каждый из которых обрабатывает определенный тип исключения.

try { // Код, который может вызвать исключение }

catch (ExceptionType1 e) { // Обработка исключения типа ExceptionType1 }

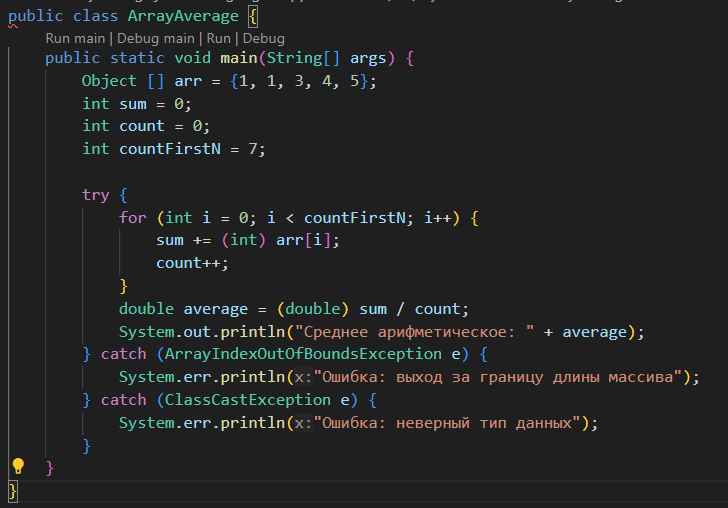
catch (ExceptionType2 e) { // Обработка исключения типа ExceptionType2 }

finally { // Код, который выполняется всегда, независимо от того, возникло ли исключение или нет }

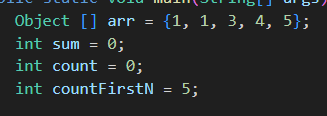
* Задание 1: Необходимо написать программу, которая будет находить среднее арифметическое элементов массива. При этом программа должна обрабатывать ошибки, связанные с выходом за границы массива и неверными данными (например, если элемент массива не является числом).
* Задание 2: Необходимо написать программу, которая будет копировать содержимое одного файла в другой. При этом программа должна обрабатывать возможные ошибки, связанные с открытием и закрытием файлов.
* Задание 3: Создайте класс CustomNumberFormatException, который будет использоваться для обработки исключения NumberFormatException. Реализуйте программу, которая пытается преобразовать строку в число (Integer.parseInt()), и, если строка не является числом, выбрасывайте исключение CustomNumberFormatException

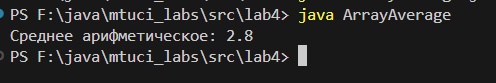
**Ход работы**

Создадим класс ArrayAverage, который будет выводить среднее арифметическое для массива чисел. При этом будем обрабатывать ошибки, касающиеся выхода индекса за пределы массива и передачи неверного типа данных. В блоке try-catch, мы сначала суммируем все числа из массива объектов и находим его среднее арифметическое. Далее при возникновении проблемы мы ловим ошибку, а именно выход за пределы массива и ввод неверных данных массива.



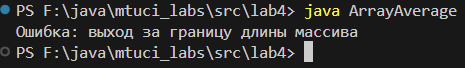
Посмотрим на корректное выполнение кода со следующими параметрами:





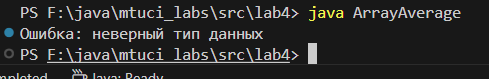
Далее проверим выход за пределы массива, изменив следующую строчку:





Теперь проверим ошибку неверного типа данных:





Приступим к выполнению второго задания:

Во 2 задании Java-код предназначен для копирования содержимого одного файла в другой. Сначала он задает названия исходного и целевого файлов. Исходный файл — это тот, из которого считываются данные, а целевой — тот, в который данные записываются. Затем объявляются две переменные, которые будут использоваться для работы с файлами: одна для чтения данных из исходного файла и другая для записи данных в целевой файл.

Вся операция по копированию обернута в блок `try`, что позволяет перехватывать возможные ошибки, возникающие при работе с файлами. Программа открывает исходный файл для чтения и целевой файл для записи. Если один из файлов невозможно открыть (например, если исходный файл отсутствует или невозможно создать целевой), выбрасывается исключение.

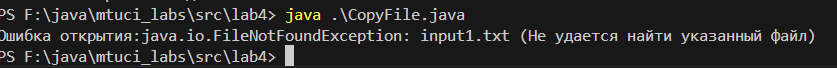
Если в процессе копирования возникает ошибка, `catch`-блок перехватывает её и выводит сообщение о возникшей проблеме. Это позволяет программе сообщить пользователю о том, что во время копирования произошла ошибка, и указать её причину.



