МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНЖЕНЕРНО-ТЕХНОЛОГИЧЕСКАЯ АКАДЕМИЯ

ИНСТИТУТ КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

КАФЕДРА МОПЭВМ

Отчет по практике

Вид практики производственная

Место прохождения практики ОАО "Сбербанк России" ДО №5230/0728 Стр Таганрогского отделения №5230

Сроки практики с 03.08.2015 по 30.08.2015 г.

Руководитель практики от предприятия

Колесникова Татьяна Александровна

Фамилия Имя Отчество

_____ _____

дата подпись

Руководитель практики от университета

Болгов Максим Викторович

Фамилия Имя Отчество

_____ _____

дата подпись

Студент

Пашинин Антон Андреевич

Фамилия Имя Отчество

_____ _____

дата подпись

Таганрог, 2015

ОТЗЫВ - ХАРАКТЕРИСТИКА

Студент(ка) 3 курса кафедры математического обеспечения и принятия ЭВМ Института компьютерных технологий и информационной безопасности Инженерно-технологической академии Южного федерального университета

Панышин Антон Анварович

Фамилия Имя Отчество

с «3» августа 2015 г. по «30» августа 2015 г. прошёл(ла) учебную практику в ОАО "Сбербанк России" ДО №5230/0728 Ставропольского отд. №5230

Наименование организации

В период практики выполнял(а) обязанности практиканта

За время прохождения практики студента Панышина Антона Анваровича показал(а) высокий уровень теоретической подготовки, а также умение применить и использовать знания, полученные в университете, для решения поставленных перед ним (ней) практических задач.

Студентом (кой) выполнены следующие виды работ: разработка приложения для отслеживания и мониторинга котировок курса доллара и евро.

Студентом (кой) проявлены следующие личностные и профессиональные качества: целеустремленность и ответственность, а также, студент Панышин А.А. проявил себя с положительной стороны, как работоспособный и внимательный.

Программа практики выполнена полностью / ~~частично~~.

Ненужное зачеркнуть

В целом работа практиканта Панышина Антона Анваровича

Фамилия Имя Отчество

заслуживает оценки отлично

Руководитель практики от предприятия

Колесникова Наталья Александровна

Фамилия Имя Отчество

Подпись _____ **удостоверяю.**

Руководитель организации

Колесникова Наталья Александровна

Фамилия Имя Отчество

_____ _____

дата подпись

Место печати

ДНЕВНИК
прохождения производственной практики

с 3 (число) августа (месяц) по 30 (число) августа (месяц) 2015 года

студентом 3 курса кафедры математического обеспечения и прикладной ЭВМ Института компьютерных технологий и информационной безопасности Инженерно-технологической академии Южного федерального университета

Пальмин Антон Анварович
Фамилия Имя Отчество

в ОАО "Сбербанк России" ДО №5230/0728 Старший операционный офис №5230
Наименование организации

Руководитель практики от организации Колесникова Татьяна Александровна
Фамилия Имя Отчество, должность

№ п/п	Дата	Краткое содержание выполненной работы	Отметка об исполнении
1	2	3	4
1	03.08	Получение инд. задания	отлично
2	04.08-05.08	Знакомство с АРТ СБ РФ	отлично
3	06.08-08.08	Изучение архитектуры прил.	отлично
4	09.08-15.08	Выбор ХМЛ парсера и его реализация.	отлично
5	16.08-23.08	Создание основных структур хранения данных	отлично
6	24.08-29.08	Тестирование и отладка приложения	отлично
7	30.08	Защита практики	отлично

Студент _____ Пальмин Антон Анварович
Подпись Фамилия Имя Отчество

Содержание

Список использованных сокращений

API – Application Programming Interface, набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) для использования во внешних программных продуктах. Используется при написании клиентских приложений.

IDE – Integrated Development Environment, интегрированная среда разработки.

Java SE – кроссплатформенный язык программирования, SE – стандартная версия платформы.

