|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |
| **Институт информационных технологий** |
| **Кафедра корпоративных информационных систем**  **Лабораторная работа №4**  по дисциплине  Объектно-ориентированное программирование  **Тема лабораторной работы:** Ввод и вывод в Java. Пакет java.io его классы.   |  |  | | --- | --- | | Студент группы ИКБО-08-18 | Валяев Данила Андреевич | |  |  | |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) | |  |  | |  |  | | Работа представлена к защите | «25» ноября 2019 г. | |  |  | |  |  | |
|  |
| **Содержание**  [**Задание 1.** Управление файловой структурой Java – класс File. 3](#_Toc27910572)  [**Задание 2.** Потоки байтов и класс RandomAccessFile 4](#_Toc27910573)  [**Задание 3.** Символьные потоки 6](#_Toc27910574)  [**Задание 4.** Сериализация объектов 6](#_Toc27910575)  [**Заключение** 9](#_Toc27910576)  [**Список используемых источников** 10](#_Toc27910577)  [**Приложение** 11](#_Toc27910578) |

# **Задание 1.** Управление файловой структурой Java – класс File.

Исследовать возможности класса File по созданию файлов

(пустых) и папок программой. Применение конструктора и метода.

-Создать файл в папке приложения с именем MyFile1.txt.

-Проверить появление файла в папке приложения.

-Создать файл с именем MyFile2.txt в корне определенного диска.

-Проверить появление файла в папке.

-Создать файл с именем MyFile3.txt в папке Имя диска\\Имя папки\\Имя

файла.

-Проверить появление файла в папке.

-Создать папку третьего уровня, например, Первая\\Вторая\\Третья.

-Проверить появление папки.

-Все операции заключить в блок try, с обработчиками исключений

IOException, Exception, FileNotFoundException.

Упражнение 2. Получить параметры файлов методами класса File.

В пунктах задания использовать объекты, созданные в задании 1.

-Проверить, что вызывающий объект содержит имя файла, а не папки и

отобразить имя файла, вызывающего объекта и его родительскую папку.

-Проверить, что вызывающий объект содержит имя папки, а не файла и

отобразить имя файла, вызывающего объекта.

-Проверить, что в папке приложения существует файл с именем

MyFile1.txt.

-Отобразить полный путь к файлу или папке объекта.

-Отобразить размер файла или папки объекта, указать единицу

измерения. Прокомментировать вид файла – папка или файл.

Упражнение 3. Модификация файловой структуры приложения средствами

класса File.

1. Добавить в папку приложения еще одну папку.

2. Сформировать массив файлов, находящихся в папке приложения,

используя метод list(). Отобразить содержимое массива.

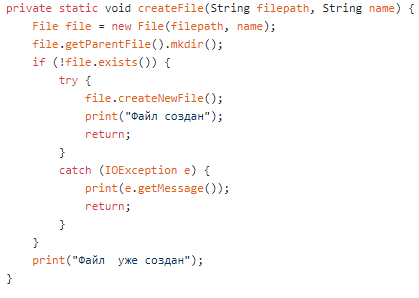
3. Сформировать массив файлов, находящихся в папке приложения,

используя метод listFiles( ). Отобразить содержимое массива.

Определить количество папок, содержащихся в файле приложения.

4. Удалить папки и файлы созданные во всех трех упражнениях

Создание файла:



# **Задание 2.** Потоки байтов и класс RandomAccessFile

1. Создать класс, указанный в варианте. Предусмотреть конструктор с

параметрами для создания объекта, контролирующий поступающие данные.

Данные состояния объекта вводятся с клавиатуры. Дополнительные методы,

указанные в варианте

2. Создать класс по управлению файлом, который должен обеспечить

следующие функции:

3. Формирование файла данных

3.1. Создать выходной поток для записи объекта в файл, располагая файл в

папке приложения. Имя файла пользователь вводит с клавиатуры.

3.2. Обеспечить проверку существования файла с таким именем и вывод

сообщения о его существовании.

3.3. Выполнить запись в файл данных нескольких объектам. Данные для

полей объекта вводятся с клавиатуры и записываются в файл.

