5. Семинар: обработка исключений

5.1. Инструментарий

— Презентация для преподавателя, ведущего семинар;
— Фон GeekBrains для проведения семинара в Zoom;
— JDK любая 11 версии и выше;
— IntelliJ IDEA Community Edition для практики и примеров используется IDEA.

5.2. Цели семинара

— Практика создания, отправки и принятия данных;
— сохранение и загрузка состояния программы между запусками;
— работа с большими текстами (поиск, замена, генерация);
— начало рассмотрения популярных пакетов ввода-вывода.

5.3. План-содержание

Что происходит	Время	Слайды	Описание
Организационный момент	5	1-5	Преподаватель ожидает студентов, поддерживает активность и коммуникацию в чате, озвучиает цели и планы на семинар. Важно упомянуть, что выполнение домашних заданий с лекции является, фактически, подготовкой к семинару
Quiz	5	6-18	Преподаватель задаёт вопросы викторины, через 30 секунд демонстрирует слайд-подсказку и ожидает ответов (6 вопросов, по минуте на ответ)
Рассмотрение ДЗ лекции	10	19-23	Преподаватель демонстрирует свой вариант решения домашнего задания с лекции, возможно, по предварительному опросу, демонстрирует и разбирает вариант решения одного из студентов
Вопросы и ответы	10	24	Преподаватель ожидает вопросов по теме прошедшей лекции, викторины и продемонстрированной работы
Задание 1	30	25-35	Сквозное задание, состоящее из объяснений в 6 пунктах, обязательных к исполнению. Всё задание выдаётся сразу целиком и является неделимым.
Перерыв (если нужен)	5	36	Преподаватель предлагает студентам перерыв на 5 минут (студенты голосуют)

Что происходит	Время	Слайды	Описание
Задание 2	40	37-49	Сквозное задание, состоящее из объяснений в 6 пунктах, обязательных к исполнению. Всё задание выдаётся сразу целиком и является неделимым.
Домашнее задание	5	50-51	Объясните домашнее задание, подведите итоги урока
Рефлексия	10	52-53	Преподаватель запрашивает обратную связь
Длительность	120		

5.4. Подробности

5.4.1. Организационный момент

- **Цель этапа:** Позитивно начать урок, создать комфортную среду для обучения.
- **Тайминг:** 3-5 минут.
- **Действия преподавателя:**
  - Запрашивает активность от аудитории в чате;
  - Презентует цели курса и семинара;
  - Презентует краткий план семинара и что студент научится делать.

5.4.2. Quiz

- **Цель этапа:** Вовлечение аудитории в обратную связь.
- **Тайминг:** 5-7 минут (4 вопроса, по минуте на ответ).
- **Действия преподавателя:**
  - Преподаватель задаёт вопросы викторины, представленные на слайдах презентации;
  - через 30 секунд демонстрирует слайд-подсказку и ожидает ответов.
- **Вопросы и ответы:**
  1. Что такое GPT? (2)
     (a) General partition trace;
     (b) GUID partition table;
     (c) Greater pass timing.
  2. Строки в языке Java – это: (3)
     (a) классы;
     (b) массивы;
     (c) объекты.
  3. Ссылка на местонахождение – это: (2)
     (a) URI;
     (b) URL;
     (c) URN.
  4. Возможно ли чтение совершенно случайного байта данных из потока, к которому подключен объект `BufferedInputStream`? (2)
     (a) да;

(b) нет;
(c) да, если это поток в программу из локального файла.

5.4.3. Рассмотрение ДЗ

— **Цель этапа:** Пояснить не очевидные моменты в формулировке ДЗ с лекции, синхронизировать прочитанный на лекции материал к началу семинара.
— **Тайминг:** 15-20 минут.
— **Действия преподавателя:**
  – Преподаватель демонстрирует свой вариант решения домашнего задания из лекции;
  – возможно, по предварительному опросу, демонстрирует и разбирает вариант решения одного из студентов.
— **Домашнее задание из лекции:**
  – Создать пару-тройку текстовых файлов. Для упрощения (чтобы не разбираться с кодировками) внутри файлов следует писать текст только латинскими буквами.
  Вариант решения
  Задание не предполагало кода, но будет плюсом, если студенты создали файлы не вручную, а программным кодом. Для этого и последующих заданий понадобится набор констант и вспомогательный массив с названиями файлов

```
private static final Random rnd = new Random();
private static final int CHAR_BOUND_L = 65;
private static final int CHAR_BOUND_H = 90;
private static final int FILES_AMOUNT = 10;
private static final int WORDS_AMOUNT = 5;
private static final int WORD_LENGTH = 10;
private static final String WORD_TO_SEARCH = "geekbrains";

String[] fileNames = new String[FILES_AMOUNT];
for (int i = 0; i < fileNames.length; i++)
   fileNames[i] = "file_" + i + ".txt";
```