Введение

Практика является одним из важнейших этапов подготовки высшими образовательными заведениями высококвалифицированных специалистов. Практика студентов Южного федерального университета является составной частью образовательной программы высшего профессионального образования и представляет собой одну из форм организации учебного процесса, заключающуюся в профессионально-практической подготовке студентов на базах практик. Основными целями практики, которые стоят перед студентами при ее прохождении согласно «Положению о практиках студентов Южного федерального университета» являются:

- закрепление, углубление и расширение теоретических знаний, умения и навыков, полученных студентами в процессе теоретического обучения;
- овладение профессионально-практическими умениями, производственными навыками и передовыми методами труда;
- овладение основами профессии в операционной сфере: ознакомление и усвоение методологии и технологии решения профессиональных задач.

1 Постановка задачи

Разработать приложение, с помощью которого пользователь мог отслеживать изменяющиеся котировки курса доллара и евро. Котировки предоставляются Центральным Банком Российской Федерации. Данная программа призвана автоматизировать процесс поиска и анализ курсов доллара и евро.

К разрабатываемой программе предъявляется следующий список требований, которым она должна удовлетворять:

1. Данная программа должна быть консольной.
2. Программа должна запускаться на следующих операционных системах:
    2.1 Windows;
    2.2 Linux;
    2.3 Solaris;
    2.4 Mac OS.
3. Пользователь должен иметь возможности просматривать текущий курс, на данный момент, а также курс на предыдущую доступную дату:
    3.1 Просматривать номинал валюты;
    3.2 Просматривать сокращенный суффикс валюты;
    3.3 Просматривать дату, на которую определён курс;
    3.4 Просматривать код доллара и евро;
    3.5 Просматривать полное название валюты;
    3.6 Просматривать само значение курса.
4. Предоставлять логгирование в файлах:
    4.1 Логгированию подлежит весь ответ от сервера ЦБ РФ;
    4.2 В файлах должны быть записаны все валюты, которые ЦБ отображает на сайте;
    4.3 Пользователь должен иметь доступ к данным файлам.

2 Список используемых технологий

Для выполнения задания по разработке программы были использованы следующие средства:

- IDE: IntelliJ IDEA 14.1.4;
- платформа: Java SE;
- XML parser: с технологией представления DOM;

2.1 IntelliJ IDEA

IntelliJ IDEA представляет собой интеллектуальный редактор для Java, JavaScript, CoffeeScript, HTML/XHTML/HAML, CSS/SASS/LESS, XML/XSL/XPath, YAML, ActionScript/MXML, Haxe, Groovy, Scala, SQL, Kotlin и других, с возможностями анализа кода на лету, предотвращения ошибок в коде и автоматизированными средствами рефакторинга для Java. Автодополнение кода в IntelliJ IDEA поддерживает спецификации Java 1.6, 1.7, 1.8. включая генераторы, сопрограммы, пространства имен, замыкания, типажи и синтаксис коротких массивов. Имеется полноценный SQL-редактор с возможностью редактирования полученных результатов запросов. Все эти возможности позволяют разработчику с большей легкостью разрабатывать проекты, что в свою очередь ускоряет процесс и улучшает качество разработки.

2.2 Java SE Platform

Бесплатно распространяемая Java SE – это стандартная версия платформы Java, предназначенная для создания и исполнения апплетов и приложений, рассчитанных на индивидуальное пользование или на использование в масштабах малого предприятия. Не включает в себя многие возможности, предоставляемые более мощной и расширенной платформой Java Enterprise Edition, рассчитанной на создание коммерческих приложений масштаба крупных и средних предприятий.

Платформа Java SE состоит из набора служб, API-интерфейсов и протоколов, предоставляющих функциональные возможности для разработки многоуровневых веб-приложений.

Java SE упрощает разработку приложений и снижает потребность в программировании и обучении программированию, обеспечивая создание стандартизованных, повторно используемых модульных компонентов и поддерживая уровень автоматической обработки многих прикладных аспектов программирования.

2.3 XML parser

DOM (Document Object Model – объектная модель документов) – платформенно-независимый программный интерфейс, позволяющий программам и скриптам управлять содержимым документов HTML и XML, а также изменять их структуру и оформление. Модель DOM не накладывает ограничений на структуру документа. Любой документ известной структуры с помощью DOM может быть представлен в виде дерева узлов, каждый узел которого содержит элемент, атрибут, текстовый, графический или любой другой объект. Узлы связаны между собой отношениями родитель-потомок.