Расположите их в файле так, чтобы потом можно было написать

алгоритм чтения из файла данных и формирование объектов..

Помните, что при вводе вы будете вводить данные и строковых и

примитивных типов (смешанный ввод).

3.4. Закрыть выходной поток.

4. Чтение данных

4.1. Создать входной поток для чтения данных из созданного файла.

4.2. Создать массив с данными о состоянии объектов, используя оду из

коллекций Java: для нечетного варианта ArrayList, для четного –

LinkedList.

4.3. Прочитать данные из файла и разместить в коллекции те объекты,

которые, удовлетворяют критериям дополнительного задания варианта

(задание 1).

5. Используя класс RandomAccessFile произвольного доступа к записям файла

обеспечить:

5.1. Создание нового файла из записей фиксированной длины,

содержащий сведения по объектам, прочитанных в массив объектов в

задании 4.3. Так как поток RandomAccessFile в качестве источника

может иметь только массивы примитивного типа, необходимо

предусмотреть структуру записи – строка и ее размер. Для этого

расширяйте строковые значения пробелами справа, так, чтобы в сумме

с примитивными значениями получить нужный размер. При таком

условии метод seek можно использовать для перемещения по записям.

# **Задание 3.** Символьные потоки

Требования: Выполнить три упражнения. Каждое упражнение в отдельном методе основного класса проекта.

Платформа Java хранит символьные значения, используя соглашения Unicode. Символьный потоковый ввод-вывод автоматически преобразовывает этот внутренний формат в и от локального набора символов. В Западных локалях локальный набор символов обычно является 8-разрядным надмножеством ASCII.

Для большинства приложений ввод-вывод с символьными потоками не более сложен чем ввод-вывод с потоками байтов. Ввод и вывод, сделанный с потоковыми классами автоматически, преобразовывает в и от локального набора символов. Программа, которая использует символьные потоки вместо потоков байтов автоматически, адаптируется к локальному набору символов и готова к интернационализации — все без дополнительного усилия программиста.

# **Задание 4.** Сериализация объектов

1. В модели лабораторной работы 2 выберите один класс, объекты

которого требуют больше данных. Создайте текстовый файл с данными для

заполнения нескольких объектов. Данные разного типа лучше располагать на

отдельных строках файла.

2. Разработайте класс ClassTextFile для управления текстовым файлом

исходных данных.

2.1. Поля

Путь к файлу

2.2. Методы

- Конструктор с параметром – путь к файлу, с проверкой

существования файла.

- Заполнить объект данными файла и вернуть его. Для этого

использовать поток Scanner для считывания из текстового файла

данных для заполнения полей объекта. Метод считывания должен

соответствовать типу поля объекта.

3. Разработать класс для представления объекта, вашего варианта

лабораторной работы 2.

3.1. Поля класса

- свойства выбранного объекта.

3.2. Методы

- Ввод данных для объекта с консоли.

- Вернуть объект.

- Залить данные в объект.

4. Разработать класс, обеспечивающий запись в файл и чтение из файла,

используя классы FileInputStream, FileOutputStream, и классы,

обеспечивающие сериализацию объектов ObjectOutputStream (сериализация

объектов) и ObjectInputStream (десериализация объектов).

4.1. Поля класса

- Путь к файлу сериализованных объектов.

- Коллекция объектов.

4.2. Методы:

- Конструктор класса – должен инициализировать путь переданным

параметром.

- Запись одного сериализованного объекта в файл FileOutputStream

посредством потока ObjectOutputStream.

- Чтение одного объекта из файла, используя потоки FileInputStream,

ObjectInputStream.

- Метод get для объекта.

- Создание коллекции объектов, запись всей коллекции в файл.

- Чтение объектов из файла в коллекцию.

- Метод Get для коллекции объектов.

- Вывод содержимого файла и коллекции.

2. Разработать приложение, демонстрирующее выполнение всех операций

над файлами объектов, предусмотренных в задании.

# **Заключение**

В процессе выполнения лабораторной работы мы изучили возможности ввода и вывода в Java и потоки ввода и вывода в Java.

