

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«МИРЭА - Российский технологический университет»**

**РТУ МИРЭА**

Институт Информационных технологий

Кафедра математического обеспечения и стандартизации

информационных технологий

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4**

по дисциплине «Объектно-ориентированное программирование»

|  |  |
| --- | --- |
| Выполнил студент группы ИКБО-12-17 | А.А. Лисовой |

|  |  |  |
| --- | --- | --- |
| Принял | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (должность, звание, ученая степень) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (И.О. Фамилия) |

|  |  |  |
| --- | --- | --- |
| Работа выполнена | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Подпись студента) | «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2018 г. |

|  |  |  |
| --- | --- | --- |
| Работа зачтена | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Подпись преподавателя) | «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2018 г. |

Москва 2018

**Постановка задачи**

Разработать классы, обеспечивающие заполнение коллекций объектов и объектов в отдельности данными, хранящимися в файлах.

Для реализации заданий выберите один объект из реализованных в лабораторной 3 и выполните разработку для него. Включите в интерфейс пользователя возможность продемонстрировать выполнение заданий.

Задание 1. Управление двоичными файлами

Разработать класс, обеспечивающий запись объектов в файл и чтение из файла, используя классы FileInputStream, FileOutputStream, и классов, обеспечивающих сериализацию объектов ObjectOutputStream (сериализация объектов) и ObjectInputStream (десериализация объектов).

Методы класса:

1. Применение потока объектов

* Запись одного сериализованного объекта в файл FileOutputStream посредством потока ObjectOutputStream. Ввод данных для объекта с консоли.
* Чтение одного объекта из файла, используя потоки FileInputStream, ObjectInputStream.
* Отображение прочитанного объекта.
* Создание коллекции объектов, запись всей коллекции в файл.
* Коллекции объектов: для четного варианта по лабораторной 3 ArrayList и для нечетного LinkedList.
* Чтение объектов из файла в коллекцию.
* Отобразить коллекцию объектов.

2. Применение потока Scanner

* Создать поток Scanner для считывания из текстового файла данных для заполнения полей объекта, метод считывания должен соответствовать типу поля объекта.

Задание 2. Исследование возможностей класса Files по управлению файлами

Создать класс, по управлению файлом, созданным в задании 1. Применение клаcса Files для ввода и вывода из файлов

Разработать методы, обеспечивающие реализацию функций:

* Проверку существования файла.
* Удаление файла.
* Копирование файла.
* Перемещение файла.
* Создание нового файла и запись в него отобранных по критерию данных.
* Чтение файла.