Объектная модель документа (DOM) представляется как иерархия узлов, объектов с интерфейсом Node. Остальные интерфейсы являются расширением Node:

Document - документ в целом. В качестве дочерних узлов может иметь Element (макс. один), ProcessingInstruction, Comment, DocumentType (макс. один);

DocumentFragment - описывает кусок документа, по сути является облегченной версией Document. В качестве дочерних узлов может иметь Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference;

DocumentType - тип документа, не может иметь дочерних узлов;

EntityReference - ссылка на xml сущность. Вместо создания узлов этого типа xml процессор может полностью раскрыть все сущности при построении документа. В качестве дочерних узлов может иметь Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference;

Element - элемент xml документа. В качестве дочерних узлов может иметь Element, Text, Comment, ProcessingInstruction, CDATASection, EntityReference;

Attr - атрибут элемента. Хотя это расширение Node, DOM не считает подобные узлы частью своей иерархии. Для них свойства parentNode, previousSibling и nextSibling всегда равны null. Узлы Text, EntityReference могут быть дочерними для представления значения атрибута;

ProcessingInstruction - специфические инструкции обработки для xml процессора;

CharacterData - вспомогательный интерфейс для доступа к символьным данным документа. Является базовым для узлов Comment и Text;

Comment - содержимое комментария;

Text - текстовое содержимое Element или Attr;

CDATASection - CDATA секция;

Entity - xml сущность, не важно разобранная или нет xml процессором. В качестве дочерних узлов может иметь Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference;

Notation - объявленная нотация в DTD (в описании типа документа), только для чтения.

Кроме этого есть дополнительные вспомогательные интерфейсы как NodeList.

3 Реализация проекта

3.1 Анализ требований и составление плана работ

Передо мной была поставлена задача разработки приложения, с помощью которого пользователь мог отслеживать изменяющиеся котировки курса доллара и евро.

Для реализации данной цели мною были изучены технологии программирования, рассмотренные в предыдущем разделе, и был составлен план, согласно которому велась разработка.

План включает следующие пункты:

- разработка меню для программы;
- анализ API ЦБ РФ;
- реализация кроссплатформенного приложения;
- тестирование и отладка.

3.2 Анализ API ЦБ ФР

Были проанализированы API для работы со сервером ЦБ РФ и было выявлено, что общая структура данных у полученного xml файла такова. Сначала идет корневой элемент ValCurs Date который несет в себе значение времени на последнюю доступную дату, на которую установлен курс. Далее идет родительский xml элемент Valute ID который несет в себе уникальный идентификатор по которому можно определить, какая это котировка. Далее идут внуки, их несколько: NumCode, CharCode, Nominal, Name, Value.

NumCode – отображает код определённой котировки в международном формате.

CharCode – отображает сокращение котировки в международном формате.

Nominal – отображает номинал котировки, то есть сколько валюты можно приобрести за определённую сумму.

Name – отображает полное название котировки в международном формате.

Value – отображает текущее значение котировки, проще говоря это и есть сам курс той или иной валюты.

Весь xml файл приходит от сервера в кодировке windows-1251, что конечно же немного затрудняет преобразование его в UTF-8, который так необходим при парсинге UNIX-подобных системах.

3.3 Программная архитектура

Разрабатываемая программа представляет собой двухуровневую архитектуру:

- клиент – принимает запрос, валидирует данные запроса, делегирует обработку запроса в соответственный сервис, и посылает ответ, обработанный на запрос;
- сервис – осуществляет работу с объектами и выдаёт xml ответ.

Для наглядности, схема взаимодействия программной части с сервисом представлена на рисунке 1.

Рисунок 1 – Общая схема взаимодействия программы с сервисом.

Заключение

Результатом прохождения практики стала разработанная программа, которая позволяет отслеживать и вовремя анализировать котировки доллара и евро, а также вносить другие курсы валют в архив.