Файл записывается простым выходным потоком, в который (в приведённом ниже примере) записывается строка, сгенерированная методом. Из строки выделяется массив байтов методом `getBytes()`.

```
private static String generateSymbols(int amount) {
   StringBuilder sequence = new StringBuilder();
   for (int i = 0; i < amount; i++)
      sequence.append((char) (CHAR_BOUND_L + rnd.nextInt(CHAR_BOUND_H - CHAR_BOUND_L)));
   return sequence.toString();
}

private static void writeFileContents(String fileName, int length) throws IOException {
   FileOutputStream fos = new FileOutputStream(fileName);
   fos.write(generateSymbols(length).getBytes());
   fos.flush();
   fos.close();
}
```

Вызов методов обёрнут в `try...catch` и формирует сразу файлы как для второго задания (конкатенации), так и третьего (поиска).

```
try {
```

```
   //#1
   for (int i = 0; i < fileNames.length; i++)
      if (i < 2)
         writeFileContents(fileNames[i], 100);
      else
         writeFileContents(fileNames[i], WORDS_AMOUNT, WORD_LENGTH);
   System.out.println("First task results are in file_0 and file_1.");
}
catch (Exception ex) { throw new RuntimeException(ex); }
```

– Написать метод, осуществляющий конкатенацию (соединение) переданных ей в качестве параметров файлов (не особенно важно, в первый допишется второй или во второй первый, или файлы вовсе объединятся в какой-то третий);

Вариант решения

Метод конкатенации моет также быть написан множеством способов, но важно, чтобы при буферизации (чаще всего решения содержат именно буферизованные варианты) не терялись символы, например, пробелы или переходы на новую строку. Также часто встречается дополнительная буферизация внутри программы, например, в строку, при помощи `StringBuilder`, что будет являться грубой ошибкой при конкатенации больших или бинарных файлов. Поэтому ниже представлен вариант решения с посимвольным чтением и записью в результирующий файл.

```
private static void concatenate(String file_in1, String file_in2, String file_out) throws
    IOException {
   FileOutputStream fos = new FileOutputStream(file_out);
   int ch;
   FileInputStream fis = new FileInputStream(file_in1);
   while ((ch = fis.read()) != -1)
      fos.write(ch);
   fis.close();

   fis = new FileInputStream(file_in2);
   while ((ch = fis.read()) != -1)
      fos.write(ch);
   fis.close();

   fos.flush();
   fos.close();
}
```

Вызов метода можно записать в секцию `try...catch` предыдущего задания или написать новую.

```
try {
   //#2
   concatenate(fileNames[0], fileNames[1], "FILE_OUT.txt");
   System.out.println("Second task result is in FILE_OUT.");
}
catch (Exception ex) { throw new RuntimeException(ex); }
```

– Написать метод поиска слова внутри файла.

Вариант решения

Для поиска слова понадобится файл не со сплошными символами (хотя и в таком алгоритм поиска должен отработать нормально), а со словами, разделёнными пробелами. Для этого можно видоизменить метод генерации файла и напи-

сать перегруженный, который будет по некоторому псевдослучайному условию вставлять в файл через пробел либо случайно сгенерированную последовательность символов, либо слово, заранее определённое, как искомое.

```
private static void writeFileContents(String fileName, int words, int length) throws
    IOException {
   FileOutputStream fos = new FileOutputStream(fileName);
   for (int i = 0; i < words; i++) {
      if(rnd.nextInt(WORDS_AMOUNT) == WORDS_AMOUNT / 2)
         fos.write(WORD_TO_SEARCH.getBytes());
      else
         fos.write(generateSymbols(length).getBytes());
      fos.write(' ');
   }
   fos.flush();
   fos.close();
}
```

Формально, можно выполнить работу следующим образом, используя `Scanner`, сравнивая слова «от пробела до пробела»

```
private static boolean fileScanner(String fileName, String word) throws IOException {
   Scanner sc = new Scanner(new FileInputStream(fileName));
   word = word.toLowerCase(); // \n
   while (sc.hasNext()) {
      String line = sc.nextLine();
      line = line.toLowerCase();
      if (line.contains(word)) return true;
   }
   return false;
}
```