# **Список используемых источников**

1. Символьные потоки [Электронный ресурс]. – URL: <http://spec-zone.ru/RU/Java/Tutorials/essential/io/charstreams.html>
2. Класс File [Электронный ресурс]. – URL: https://metanit.com/java/tutorial/6.11.php

# **Приложение**

Book

package com.company;

public class book {

private String Author;

private String Name;

private String Number;

private String Publisher;

private String Count;

private String Cost;

public book(String author, String name, String number, String publisher, String count, String cost){

setAuthor(author);

setName(name);

setNumber(number);

setPublisher(publisher);

setCount(count);

setCost(cost);

}

public String getAuthor() { return Author; }

public String getName() { return Name; }

public String getNumber() { return Number; }

public String getPublisher() { return Publisher; }

public String getCount() { return Count; }

public String getCost() { return Cost; }

public void setAuthor(String author) { this.Author = author; }

public void setName(String name) { this.Name = name; }

public void setNumber(String number) { this.Number = number; }

public void setPublisher(String publisher) { this.Publisher = publisher; }

public void setCount(String count) { this.Count = count; }

public void setCost(String cost) { this.Cost = cost; }

private String getNumber(String name, String author){

return Number;

}

private String checkBooks(){

if(true)

return "Книги изданы в 1 издании";

else

return "Книги не изданы в 1 издании";

}

private void authorBooks(String author){

print(" ");

}

private void changePublisher(String pub1, String pub2){

print("");

}

private void print(String txt){

System.out.println(txt);

}

}

Serialization

package com.company;

import java.io.\*;

import java.io.File;

import java.util.ArrayList;

import java.util.Scanner;

class Serialization {

String file\_1 = "C:\\Users\\dakfa\\Desktop\\st\\Java\\lab4\_2\\Serialization.bin";

String file\_2 = "C:\\Users\\dakfa\\Desktop\\st\\Java\\lab4\_2\\SerializationColl.bin";

private ArrayList<book> Books = new ArrayList<book>();

Serialization(String file\_1) throws IOException {

try {

if (!new File(file\_1).createNewFile()) {

System.out.println("Файл существует!");

} else {

this.file\_1 = file\_1;

}

} catch(IOException e) {}

}

public void outBook() throws IOException {

Scanner in = new Scanner(System.in);

print("Введите автора: ");

String author = in.nextLine();

print("Введите название: ");

String name = in.nextLine();

print("Введите id: ");

String id = in.nextLine();

print("Введите издание: ");

String pub = in.nextLine();

print("Введите кол-во страниц: ");

String count = in.nextLine();

print("Введите цену: ");

String cost = in.nextLine();

book Book = new book(author, name, id, pub, count, cost);

Book.setAuthor(author);

Book.setName(name);

Book.setNumber(id);

Book.setPublisher(pub);

Book.setCount(count);

Book.setCost(cost);

ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream(file\_1));

out.writeUTF(Book.getAuthor());

out.writeUTF(Book.getName());

out.writeUTF(Book.getNumber());

out.writeUTF(Book.getPublisher());

out.writeUTF(Book.getCount());

out.writeUTF(Book.getCost());

out.close();

}

void inpBook () throws IOException {

ObjectInputStream in = new ObjectInputStream(new FileInputStream(file\_1));

book input = new book(in.readUTF(), in.readUTF(), in.readUTF(), in.readUTF(), in.readUTF(), in.readUTF());

System.out.println("Информация о книге: "+input.getAuthor()+" "+input.getName()+" "+input.getNumber()

+" "+input.getPublisher()+" "+input.getCount()+" "+input.getCost());

in.close();

}

public void addToColl() {

Scanner in = new Scanner(System.in);

print("Кол-во книг: ");

int count = in.nextInt();

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

print("Введите автора: ");

String author = in.nextLine();

print("Введите название: ");

String name = in.nextLine();

print("Введите id: ");

String id = in.nextLine();

print("Введите издание: ");

String pub = in.nextLine();

print("Введите кол-во страниц: ");