Посмотрим на вывод корректного выполнения:

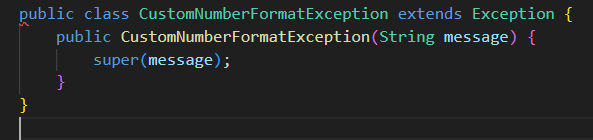


Если попытаться открыть несуществующий файл, то будет ошибка открытия:

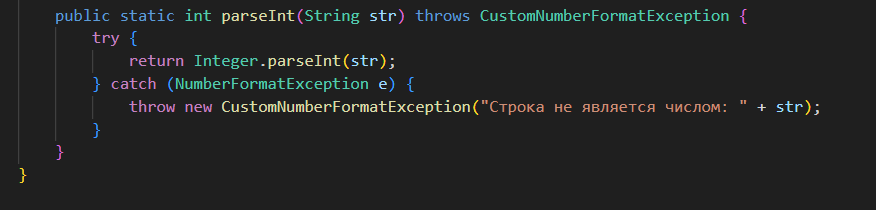


Далее приступим к выполнению 3 задания.

Создадим класс для нашей кастомной ошибки:



Далее создадим метод для преобразования строки в число, который должен быть в пределах.



Для обоих исключений предусмотрен вызов метода `logException`. Этот метод создаёт или открывает файл `log.txt` и записывает в него сообщение об ошибке. Он проверяет, существует ли файл `log.txt`, а если его нет, создает его. Используется `FileWriter` с параметром `true`, чтобы добавить новое сообщение об ошибке в конец файла, не перезаписывая его.

Если метод `logError` не может записать сообщение в файл из-за ошибки ввода-вывода (например, если файл не может быть создан или открыт), он перехватывает исключение `IOException` и выводит в консоль сообщение о том, что запись в лог не удалась.

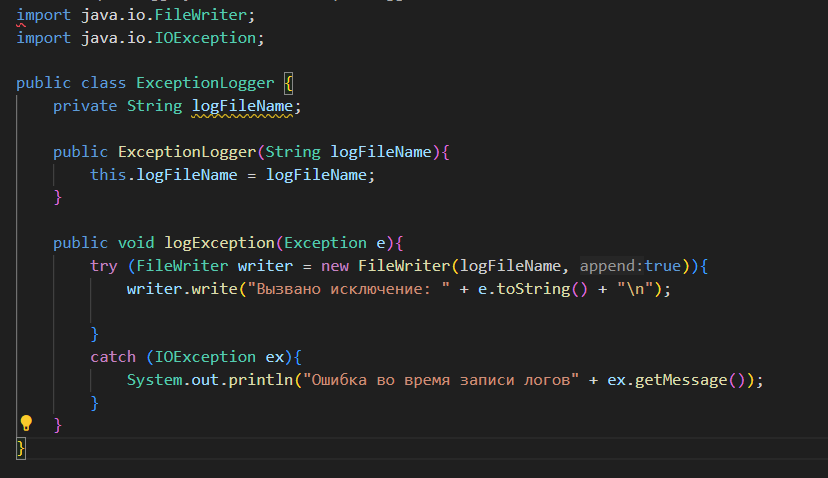
Таким образом, программа выполняет следующие действия:

1. Читает аргументы командной строки и пытается преобразовать их в числа.

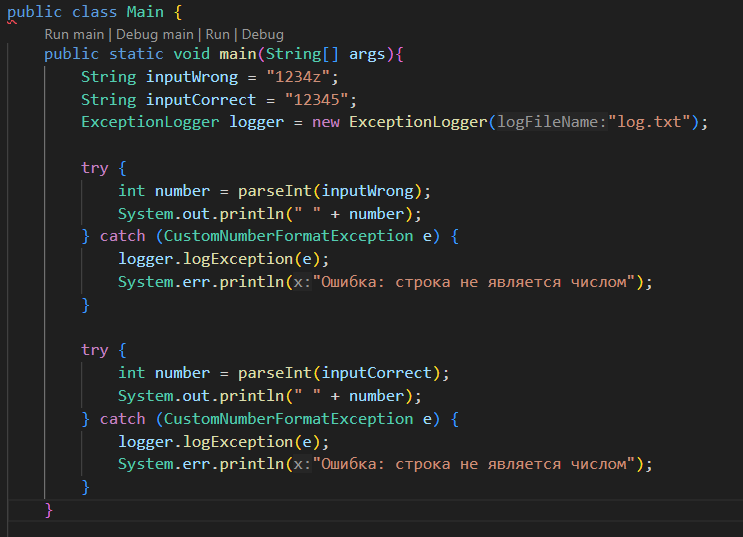
2. Выполняет преобразование строки к числу с обработкой возможных исключений.

3. Если возникает ошибка, выводит сообщение об ошибке и записывает её в лог-файл `log.txt`.

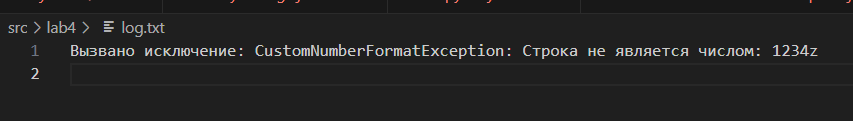
Напишем логгер:

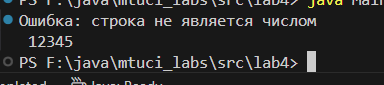


Напишем исполняемый файл для тестирования:



Убедимся в корректности работы нашего кода:





Вывод: изучили теорию по исключениям в java и выполнили задания.