Міністерство освіти і науки України  
НТУУ «КПІ ім. І. Сікорського»

Кафедра автоматизації проектування енергетичних процесів і систем

Лабораторна робота №2

«Java ООП»

Виконав: студент II курсу ТЕФ, групи TI-92

Сопронюк Ю. А.

Перевірив: Касьянов А. С.

Дата виконання: 10.11.2020

Київ-2020

**Варіант №11**

ЗАВДАННЯ

Розробити програму - консольний додакок, з визначенням класу для конкретної предметної області (відповідно до варіанту). Визначення класу має містити лише поля і методи, що властиві для об’єктів цього класу. Для класу визначити декілька конструкторів — з параметрами і без. Визначити геттери і сеттери для полів, для яких це є доречним. Передбачити щонайменше 3 методи для створених класів (не ураховуючи геттери і сеттери). Передбачити щонайменше 1 перевантажений метод. У середені конструкторів, сеттерів і (за наявності) інших методів, що модифікують значення полів, передбачити перевірку коректності введених значень. Поля, для яких слід перевіряти коректність і критерії коректності обирайте самі для свого варіанту. В разі невиконання критеріїв коректності виводьте у консоль попередження і завершуйте роботу програми (можете використовувати, наприклад, System.exit()). Використати об'єкти визначених класів і їх методи у програмі, за межами визначеного класу (наприклад у main і методах, що викликаються з main).

***Об’єкт Книга в бібліотеці – автор, назва, рік видання, загальна кількість видачі книги читачам, середня кількість видачі на рік.***

**ВИКОНАНАННЯ РОБОТИ**

**Код програми**

***Bookish.java – інтерфейс класу Book***

public interface Bookish {  
 int getPublish();  
 void setPublish(int publish);  
  
 long getAmountIssuance();  
 void setAmountIssuance(long amountIssuance);  
  
 int getAverageIssuance();  
 void setAverageIssuance(int averageIssuance);  
  
 String getAuthor();  
 void setAuthor(String author);  
  
 String getName();  
 void setName(String name);  
}

***JSON.java – простий клас для роботи JSON-даними (був створений спеціально для демонстрації класу Book)***

import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
import org.json.simple.parser.ParseException;  
  
import java.io.FileNotFoundException;  
import java.io.FileReader;  
import java.util.Scanner;  
  
public class JSON {  
 public static String getDataFromFile(String src) {  
 StringBuilder jsonStrBuilder = new StringBuilder();  
 Scanner inputScanner;  
 try{  
 inputScanner = new Scanner(new FileReader(src));   
 while(inputScanner.hasNext()) {  
 jsonStrBuilder.append(inputScanner.next()).append(" "); // Цикл чтения из файла  
 }  
 inputScanner.close();  
 }  
 catch(FileNotFoundException e){  
 System.*err*.println(e.toString());  
 }  
 return jsonStrBuilder.toString();  
 }  
  
 public static JSONObject getJSON(String data) {  
 JSONParser jsonParser = new JSONParser();  
 JSONObject resultJson = new JSONObject();  
 try {  
 resultJson = (JSONObject) jsonParser.parse(data);  
 } catch (ParseException e) {  
 System.*err*.println(e.toString());  
 }  
 return resultJson;  
 }  
}