**Программная реализация**

|  |
| --- |
| package Lab\_4;  // Список импортов  import java.io.File;  import java.io.FileOutputStream;  import java.io.FileInputStream;  import java.io.IOException;  import java.io.ObjectOutputStream;  import java.io.ObjectInputStream;  import java.io.PrintWriter;  import java.util.LinkedList;  import java.util.Scanner;  // Основной класс первого задания  public class Lab\_4\_1  {  // Основной метод класса  public static void main(String[] args)  {  int command;  Scanner input = new Scanner(System.in);  // Вывод меню программы  show\_menu();  do  {  System.out.print("\nВведите число: ");  // Проверка ввода переменной выбора  while (!input.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите вариант: ");  input.next();  }  command = input.nextInt();  switch (command)  {  case 1:  // Запись сериализованного объекта в файл  one\_object\_write();  break;  case 2:  // Чтение и отбражение сериализованного объекта  one\_object\_show();  break;  case 3:  // Создание коллекции объекта и запись её в файл  object\_collection();  break;  case 4:  // Чтение и отображение коллекции объектов из файла  collection\_show();  break;  case 5:  // Запись текста в текстовый файл  file\_write();  break;  case 6:  // Считывание и вывод данных из текстового файла  file\_show();  break;  case 7:  // Повторный вывод меню  show\_menu();  break;  case 8:  System.out.println("Выход из программы");  break;  default:  System.out.println("Ошибка ввода");  }  }  while (command != 8);  }  // Метод вывода меню программы  public static void show\_menu()  {  System.out.println("Меню программы:");  System.out.println("1 - Запись сериализованного объекта в файл");  System.out.println("2 - Чтение и отбражение сериализованного  объекта");  System.out.println("3 - Создание коллекции объектов и запись её в  файл");  System.out.println("4 - Чтение и отображение коллекции объектов из  файла");  System.out.println("5 - Запись текста в текстовый файл");  System.out.println("6 - Считывание и вывод данных из текстового  файла");  System.out.println("7 - Повторный вывод меню");  System.out.println("8 - Выход из программы");  }  // Метод проверки факта наличия файла  public static boolean file\_existension(String file\_name)  {  File file = new File(file\_name);  return file.isFile();  }  // Метод записи одиночного объекта в файл  public static void one\_object\_write()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  Scanner scanner = new Scanner(System.in);  System.out.print("Введите номер дня: ");  while (!scanner.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите номер дня: ");  scanner.next();  }  int temp\_day = scanner.nextInt();  System.out.print("Введите место встречи: ");  String temp\_place = input.nextLine();  System.out.print("Введите время начала: ");  while (!scanner.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите время начала ");  scanner.next();  }  int temp\_begin = scanner.nextInt();  System.out.print("Введите время конца: ");  while (!scanner.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите время конца: ");  scanner.next();  }  int temp\_end = scanner.nextInt();  Request request = new Request(temp\_day, temp\_place, temp\_begin,  temp\_end);  try  {  // Запись объекта в файл  ObjectOutputStream out = new ObjectOutputStream(new  FileOutputStream(file\_name));  out.writeObject(request);  System.out.println("Объект записан в файл");  // Закрытие потока  try  {  if (out != null) out.close();  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка закрытия потока");  }  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка записи объекта в файл");  }  }  // Метод чтения одного объекта из файла  public static String one\_object\_read(String file\_name)  {  try  {  // Чтение объекта из файла  ObjectInputStream in = new ObjectInputStream(new  FileInputStream(file\_name));  String string = in.readObject().toString();  // Закрытие потока  try  {  if (in != null) in.close();  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка закрытия потока");  }  return string;  }  catch (Exception exception)  {  System.out.println(exception.getMessage());  return null;  }  }  // Метод отображение прочтитанного объекта  public static void one\_object\_show()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  // Проверка существования файла  if (file\_existension(file\_name))  {  // Чтение объекта из файла  String string = one\_object\_read(file\_name);  if (string == null)  {  System.out.println("Ошибка при чтении файла");  }  else  {  // Вывод объекта  System.out.println("Прочитанный объект:");  System.out.println(string);  }  }  else System.out.println("Ошибка: нет файла для чтения");  }  // Метод создания и записи в файл коллекции объектов  public static void object\_collection()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  Scanner scanner = new Scanner(System.in);  LinkedList<Request> list = new LinkedList<Request>();  System.out.print("Введите количество записей в списке: ");  int size = scanner.nextInt();  if (size < 0)  {  System.out.println("Ошибка: недопустимое количество записей в  списке");  }  else  {  scanner.nextLine();  // Заполнение списка  for (int index = 0; index < size; index++)  {  System.out.print("Введите номер дня: ");  while (!scanner.