Но более гибким получится алгоритм, ищущий непосредственно последовательность символов, сравнивая их по одному, это позволяет искать не только словосочетания, но и, например, диалоги персонажей в книге. Данная ниже реализация может найти представленную в комментарии комбинацию, когда неверный символ слова является началом верной последовательности.

```
private static boolean isInFile(String fileName, String search) throws IOException {
   FileInputStream fis = new FileInputStream(fileName);
   byte[] searchSequence = search.getBytes();
   int ch;
   int i = 0;        // geekgeekbrains
   while ((ch = fis.read()) != -1) {
      if (ch == searchSequence[i])
         i++;
      else {
         i = 0;
         if (ch == searchSequence[i]) i++;
      }
      if (i == searchSequence.length) {
         fis.close();
         return true;
      }
   }
   fis.close();
   return false;
}
```

Ещё более точной будет реализация, в которой формируется некоторое окно поиска, фактически, буфер, смещающийся посимвольно по читаемому файлу, это

позволяет найти слово «гоголь» в комбинации «гогоголь».

```
private static boolean isInFile(String fileName, String search) throws IOException {
   try (FileInputStream fis = new FileInputStream(fileName)) {
      int ch;
      StringBuilder sb = new StringBuilder();
      while ((ch = fis.read()) != -1 && sb.length() < search.length()) {
         sb.append((char) ch);
      }

      do {
         if (sb.toString().equals(search))
            return true;
         sb.deleteCharAt(0);
         sb.append((char) ch);
      } while ((ch = fis.read()) != -1);
   } catch (IOException e) {
        throw new RuntimeException(e);
   }
   return false;
}
```

5.4.4. Вопросы и ответы

- **Ценность этапа** Задать задание для самостоятельного выполнения между занятиями.
- **Тайминг** 5-15 минут
- **Действия преподавателя**
  - Преподаватель ожидает вопросов по теме прошедшей лекции, викторины и продемонстрированной работы;
  - Если преподаватель затрудняется с ответом, необходимо мягко предложить студенту ответить на его вопрос на следующем семинаре (и не забыть найти ответ на вопрос студента!);
  - Предложить и показать пути самостоятельного поиска студентом ответа на заданный вопрос;
  - Посоветовать литературу на тему заданного вопроса;
  - Дополнительно указать на то, что все сведения для выполнения домашнего задания, прохождения викторины и работы на семинаре были рассмотрены в методическом материале к этому или предыдущим урокам.

5.4.5. Задание 1

- **Ценность этапа** Сохранение состояния приложения между запусками
- **Тайминг** 15-20 мин
- **Действия преподавателя**
  - Выдать задание студентам;
  - Подробно объяснить, что именно требуется от студентов, избегая упоминания конкретных языковых конструкций;
  - Если группа студентов справилась с заданием, а времени осталось более 5 минут, выдавать группе задания «со звёздочкой».
- **Задание**:

– Создать массив из 9 цифр и записать его в файл, используя поток вывода.

Вариант решения

```
int[] ar0 = {1,2,3,4,5,6,7,8,0,8,7,6,5,4,3};
final int DIGIT_BOUND = 48;

FileOutputStream fos = new FileOutputStream("save.out");
fos.write('[');
for (int i = 0; i < ar0.length; i++) {
   fos.write(DIGIT_BOUND + ar0[i]);
   if (i < ar0.length - 1) fos.write(',');
}
fos.write(']');
fos.flush();
fos.close();
```

$*_1$ Удостовериться, что числа записаны не символами, а цифрами, что сократит объём хранения в 8 и более раз (из-за представления цифр в виде ASCII символов). При этом важно помнить о допущениях такого способа записи, поскольку числа нужно как-то отделять друг от друга, а любой символ, например, пробел (32), имеет числовое представление, и внутри файла будет неотличим от числа 20. А любое отрицательное число будет воспринято потоком чтения как конец потока. Для выполнения задания сделать разделителем число 0.

Вариант решения

```
// assuming 0 is divider
int[] ar1 = {1,2,3,4,5,6,7,8,9,8,7,6,5,4,3};

FileOutputStream fos = new FileOutputStream("save0.out");
for (int i = 0; i < ar0.length; i++) {
   fos.write(ar1[i]);
   fos.write(0);
}
fos.flush();
fos.close();
```

$*_2$ Предположить, что числа в исходном массиве имеют диапазон $[0, 3]$, и представляют собой, например, состояния ячеек поля для игры в крестики-нолики, где 0 – это пустое поле, 1 – это поле с крестиком, 2 – это поле с ноликом, 3 – резервное значение. Такое предположение позволит хранить в одном числе типа `int` всё поле 3x3. Записать в файл 9 значений так, чтобы они заняли три байта.