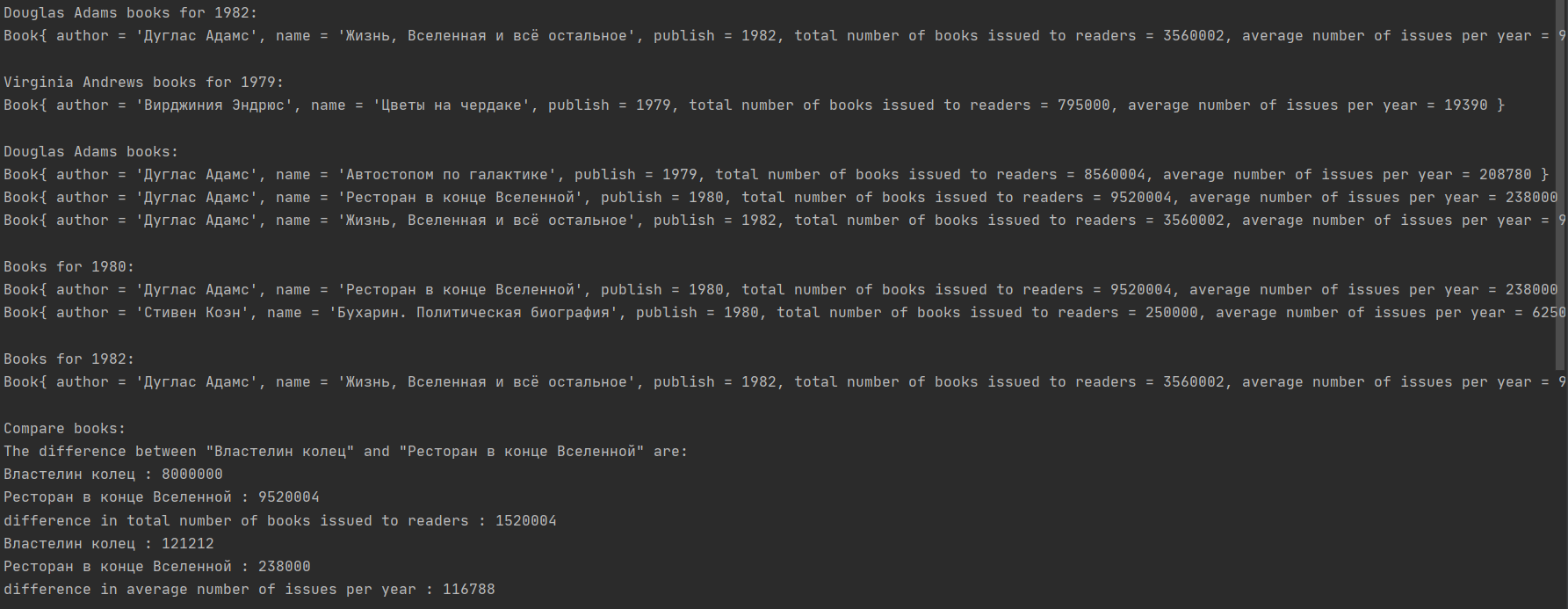
***Book.java – клас, який описує сущність “книга”***