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите номер дня: ");  scanner.next();  }  int temp\_day = scanner.nextInt();  System.out.print("Введите место встречи: ");  String temp\_place = input.nextLine();  System.out.print("Введите время начала: ");  while (!scanner.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите время начала ");  scanner.next();  }  int temp\_begin = scanner.nextInt();  System.out.print("Введите время конца: ");  while (!scanner.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите время конца: ");  scanner.next();  }  int temp\_end = scanner.nextInt();  list.add(new Request(temp\_day, temp\_place, temp\_begin,  temp\_end));  }  try  {  // Запись списка в файл  ObjectOutputStream out = new ObjectOutputStream(new  FileOutputStream(file\_name));  out.writeObject(list);  System.out.println("Коллекция записана в файл");  // Закрытие потока  try  {  if (out != null) out.close();  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка закрытия потока");  }  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка записи коллекции элементов в  файл");  }  }  }  // Метод чтения коллекции объектов из файла  public static LinkedList<Request> collection\_read(String file\_name)  {  try  {  // Чтение коллекции из файла  ObjectInputStream in = new ObjectInputStream(new  FileInputStream(file\_name));  LinkedList<Request> list =  (LinkedList<Request>)in.readObject();  // Закрытие потока  try  {  if (in != null) in.close();  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка закрытия потока");  }  return list;  }  catch (Exception exception)  {  System.out.println(exception.getMessage());  return null;  }  }  // Метод отображения прочитанной коллекции файлов  public static void collection\_show()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  if (file\_existension(file\_name))  {  // Чтение коллекции из файла  LinkedList<Request> list = collection\_read(file\_name);  if (list == null)  {  System.out.println("Ошибка при чтении файла");  }  else  {  // Вывод коллекции объектов  System.out.println("Содержимое файла:");  int size = list.size();  for (int index = 0; index < size; index++)  {  System.out.println(list.get(index).toString());  }  }  }  else System.out.println("Ошибка: нет файла для чтения");  }  // Метод записи текста в текстовый файл  public static void file\_write()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  Scanner scanner = new Scanner(System.in);  System.out.print("Введите текст: ");  String text = scanner.nextLine();  try  {  PrintWriter writer = new PrintWriter(file\_name);  writer.write(text);  System.out.println("Текст записан в текстовый файл");  // Закрытие потока  try  {  if (writer != null)  writer.close();  }  catch (Exception exception)  {  exception.printStackTrace();  System.out.println("Ошибка закрытия потока");  }  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка записи текста в файл");  }  }  // Метод считывания данных из текстового файла  public static LinkedList<String> file\_read(String file\_name)  {  try  {  LinkedList<String> list = new LinkedList<String>();  File file = new File(file\_name);  Scanner scanner = new Scanner(file);  // Построчное чтение текстового файла  while (scanner.hasNext())  {  list.add(scanner.nextLine());  }  return list;  }  catch (Exception exception)  {  exception.printStackTrace();  return null;  }  }  // Вывод содержимого файла  public static void file\_show()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  if (file\_existension(file\_name))  {  // Чтение текстового файла  LinkedList<String> list = file\_read(file\_name);  if (list == null)  {  System.out.println("Ошибка при чтении файла");  }  else  {  // Вывод текста из файла  System.out.println("Содержимое файла:");  int size = list.size();  for (int index = 0; index < size; index++)  {  System.out.println(list.get(index));  }  }  }  else System.out.println("Ошибка: нет файла для чтения");  }  } |