Вариант решения

```
int[] ar2 = {0,1,2,3,0,1,2,3,0};

FileOutputStream fos = new FileOutputStream("save1.out");
for (int b = 0; b < 3; b++) { // write to 3 bytes
   byte wr = 0;
   for (int v = 0; v < 3; v++) { // write by 3 values in each
      wr += (byte) (ar2[3 * b + v] << (v * 2));
   }
   fos.write(wr);
}
fos.flush();
fos.close();
```

5.4.6. Задание 2

- **Ценность этапа** Загрузка состояния приложения при запуске
- **Тайминг** 15-20 мин
- **Действия преподавателя**
  - Выдать задание студентам;
  - Подробно объяснить, что именно требуется от студентов, избегая упоминания конкретных языковых конструкций;
  - Если группа студентов справилась с заданием, а времени осталось более 5 минут, выдавать группе задания «со звёздочкой».
- **Задание**:
  - Создать массив целых чисел и заполнить его информацией из файла, записанного в предыдущем задании.

    Вариант решения

```
int[] ar00 = new int[15];
final int DIGIT_BOUND = 48;

FileInputStream fis = new FileInputStream("save.out");
fis.read(); // '['
for (int i = 0; i < ar00.length; i++) {
   ar00[i] = fis.read() - DIGIT_BOUND;
   fis.read(); // ','
}
fis.close();

System.out.println(Arrays.toString(ar00));
```

  $*_1$ Прочитать из файла с числами, предполагая, что разделитель – это число 0.

    Вариант решения

```
int[] ar10 = new int[15];
// assuming 0 is divider

FileInputStream fis = new FileInputStream("save0.out");
int b;
int i = 0;
while ((b = fis.read()) != -1) {
   if (b != 0) {
      ar10[i++] = b;
   }
}
fis.close();

System.out.println(Arrays.toString(ar10));
```

  $*_2$ Прочитать из файла, полученного в результате выполнения задания $1*_2$.

    Вариант решения

```
int[] ar20 = new int[9];

FileInputStream fis = new FileInputStream("save1.out");
int b;
int i = 0;
while ((b = fis.read()) != -1) {
   for (int v = 0; v < 3; ++v) { // 3 values of four possible
      ar20[i++] = b >> (v * 2) & 0x3;
   }
```

```
}
fis.close();

System.out.println(Arrays.toString(ar20));
```

5.4.7. Домашнее задание

- **Ценность этапа** Задать задание для самостоятельного выполнения между занятиями.
- **Тайминг** 5-10 минут.
- **Действия преподавателя**
  - Пояснить студентам в каком виде выполнять и сдавать задания
  - Уточнить кто будет проверять работы (преподаватель или ревьювер)
  - Объяснить к кому обращаться за помощью и где искать подсказки
  - Объяснить где взять проект заготовки для дз
- **Задания**

5-25 мин Выполнить все задания семинара, если они не были решены, без ограничений по времени;

Все варианты решения приведены в тексте семинара выше

15 мин 1. В класс покупателя добавить перечисление с гендерами, добавить в сотрудника свойство «пол» со значением созданного перечисления. Добавить геттеры, сеттеры.

Листинг 1: Свойства сотрудника

```
enum Genders{MALE, FEMALE};

// ...
Genders gender;

public Employee(String name, String midName, String surName, String phone, String position,
    int salary, int birth, Genders gender) {
   // ...
   this.gender = gender;
}

public Genders getGender() {
   return gender;
}

public void setGender(Genders gender) {
   this.gender = gender;
}
```

20-25 мин 2. Добавить в основную программу перечисление с праздниками (нет праздника, Новый Год, 8 марта, 23 февраля), написать метод, принимающий массив сотрудников, поздравляющий всех сотрудников с Новым Годом, женщин с 8 марта, а мужчин с 23 февраля, если сегодня соответствующий день.