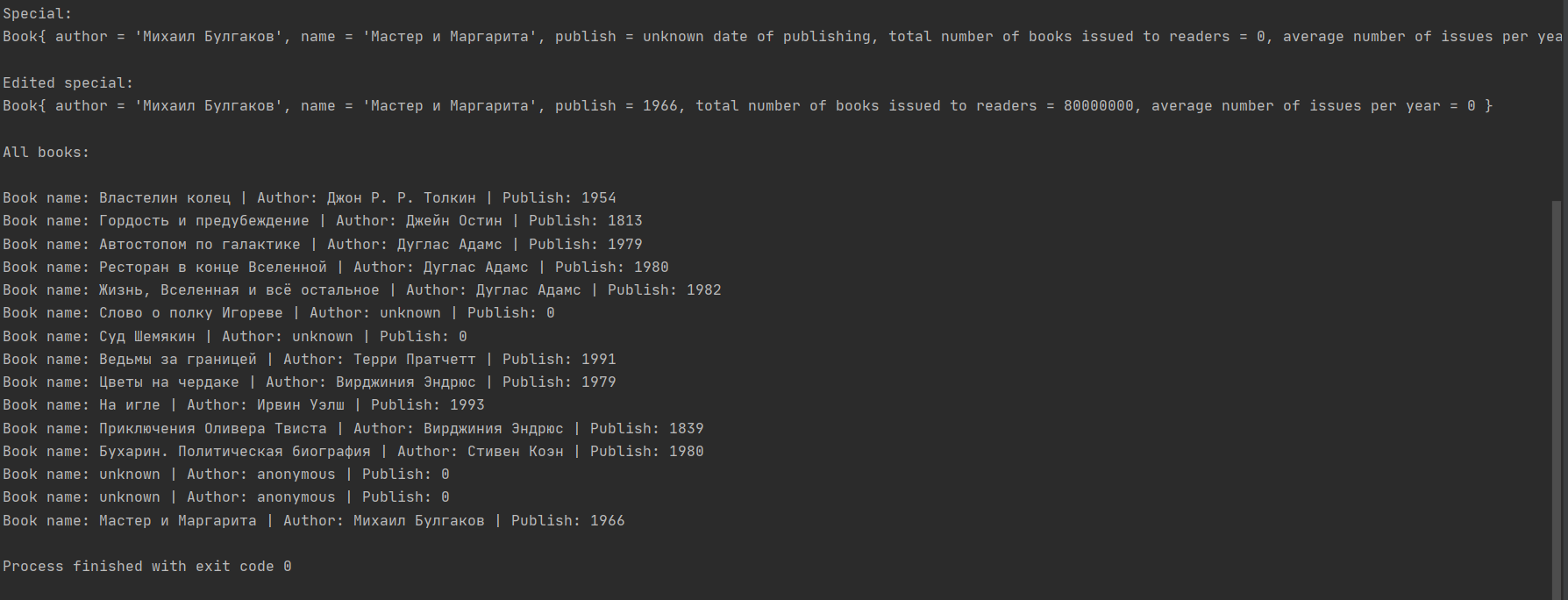
import java.time.Year;  
import java.util.ArrayList;  
  
*/\*\*  
 \* Класс библиотечной книги со свойствами <b>author</b>, <b>name</b>,  
 \* <b>publish</b>, <b>amountIssuance</b> и <b>averageIssuance</b>.  
 \** ***@author*** *Сопронюк Юрій  
 \** ***@version*** *1.0  
 \*/*public class Book implements Bookish {  
  
 */\*\* Поле автор книги \*/* private String author;  
  
 */\*\* Поле название книги \*/* private String name;  
  
 */\*\* Поле дата публикации книги \*/* private int publish;  
  
 */\*\* Поле общее количество выдачи книги читателям \*/* private long amountIssuance;  
  
 */\*\* Поле среднее количество выдачи книги за 1 год \*/* private int averageIssuance;  
  
 */\*\*  
 \* Конструктор по умолчанию - создание нового объекта  
 \** ***@see*** *Book#Book()  
 \*/* public Book() {  
 this.author = "anonymous";  
 this.name = "unknown";  
 setPublish(0);  
 setAverageIssuance(0);  
 }  
  
 */\*\*  
 \* Конструктор - создание нового объекта с определенными значениями  
 \** ***@param*** *author - автор книги  
 \** ***@param*** *name - название книги  
 \** ***@param*** *publish - дата публикации книги  
 \** ***@param*** *amountIssuance - общее количество выдачи книги читателям  
 \** ***@see*** *Book#Book(String, String, int, long)  
 \*/* public Book(String author, String name, int publish, long amountIssuance) {  
 setAuthor(author);  
 setName(name);  
 setPublish(publish);  
 setAmountIssuance(amountIssuance);  
 calcAverageIssuance(); //задает поле averageIssuance  
 }  
  
 */\*\*  
 \* Конструктор - создание нового объекта с определенными значениями  
 \** ***@param*** *author - автор книги  
 \** ***@param*** *name - название книги  
 \** ***@see*** *Book#Book(String, String)  
 \*/* public Book(String author, String name) {  
 setAuthor(author);  
 setName(name);  
 setPublish(0);  
 setAmountIssuance(0);  
 setAverageIssuance(0);  
 }  
  
 @Override  
 public String toString() {  
 return "Book{ " +  
 "author = '" + author + '\'' +  
 ", name = '" + name + '\'' +  
 ", publish = " + ((publish == 0) ? "unknown date of publishing" : publish) +  
 ", total number of books issued to readers = " + amountIssuance +  
 ", average number of issues per year = " + averageIssuance +  
 " }\n";  
 }  
  