Листинг 1. Исходный код класса Lab\_4\_1.

|  |
| --- |
| package Lab\_4;  // Список импортов  import java.io.File;  import java.io.FileOutputStream;  import java.io.FileInputStream;  import java.io.IOException;  import java.io.ObjectOutputStream;  import java.nio.file.Files;  import java.nio.file.Path;  import java.nio.file.Paths;  import java.nio.file.StandardCopyOption;  import java.util.LinkedList;  import java.util.Scanner;  // Основной класс второго задания  public class Lab\_4\_2  {  // Основной метод класса  public static void main(String[] args)  {  int command;  Scanner input = new Scanner(System.in);  // Вывод меню программы  show\_menu();  do  {  System.out.print("\nВведите число: ");  // Проверка ввода переменной выбора  while (!input.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите вариант: ");  input.next();  }  command = input.nextInt();  switch (command)  {  case 1:  // Чтение и вывод файла  read\_file();  break;  case 2:  // Удаление файла  delete\_file();  break;  case 3:  // Копирование файла  copy\_file();  break;  case 4:  // Перемещение файла  move\_file();  break;  case 5:  // Создание нового файла и запись в него отобранных по  длине пароля из коллекции пользователей  pick\_users\_from\_file();  break;  case 6:  // Повторный вывод меню  show\_menu();  break;  case 7:  System.out.println("Выход из программы");  break;  default:  System.out.println("Ошибка ввода");  }  }  while (command != 7);  }  // Метод вывода меню программы  public static void show\_menu()  {  System.out.println("Меню программы:");  System.out.println("1 - Чтение и вывод файла");  System.out.println("2 - Удаление файла");  System.out.println("3 - Копирование файла");  System.out.println("4 - Перемещение файла");  System.out.println("5 - Создание нового файла и запись в него  отобранных по длине логина из коллекции пользователей");  System.out.println("6 - Повторный вывод меню");  System.out.println("7 - Выход из программы");  }  // Метод проверки факта наличия файла  public static boolean file\_existension(String file\_name)  {  File file = new File(file\_name);  return file.isFile();  }  // Метод чтения и вывода файла  public static void read\_file()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  // Проверка существования файла  if (file\_existension(file\_name))  {  try  {  FileInputStream in = new FileInputStream(new  File(file\_name));  int content;  System.out.println("Содержимое файла:");  // Посимвольный вывод содержимого файла  while ((content = in.read()) != -1)  {  // Преобразование в символ и вывод информации  System.out.print((char)content);  }  System.out.println();  // Закрытие потока  try  {  if (in != null) in.close();  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка закрытия потока");  }  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка чтения файла");  }  }  else System.out.println("Ошибка: нет файла для чтения");  }  // Метод копирования файла  public static void copy\_file()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  // Проверка существования файла  if (file\_existension(file\_name))  {  try  {  System.out.print("Введите имя копии файла: ");  String copy\_name = input.nextLine();  // Копирование файла  File old\_file = new File(file\_name);  File new\_file = new File(copy\_name);  Files.copy(old\_file.toPath(), new\_file.toPath());  System.out.println("Копия создана");  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка копирования файла");  }  }  else System.out.println("Ошибка: нет файла для копирования");  }  // Метод удаления файла  public static void delete\_file()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла: ");  String file\_name = input.nextLine();  // Проверка существования файла  if (file\_existension(file\_name))  {  File file = new File(file\_name);  // Удаление файла  if (file.delete()) System.out.println("Файл удалён");  else System.out.println("Ошибка при удалении файла");  }  else System.out.println("Ошибка: нет файла для удаления");  }  // Перемещение файла  public static void move\_file()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите полное имя файла: ");  String file\_name = input.nextLine();  // Проверка существования файла  if (file\_existension(file\_name))  {  System.out.print("Введите новое полное имя файла: ");  String new\_name = input.nextLine();  try  {  // Перемещение файла  Path source\_path = Paths.get(file\_name);  Path destination\_path = Paths.get(new\_name);  Files.move(source\_path, destination\_path,  StandardCopyOption.REPLACE\_EXISTING);  System.out.println("Файл перемещён");  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка копирования файла");  }  }  else System.out.println("Ошибка: нет файла для перемещения");  }  // Создание нового файла и запись в него отобранных по номеру дня  объектов  public static void pick\_requests\_from\_file()  {  Scanner input = new Scanner(System.in);  System.out.print("Введите имя файла с коллекцией: ");  String file\_name = input.nextLine();  // Проверка существования файла  if (file\_existension(file\_name))  {  LinkedList<Request> list = Lab\_4\_1.collection\_read(file\_name);  if (list == null)  {  System.out.println("Ошибка при чтении файла");  }  else  {  // Отбор данных по критерию  System.out.print("Введите номер дня, по которому будут  отобраны запросы: ");  while (!input.hasNextInt())  {  System.out.println("Ошибка ввода");  System.out.println();  System.out.print("Введите номер дня ");  input.next();  }  int current\_day = input.nextInt();  int size = list.size();  LinkedList<Request> new\_list = new LinkedList<Request>();  for (int index = 0; index < size; index++)  {  if (list.get(index).get\_day() == current\_day)  {  new\_list.add(list.get(index));  }  }  System.out.print("Введите имя нового файла с коллекцией:  ");  input.nextLine();  file\_name = input.nextLine();  try  {  ObjectOutputStream out = new ObjectOutputStream(new  FileOutputStream(file\_name));  // Запись списка в файл  out.writeObject(new\_list);  System.out.println("Коллекция записана в файл");  // Закрытие потока  try  {  if (out != null) out.close();  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка закрытия потока");  }  }  catch (IOException exception)  {  exception.printStackTrace();  System.out.println("Ошибка записи коллекции в файл");  }  }  }  else System.out.println("Ошибка: нет файла для отбора данных");  }  } |