Данная программа была реализована на языке программирования Java с использование XML парсера, с технологией DOM.

Разработанный проект полностью соответствует всем поставленным в техническом задании требованиям.

В результате прохождения практики я углубил свои знания языка программирования Java, парсинга XML файлов и получил бесценный опыт разработки Enterprise приложений.

Список используемых источников

1. Положение о практиках студентов Южного федерального университета: принято на Ученом совете Южного федерально университета 27 апреля 2013 года.
2. API — Википедия [Электронный ресурс],
   URL: https://ru.wikipedia.org/wiki/API (дата обращения: 03.08.2015).
3. DOM — Википедия [Электронный ресурс],
   URL: https://ru.wikipedia.org/wiki/Document_Object_Model (дата обращения: 05.08.2015)
4. IntelliJ IDEA — Википедия [Электронный ресурс],
   URL: https://ru.wikipedia.org/wiki/IntelliJ_IDEA (дата обращения: 08.08.2015).
5. Java DOM // Oracle: Reading XML Data into a DOM.
   URL: https://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html
   (дата обращения: 30.08.2015).
6. XML Standard API – Apache [Электронный ресурс],
   URL: https://xerces.apache.org/xerces2-j/javadocs/api/org/w3c/dom/Document.html
   (дата обращения: 09.08.2015).

Приложение А

Листинг файла FileWorker.java:

```
import java.io.*;
import java.nio.file.Files;

public class FileWorker {

   private String path;
   private RandomAccessFile file;

   public FileWorker(String path) {
      this.path = path; // initialized path
   }

   public long goTo(int num) throws IOException { // transition to the specified symbol
      file = new RandomAccessFile(path, "r");
      file.seek(num);
      long pointer = file.getFilePointer(); // get cursor in file
      file.close();
      return pointer;
   }

   public static void copy(File source, File dest) throws IOException {
      Files.copy(source.toPath(), dest.toPath());
   }

   public static void delete(String nameFile) throws FileNotFoundException {
      exists(nameFile);
      new File(nameFile).delete();
   }

   public static void update(String nameFile, String newText) throws FileNotFoundException {
      exists(nameFile);
      StringBuilder sb = new StringBuilder();
      String oldFile = read(nameFile);
      sb.append(oldFile);
      sb.append(newText);
      write(nameFile, sb.toString());
   }

   public static void write(String fileName, String text) {
      File file = new File(fileName);
      try {
         if(!file.exists()){
            file.createNewFile();
         }
```

```
            try (PrintWriter out = new PrintWriter(file.getAbsoluteFile())) {
                out.print(text);
            }
        } catch(IOException e) {
            throw new RuntimeException(e);
        }
    }

    public String read() throws IOException { // read file and displays contents
        file = new RandomAccessFile(path, "r");
        String res = "";
        int b = file.read();
        while(b != -1){
            res = res + (char)b;
            b = file.read();
        }
        file.close();
        return res;
    }

    public String readFrom(int numberSymbol) throws IOException { // read file with a particular
symbol
        file = new RandomAccessFile(path, "r");
        String res = "";
        file.seek(numberSymbol);
        int b = file.read();
        while(b != -1){
            res = res + (char)b;
            b = file.read();
        }
        file.close();
        return res;
    }

    public void write(String st) throws IOException { // write file
        file = new RandomAccessFile(path, "rw"); // modifier rw (read & write)
        file.write(st.getBytes());
        file.close();
    }

    public void removeLineFromFile(String file, String lineToRemove) {
        try {
            File inFile = new File(file);
            if (!inFile.isFile()) {
                System.out.println("Parameter is not an existing file");
                return;
            }
```

```
        File tempFile = new File(inFile.getAbsolutePath() + ".tmp"); // construct the new file that
will later be renamed to the original filename
        BufferedReader br = new BufferedReader(new FileReader(file));
        PrintWriter pw = new PrintWriter(new FileWriter(tempFile));

        String line;
        //Read from the original file and write to the new
        //unless content matches data to be removed
        while ((line = br.readLine()) != null) {
           if (!line.trim().equals(lineToRemove)) {
              pw.println(line);
              pw.flush();
           }
        }
        pw.close();
        br.close();

        //Delete the original file
        if (!inFile.delete()) {
           System.out.println("Could not delete file");
           return;
        }

        // rename the new file to the filename the original file had
        if (!tempFile.renameTo(inFile)) {
           System.out.println("Could not rename file");
        }
     } catch (IOException ex) {
        ex.printStackTrace();
     }
  }

  private static void exists(String fileName) throws FileNotFoundException {
     File file = new File(fileName);
     assert file.exists() : file.getName();
  }

  public static String read(String fileName) throws FileNotFoundException {
     File file = new File(fileName);
     StringBuilder sb = new StringBuilder();
     exists(fileName);
     try (BufferedReader in = new BufferedReader(new FileReader(file.getAbsoluteFile()))) {
        String s;
        while ((s = in.readLine()) != null) {
           sb.append(s);
           sb.append("\n");
        }
     } catch (IOException e) {
        throw new RuntimeException(e);
```

```
        }
        return sb.toString();
    }
}
```