 */\*\*  
 \* Функция получения значения поля {****@link*** *Book#publish}  
 \** ***@return*** *возвращает дату публикации книги  
 \*/* public int getPublish() {  
 return publish;  
 }  
  
 */\*\*  
 \* Процедура определения даты публикации {****@link*** *Book#publish}  
 \** ***@param*** *publish - дата публикации ()  
 \*/* public void setPublish(int publish) {  
 //Первая точно датированная печатная книга — буддийская «Алмазная сутра» — была издана 11 мая 868 года  
 //если указан 0, то неизвестаня дата публикации  
 if(publish < 868 && publish > getCurrentYear() && publish != 0) {  
 System.*out*.println("Incorrect date of publication");  
 return;  
 }  
 this.publish = publish;  
 }  
  
 */\*\*  
 \* Функция получения значения поля {****@link*** *Book#amountIssuance}  
 \** ***@return*** *возвращает общее количество выдачи книги читателям  
 \*/* public long getAmountIssuance() {  
 return amountIssuance;  
 }  
  
 */\*\*  
 \* Процедура определения общего числа выдачи книги читателям {****@link*** *Book#amountIssuance}  
 \** ***@param*** *amountIssuance - общее число выдачи книги читателям (за все года)  
 \*/* public void setAmountIssuance(long amountIssuance) {  
 //общее количество выдачи книги читателям (то есть книг, вышедших в тираж может быть больше миллиарда)  
 //если брать большой промежуток времени и популярную книгу  
 if(amountIssuance >= 0) this.amountIssuance = amountIssuance;  
 }  
  
 */\*\*  
 \* Функция получения значения поля {****@link*** *Book#averageIssuance}  
 \** ***@return*** *возвращает среднее количество выдачи книги за 1 год  
 \*/* public int getAverageIssuance() {  
 return averageIssuance;  
 }  
  
 */\*\*  
 \* Процедура <b>установки</b> среднего количество выдачи книги за 1 год {****@link*** *Book#averageIssuance}  
 \* Эта функция будет использована только в том случае, если дата публикации не известна (равна 0)  
 \** ***@param*** *averageIssuance - среднее количество выдачи книги за 1 год  
 \*/* public void setAverageIssuance(int averageIssuance) {  
 if(averageIssuance >= 0) this.averageIssuance = averageIssuance;  
 }  
  
 */\*\*  
 \* Приватная процедура <b>определения</b> среднего количество выдачи книги за 1 год {****@link*** *Book#amountIssuance}  
 \*/* private void calcAverageIssuance() {  
 if (publish == 0) {  
 averageIssuance = 0; //либо в производственном коде используйте setAverageIssuance со своим значением  
 return;  
 }  
 int amount = getCurrentYear() - publish;  
 if (amount == 0) averageIssuance = 0;  
 averageIssuance = (int) Math.*floor*((float)amountIssuance / amount);  
 }  
  
 */\*\*  
 \* Внутренняя функция для получения значения текущего года  
 \** ***@return*** *возвращает текущий год в виде числа  
 \*/* private int getCurrentYear() {  
 return Year.*now*().getValue();  
 }  
  
 */\*\*  
 \* Функция получения значения поля {****@link*** *Book#author}  
 \** ***@return*** *возвращает имя автора книги  
 \*/* public String getAuthor() {  
 return author;  
 }  
  
 */\*\*  
 \* Процедура для установки автора книги {****@link*** *Book#author}  
 \** ***@param*** *author - автор книги  
 \*/* public void setAuthor(String author) {  
 if (author.length() > 40) {  
 System.*out*.println("The author name is too long.");  
 return;  
 }  
 this.author = author;  
 }  
  
 */\*\*  
 \* Функция получения значения поля {****@link*** *Book#name}  
 \** ***@return*** *возвращает имя книги  
 \*/* public String getName() {  
 return name;  
 }  
  