Листинг 2. Исходный код класса Lab\_4\_2.

|  |
| --- |
| package Lab\_4;  // Список импортов  import java.io.Serializable;  // Класс запроса из предыдущей лабораторной работы  public class Request implements Serializable  {  // Поля класса  private int day; // День недели  private String place; // Место встречи  private int start\_hour; // Время время начала  private int end\_hour; // Время окончания  // Конструктор с параметрами  public Request(int day, String place, int from, int to)  {  this.day = day;  this.place = place;  this.start\_hour = from;  this.end\_hour = to;  }  // Метод получения места встречи  public String get\_place()  {  return place;  }  // Метод получения дня недели встречи  public int get\_day()  {  return day;  }  // Метод получения времени начала встречи  public int get\_start\_hour()  {  return start\_hour;  }  // Метод получения времени завершения встречи  public int get\_end\_hour()  {  return end\_hour;  }  // Метод сериализации объекта класса  @Override  public String toString()  {  return this.day + " " + this.place + " " + this.start\_hour + " " +  this.end\_hour;  }  } |

Листинг 3. Исходный код класса Request.

**Тестирование**

**Задание 1**

Меню программы:

1 - Запись сериализованного объекта в файл

2 - Чтение и отбражение сериализованного объекта

3 - Создание коллекции объектов и запись её в файл

4 - Чтение и отображение коллекции объектов из файла

5 - Запись текста в текстовый файл

6 - Считывание и вывод данных из текстового файла

7 - Повторный вывод меню

8 - Выход из программы

Введите число: g

Ошибка ввода

Введите вариант: 566

Ошибка ввода

Введите число: п

Ошибка ввода

Введите вариант: 2

Введите имя файла: req

Ошибка: нет файла для чтения

Введите число: 1

Введите имя файла: request

Введите номер дня: 5

Введите место встречи: Cafe

Введите время начала: 14

Введите время конца: 16

Объект записан в файл

Введите число: 7

Меню программы:

1 - Запись сериализованного объекта в файл

2 - Чтение и отбражение сериализованного объекта

3 - Создание коллекции объектов и запись её в файл

4 - Чтение и отображение коллекции объектов из файла

5 - Запись текста в текстовый файл

6 - Считывание и вывод данных из текстового файла

7 - Повторный вывод меню

8 - Выход из программы

Введите число: 2

Введите имя файла: request

Прочитанный объект:

5 Cafe 14 16

Введите число: 3

Введите имя файла: requests

Введите количество записей в списке: 3

Введите номер дня: 3

Введите место встречи: Cafe

Введите время начала: 14

Введите время конца: 16

Введите номер дня: 5

Введите место встречи: School

Введите время начала: 17

Введите время конца: 18

Введите номер дня: 3

Введите место встречи: Park

Введите время начала: 10

Введите время конца: 16

Коллекция записана в файл

Введите число: 4

Введите имя файла: reqs

Ошибка: нет файла для чтения

Введите число: 7

Меню программы:

1 - Запись сериализованного объекта в файл

2 - Чтение и отбражение сериализованного объекта

3 - Создание коллекции объектов и запись её в файл

4 - Чтение и отображение коллекции объектов из файла

5 - Запись текста в текстовый файл

6 - Считывание и вывод данных из текстового файла

7 - Повторный вывод меню

8 - Выход из программы

Введите число: g

Ошибка ввода

Введите вариант: 4

Введите имя файла: requests

Содержимое файла:

3 Cafe 14 16

5 School 17 18

3 Park 10 16

Введите число: 6

Введите имя файла: textant

Ошибка: нет файла для чтения

Введите число: 5

Введите имя файла: text

Введите текст: I very want sleeeeeeeeeo

Текст записан в текстовый файл

Введите число: 6

Введите имя файла: text

Содержимое файла:

I very want sleeeeeeeeeo

Введите число: g

Ошибка ввода

Введите вариант: 7

Меню программы:

1 - Запись сериализованного объекта в файл

2 - Чтение и отбражение сериализованного объекта

3 - Создание коллекции объектов и запись её в файл

4 - Чтение и отображение коллекции объектов из файла

5 - Запись текста в текстовый файл

6 - Считывание и вывод данных из текстового файла

7 - Повторный вывод меню

8 - Выход из программы

Введите число: 0

Ошибка ввода

Введите число: 8

Выход из программы

**Задание 2**

Меню программы:

1 - Чтение и вывод файла

2 - Удаление файла

3 - Копирование файла

4 - Перемещение файла

5 - Создание нового файла и запись в него отобранных по номеру дня из коллекции запросов

6 - Повторный вывод меню

7 - Выход из программы

Введите число: ggg

Ошибка ввода

Введите вариант: 5

Введите имя файла с коллекцией: reqsedf

Ошибка: нет файла для отбора данных

Введите число: 333

Ошибка ввода

Введите число: 1

Введите имя файла: text

Содержимое файла:

I very want sleeeeeeeeeo

Введите число: 1

Введите имя файла: request

Содержимое файла:

Lab\_4.Requestæ²RÄV[x I dayIend\_hourI

start\_hourL placet Ljava/lang/String;xp

t Cafe

Введите число: 3

Введите имя файла: request

Введите имя копии файла: requestic

Копия создана

Введите число: 2

Введите имя файла: request

Файл удалён

Введите число: 1

Введите имя файла: request

Ошибка: нет файла для чтения

Введите число: 1

Введите имя файла: requestic

Содержимое файла:

Lab\_4.Requestæ²RÄV[x I dayIend\_hourI

start\_hourL placet Ljava/lang/String;xp t Cafe

Введите число: 4

Введите полное имя файла: requestic

Введите новое полное имя файла: C:\Users\Delamart\Documents\requestic

Файл перемещён

Введите число: 5

Введите имя файла с коллекцией: requests

Введите номер дня, по которому будут отобраны запросы: 3

Введите имя нового файла с коллекцией: requests3

Коллекция записана в файл

Введите число: 1

Введите имя файла: requests3

Содержимое файла:

Lab\_4.Requestæ²RÄV[x I dayIend\_hourI

start\_hourL placet Ljava/lang/String;xp t Cafesq ~

t Parkx

Введите число: 1

Введите имя файла: request

Ошибка: нет файла для чтения

Введите число: 1

Введите имя файла: requests

Содержимое файла:

Lab\_4.Requestæ²RÄV[x I dayIend\_hourI

start\_hourL placet Ljava/lang/String;xp t Cafesq ~ t Schoolsq ~

t Parkx

Введите число: 7

Выход из программы

**Выводы**

В ходе выполнения лабораторной работы были получены практические навыки по работе с двоичными и текстовыми файлами. Для удобства работы с программой было разработано консольное меню.