Приложение Б

Листинг файла Main.java:

```
import java.io.*;
import java.net.*;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import java.io.File;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Date;
import org.w3c.dom.Element;
import org.xml.sax.SAXException;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

public class Main {

   protected static String valCursDateCurrent, valCursDatePrev, valCursDateNew;

   public static void main(String[] args) throws IOException, ParserConfigurationException,
SAXException, ParseException {

      for (int i = 0; i < 1; i++) {

         URL url = new URL("http://www.cbr.ru/scripts/XML_daily_eng.asp?date_req="); // no
HTTPS :(
         URLConnection urlConnection = url.openConnection();
         BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(urlConnection.getInputStream(), "windows-1251"));

         String inputLine;
         PrintWriter printWriter = new PrintWriter(new File("temp.xml"));

         while ((inputLine = bufferedReader.readLine()) != null) printWriter.println(inputLine);
         printWriter.close();
         bufferedReader.close();

/* parsing XML with DOM */

         Document document =
DocumentBuilderFactory.newInstance().newDocumentBuilder().parse(new File("temp.xml"));
         NodeList nodeList = document.getElementsByTagName("Valute");
```

```
        for (int j = 0; j < nodeList.getLength(); j++) {

            Element element = (Element) nodeList.item(j);
            String numCode =
element.getElementsByTagName("NumCode").item(0).getChildNodes().item(0).getNodeValue();
            String charCode =
element.getElementsByTagName("CharCode").item(0).getChildNodes().item(0).getNodeValue();
            int nominal =
Integer.parseInt(element.getElementsByTagName("Nominal").item(0).getChildNodes().item(0).get
NodeValue());
            String name =
element.getElementsByTagName("Name").item(0).getChildNodes().item(0).getNodeValue();
            String value =
element.getElementsByTagName("Value").item(0).getChildNodes().item(0).getNodeValue();
            valCursDateCurrent =
element.getParentNode().getAttributes().getNamedItem("Date").getNodeValue();

            FileWriter fileWriterQuotation = new FileWriter("quotation_" + valCursDateCurrent +
".txt", true);
            fileWriterQuotation.write(valCursDateCurrent + " " + numCode + " " + charCode + " " +
nominal + " " + name + " " + value + "\n");
            fileWriterQuotation.close();
        }

        DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofLocalizedDate(FormatStyle.MEDIUM);
        LocalDate localDate = LocalDate.parse(valCursDateCurrent, dateTimeFormatter); // convert
from dd.MM.yyyy to yyyy-MM-dd
        LocalDate localDatePrev = localDate.minusDays(1); // minus one day (pattern format yyyy-
mm-dd)
        valCursDatePrev = String.valueOf(localDatePrev); // convert from date to String

        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
        Date date = formatter.parse(valCursDatePrev);
        SimpleDateFormat formatterPrev = new SimpleDateFormat("dd.MM.yyyy"); // convert date
from yyyy-MM-dd to dd.MM.yyyy
        valCursDatePrev = formatterPrev.format(date);
    }

    URL url = new URL("http://www.cbr.ru/scripts/XML_daily_eng.asp?date_req=" +
valCursDatePrev); // get url connection to previous date
    URLConnection urlConnection = url.openConnection(); // server give previous accessed date !
    BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(urlConnection.getInputStream(), "windows-1251"));

    String inputLine;
    PrintWriter printWriter = new PrintWriter(new File("temp.xml"));

    while ((inputLine = bufferedReader.readLine()) != null) printWriter.println(inputLine);
```

```
        printWriter.close();
        bufferedReader.close();

        Document document =