 */\*\*  
 \* Процедура для установки имени книги {****@link*** *Book#name}  
 \** ***@param*** *name - имя книги  
 \*/* public void setName(String name) {  
 if (name.length() > 100) {  
 System.*out*.println("The book name is too long.");  
 return;  
 }  
 this.name = name;  
 }  
  
 */\*\*  
 \* Перегруженный статический метод для вывода книг по критериям  
 \* {****@link*** *Book#author}, {****@link*** *Book#publish}  
 \** ***@param*** *arr - список книг класса Book  
 \** ***@param*** *author - критерий - имя автора  
 \** ***@param*** *publish - критерий - дата публикации  
 \*/* public static void booksByCriterion(ArrayList<Book> arr, String author, int publish) {  
 for(Book key: arr) {  
 if(key.getAuthor().equals(author) && key.getPublish() == publish) {  
 System.*out*.print(key);  
 }  
 }  
 }  
  
 */\*\*  
 \* Перегруженный статический метод для вывода книг по критериям  
 \* {****@link*** *Book#author}  
 \** ***@param*** *arr - список книг класса Book  
 \** ***@param*** *author - критерий - имя автора  
 \*/* public static void booksByCriterion(ArrayList<Book> arr, String author) {  
 for(Book key: arr) {  
 if(key.getAuthor().equals(author)) {  
 System.*out*.print(key);  
 }  
 }  
 }  
  
 */\*\*  
 \* Перегруженный статический метод для вывода книг по критериям  
 \* {****@link*** *Book#publish}  
 \** ***@param*** *arr - список книг класса Book  
 \** ***@param*** *publish - критерий - дата публикации  
 \*/* public static void booksByCriterion(ArrayList<Book> arr, int publish) {  
 for(Book key: arr) {  
 if(key.getPublish() == publish) {  
 System.*out*.print(key);  
 }  
 }  
 }  
  
 */\*\*  
 \* Статический метод для сравнения двух книг по их двум полям:  
 \* {****@link*** *Book#amountIssuance}, {****@link*** *Book#averageIssuance}  
 \** ***@param*** *b1 - первая книга для сравнения  
 \** ***@param*** *b2 - вторая книга для сравнения  
 \*/* public static void compareBooks(Book b1, Book b2) {  
 long diffIssuanceAmount = b1.getAmountIssuance() - b2.getAmountIssuance();  
 int diffIssuanceAverage = b1.getAverageIssuance() - b2.getAverageIssuance();  
  
 System.*out*.println("The difference between \"" + b1.getName() + "\" and \"" + b2.getName() + "\" are:");  
  
 System.*out*.println(b1.getName() + " : " + b1.getAmountIssuance());  
 System.*out*.println(b2.getName() + " : " + b2.getAmountIssuance());  
 System.*out*.println("difference in total number of books issued to readers : " + Math.*abs*(diffIssuanceAmount));  
  
 System.*out*.println(b1.getName() + " : " + b1.getAverageIssuance());  
 System.*out*.println(b2.getName() + " : " + b2.getAverageIssuance());  
 System.*out*.println("difference in average number of issues per year : " + Math.*abs*(diffIssuanceAverage));  
 }  
  
 */\*\*  
 \* Статический метод для вывода списка книг в консоле  
 \** ***@param*** *books - список книг  
 \*/* public static void showBooks(ArrayList<Book> books) {  
 for(Book value: books) {  
 System.*out*.print("Book name: " + value.getName());  
 System.*out*.print(" | Author: " + value.getAuthor());  
 System.*out*.print(" | Publish: " + value.getPublish());  
 System.*out*.println();  
 }  
 }  
}

***Main.java – точка входу програми, яка працює з екземплярами класу Book***

import org.json.simple.JSONObject;  
  
import java.util.ArrayList;  
  
public class Main {  
 public static void main(String[] args) {  
  
 JSONObject arrJSON, json; // Объект со всеми данными из JSON файла  
 String data = JSON.*getDataFromFile*("src/main/resources/db.json");  
 arrJSON = JSON.*getJSON*(data);  
  