Листинг 2: Праздники

```
enum Parties{NONE, NEW_YEAR, MARCH_8, FEB_23}
private static final Parties today = Parties.NONE;

private static void celebrate(Employee[] emp) {
```

```
    for (int i = 0; i < emp.length; i++) {
        switch (today) {
            case NEW_YEAR:
                System.out.println(emp[i].name + ", happy New Year!");
                break;
            case FEB_23:
                if (emp[i].gender == Employee.Genders.MALE)
                    System.out.println(emp[i].name + ", happy February 23rd!");
                break;
            case MARCH_8:
                if (emp[i].gender == Employee.Genders.FEMALE)
                    System.out.println(emp[i].name + ", happy march 8th!");
                break;
            default:
                System.out.println(emp[i].name + ", celebrate this morning!");
        }
    }
}
```

5.4.8. Рефлексия и завершение семинара

- **Цель этапа:** Привести урок к логическому завершению, посмотреть что студентам удалось, что было сложно и над чем нужно еще поработать
- **Тайминг:** 5-10 минут
- **Действия преподавателя:**
  - Запросить обратную связь от студентов.
  - Подчеркните то, чему студенты научились на занятии.
  - Дайте рекомендации по решению заданий, если в этом есть необходимость
  - Дайте краткую обратную связь студентам.
  - Поделитесь ощущением от семинара.
  - Поблагодарите за проделанную работу.

Приложения

A. Домашнее задание 3

Листинг 3: Основная программа

```
package ru.gb.jcore;

import java.util.Arrays;

public class Exceptional {
   private static final class ColumnMismatchException extends RuntimeException {
      ColumnMismatchException(String message) {
         super("Columns exception: " + message);
      }
   }

   private static final class NumberIsNotNumberException extends RuntimeException {
      NumberIsNotNumberException(String message) {
         super("Not a number found: " + message);
      }
   }

   private static final class RowMismatchException extends RuntimeException {
      RowMismatchException(int expected, int current, String value) {
         super(String.format("Rows exception: expected %d rows. Received %d rows in '%s' string",
               expected, current, value));
      }
   }

   private static final String CORRECT_STRING= "1 3 1 2\n2 3 2 2\n5 6 7 1\n3 3 1 0";
   private static final String EXTRA_ROW_STRING= "1 3 1 2\n2 3 2 2\n5 6 7 1\n3 3 1 0\n1 2 3 4";
   private static final String EXTRA_COL_STRING= "1 3 1 2 1\n2 3 2 2 1\n5 6 7 1 1\n3 3 1 0 1";
   private static final String NO_ROW_STRING= "1 3 1 2\n2 3 2 2\n5 6 7 1";
   private static final String NO_COL_STRING= "1 3 1 2\n2 3 2 2\n5 6 7 1\n3 3 1";
   private static final String HAS_CHAR_STRING= "1 3 1 2\n2 3 2 2\n5 6 7 1\n3 3 1 A";

   private static final int MATRIX_ROWS= 4;
   private static final int MATRIX_COLS= 4;

   private static String[][] stringToMatrix(String value) {
      String[] rows = value.split("\n");
      if (rows.length !=MATRIX_ROWS)
         throw new RowMismatchException(MATRIX_ROWS, rows.length, value);

      String[][] result = new String[MATRIX_ROWS][];
      for (int i = 0; i < result.length; i++) {
         result[i] = rows[i].split(" ");
         if (result[i].length !=MATRIX_COLS)
            throw new ColumnMismatchException(result[i].length + ":\n" + value);
      }
      return result;
   }

   private static float calcMatrix(String[][] matrix) {
      float result = 0;
      int len = 0;
      for (int i = 0; i < matrix.length; i++) {
         for (int j = 0; j < matrix[i].length; j++) {
            try {
               result += Integer.parseInt(matrix[i][j]);
               ++len;
            } catch (NumberFormatException e) {
               throw new NumberIsNotNumberException(matrix[i][j]);
```

```
            }
         }
      }
      return result / len;
   }

   public static void main(String[] args) {
      try {
//          final String[][] matrix = stringToMatrix(CORRECT_STRING);
//          final String[][] matrix = stringToMatrix(NO_ROW_STRING);
//          final String[][] matrix = stringToMatrix(NO_COL_STRING);
         final String[][] matrix =stringToMatrix(HAS_CHAR_STRING);
         System.out.println(Arrays.deepToString(matrix));
         System.out.println("Half sum = " +calcMatrix(matrix));
      } catch (NumberIsNotNumberException exceptionObjectName) {
         System.out.println("A NumberFormatException is thrown: " + exceptionObjectName.getMessage());
      } catch (RowMismatchException | ColumnMismatchException superExceptionName) {
         System.out.println("A RuntimeException successor is thrown: " + superExceptionName.getMessage());
      }
   }
}
```