DocumentBuilderFactory.newInstance().newDocumentBuilder().parse(new File("temp.xml"));
        NodeList nodeList = document.getElementsByTagName("Valute");

        for (int i = 0; i < nodeList.getLength(); i++) {

            Element element = (Element) nodeList.item(i);
            String numCode =
element.getElementsByTagName("NumCode").item(0).getChildNodes().item(0).getNodeValue();
            String charCode =
element.getElementsByTagName("CharCode").item(0).getChildNodes().item(0).getNodeValue();
            int nominal =
Integer.parseInt(element.getElementsByTagName("Nominal").item(0).getChildNodes().item(0).get
NodeValue());
            String name =
element.getElementsByTagName("Name").item(0).getChildNodes().item(0).getNodeValue();
            String value =
element.getElementsByTagName("Value").item(0).getChildNodes().item(0).getNodeValue();
            valCursDateNew =
element.getParentNode().getAttributes().getNamedItem("Date").getNodeValue();

            FileWriter fileWriterQuotation = new FileWriter("quotation_" + valCursDateNew + ".txt",
true);
            fileWriterQuotation.write(valCursDateNew + " " + numCode + " " + charCode + " " +
nominal + " " + name + " " + value + "\n");
            fileWriterQuotation.close();
        }

        File fileValCursDateCurrent = new File("quotation_" + valCursDateCurrent + ".txt");
        File fileValCursNew = new File("quotation_" + valCursDateNew + ".txt");

        for (int i = 0; i < 1; i++) {
            File file = new File("quotation_" + valCursDateNew + ".txt");
            File file3 = new File("quotation" + valCursDateNew + ".txt");
            if (file3.exists() && file3.isFile()) {
                for (int j = 0; j < 1; j++) {
                    file3.delete();
                }
                FileWorker.copy(file, file3);
            } else {
                FileWorker.copy(file, file3);
            }
        }

        for (int i = 0; i < 1; i++) {
            File file4 = new File("quotation_" + valCursDateCurrent + ".txt");
```

```
            File file5 = new File("quotation" + valCursDateCurrent + ".txt");
            if (file5.exists() && file5.isFile()) {
                for (int j = 0; j < 1; j++) {
                    file5.delete();
                }
                FileWorker.copy(file4, file5);
            } else {
                FileWorker.copy(file4, file5);
            }
        }

        System.out.println("************************************************");

        int begin = 10; // string from which to start reading
        int end = 11; // string which finishes reading
        int counter = 0;
        BufferedReader br = new BufferedReader(new FileReader(new File("quotation" +
valCursDateCurrent + ".txt")));
        String str;
        while((str = br.readLine()) != null) {
            counter++;
            if(counter > begin-1) System.out.println(str);
            if(counter == end) break;
        }

        System.out.println("************************************************");

        int begin1 = 10; // string from which to start reading
        int end1 = 11; // string which finishes reading
        int counter1 = 0;
        BufferedReader br1 = new BufferedReader(new FileReader(new File("quotation" +
valCursDateNew + ".txt")));
        String str1;
        while((str1 = br1.readLine()) != null) {
            counter1++;
            if(counter1 > begin1-1) System.out.println(str1);
            if(counter1 == end1) break;
        }

        System.out.println("************************************************");
        System.out.println("\n");
        System.out.println(new File("temp.xml").delete() && fileValCursDateCurrent.delete() &&
fileValCursNew.delete() ? "All temp xml files successfully deleted :)" : "Temp xml files not deleted
:)");
        System.in.read();
    }
}
```