 ArrayList<JSONObject> arr = (ArrayList<JSONObject>) arrJSON.get("books"); // Получаем массив  
 ArrayList<Book> books = new ArrayList<>();  
  
 String author, name;  
 long publish;  
 long amount;  
 for (JSONObject obj: arr) {  
 json = JSON.*getJSON*(obj.toString());  
 author = json.get("author").toString();  
 name = (String) json.get("name");  
 publish = (long) json.get("year");  
 amount = (long) json.get("amountIssuance");  
 books.add(new Book(author, name, (int)publish, amount));  
 }  
  
 books.add(new Book());  
 books.add(new Book());  
  
 System.*out*.println("\nDouglas Adams books for 1982: ");  
 Book.*booksByCriterion*(books, "Дуглас Адамс", 1982);  
  
 System.*out*.println("\nVirginia Andrews books for 1979: ");  
 Book.*booksByCriterion*(books, "Вирджиния Эндрюс", 1979);  
  
 System.*out*.println("\nDouglas Adams books: ");  
 Book.*booksByCriterion*(books, "Дуглас Адамс");  
  
 System.*out*.println("\nBooks for 1980: ");  
 Book.*booksByCriterion*(books, 1980);  
 System.*out*.println("\nBooks for 1982: ");  
 Book.*booksByCriterion*(books, 1982);  
  
 System.*out*.println("\nCompare books: ");  
 Book.*compareBooks*(books.get(0), books.get(3));  
  
 Book specialBook = new Book("Михаил Булгаков", "Мастер и Маргарита");  
 books.add(specialBook);  
 System.*out*.println("\nSpecial: \n" + specialBook);  
 specialBook.setPublish(1966);  
 specialBook.setAmountIssuance(80000000);  
 System.*out*.println("Edited special: \n" + specialBook);  
  
 System.*out*.println("All books: \n");  
 Book.*showBooks*(books);  
 }  
}

**Результат роботи програми**





*В консолі було виведено всі книги, які містились в файлі* ***db.json****, а також кастомні об’єкти створені в* ***Main.java****. Також показано роботу різних статичних методів класу Book*

**Також, усі файли проекту можна отримати за посиланням (ветка lab2):**

*https://github.com/steadyGuy/oop\_labs/tree/lab2*

**ВИСНОВОК**

На даній лабораторній роботі була створена програма мовою програмування Java, метою якої було створити клас, предметна область якого вважалась “бібліотека”. Предметно-значуща сутність (сущность) була “книга”. Під час написання класу було використано один із основних принцпипів ООП.

Інкапсуаляція. Усі поля класу, а також деякі методи були приховані всередині самого класу за допомогою модифікатора *private.* Таким чином, дані були захищені від несанкціонованого доступу ззовні (як для отримання значення, так і для його зміни). При цьому, для того, щоб записати й отримати інформацію використовувалися відповідні аксесори (геттери та сеттери). В ці методи було вкладено додаткову логіку (перевірка корректності введених даних).

Поліморфізм. Використання статичного поліморфізму в лаболаторній (перевантаження методів). При створенні екземплярів класу, можуть бути різні вхідні дані, тому клас реалізує конструктор, який є перевантаженим декілька раз (в особливості конструктор з параметрами і без). Було створено статичний перевантажений метод *booksByCriterion*, який приймає різну кількість параметрів від користувача. Також, було спроектовано інтерфейс – *Bookish*, для імплементації класу.

Було створено додаткові методи для роботи з полями книг (наприклад метод *calcAverageIssuance*, *getCurrentYear*), а також різні статичні методи для роботи з самими екземплярами класу (наприклад статичний метод *compareBooks*, який порівнює книги або метод *showBooks*).

Для роботи з книгами у форматі JSON, а також для демонстрації роботи програми було створено спеціальний клас *JSON.java*, а також використано бібліотеку *json-simple*.