String coun = in.nextLine();

print("Введите цену: ");

String cost = in.nextLine();

book bk = new book(author, name, id, pub, coun, cost);

bk.setAuthor(author);

bk.setName(name);

bk.setNumber(id);

bk.setPublisher(pub);

bk.setCount(coun);

bk.setCost(cost);

Books.add(bk);

}

}

public void outColl() throws IOException {

ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream(file\_2));

for (book bk: Books) {

out.writeUTF(bk.getAuthor());

out.writeUTF(bk.getName());

out.writeUTF(bk.getNumber());

out.writeUTF(bk.getPublisher());

out.writeUTF(bk.getCount());

out.writeUTF(bk.getCost());

}

out.close();

}

void inpColl() throws IOException {

ObjectInputStream in = new ObjectInputStream(new FileInputStream(file\_2));

Books.clear();

while(true) {

try {

Books.add(new book(in.readUTF(), in.readUTF(), in.readUTF(), in.readUTF(), in.readUTF(), in.readUTF()));

} catch (EOFException e) {

in.close();

break;

}

}

}

void printColl() {

for (book bk : Books) {

System.out.println(bk.getAuthor()+" "+bk.getName()+" "+bk.getNumber()

+" "+bk.getPublisher()+" "+bk.getCount()+" "+bk.getCost());

}

}

ArrayList<book> getColl() {

return Books;

}

private static void print(String txt){

System.out.println(txt);

}

}

Main

package com.company;

import java.io.File;

import java.io.\*;

import java.nio.charset.Charset;

import java.util.ArrayList;

import java.util.Arrays;

import java.io.IOException;

import java.util.Scanner;

public class Main

{

public static void main(String[] args) {

try{

print("\n1.Исследовать возможности класса File по созданию файлов (пустых) и папок программой. Применение конструктора и метода");

task1\_1();

print("\n2.Получить параметры файлов методами класса File, в пунктах задания использовать объекты, созданные в задании 1");

task1\_2();

print("\n3.Модификация файловой структуры приложения средствами класса File");

task1\_3();

task2();

print("\n1.Чтение из одного файла текстового файла и запись в другой");

task3\_1();

print("\n2.Применение буферизированных потоков для чтения и записи текстовых файлов");

task3\_2();

print("\n3.Настройка кодировки символов для входного и выходного потоков");

task3\_3();

print("Применение классов ObjectOutputStream и ObjectInputStream для сериализации и десериализации объектов ");

task4();

}

catch (Exception e){

print(String.valueOf(e));

}

}

private static void createFile(String filepath, String name) {

File file = new File(filepath, name);

file.getParentFile().mkdir();

if (!file.exists()) {

try {

file.createNewFile();

print("Файл создан");

return;

}

catch (IOException e) {

print(e.getMessage());

return;

}

}

print("Файл уже создан");

}

private static void createDirectory(String filepath) {

File file = new File(filepath);

if (!file.exists()) {

file.mkdirs();

print("Файл создан");

return;

}

print("Файл уже создан");

}

private static void deleteFile(String filepath, String name) {

File file = new File(filepath, name);

if (file.delete()) {

print("Файл " + file.getName() + " удален");

return;

}

print("Ошибка удаления");

}

private static void deleteDirectory(String filepath) {

File file = new File(filepath);

if (file.delete()) {

print("Папка " + file.getName() + " удалена");

return;

}

print("Ошибка удаления");

}

private static String[] getFiles() {

File file = new File(".");

return file.list();

}

private static File[] getFolders() {

File file = new File(".");

return file.listFiles();

}

private static void foldersCount(File[] files) {

int count = 0;

for (File file : files) {

if (file.isDirectory()) {

++count;

}

}

print(String.valueOf(count));

}

private static void task1\_1(){

createFile(".", "MyFile1.txt");

createFile("F:\\", "MyFile2.txt");

createFile("C:\\Steam\\", "MyFile3.txt");

createDirectory("Первая\\Вторая\\Третья");

}

private static void task1\_2() {

File file = new File("C:\\Steam\\", "MyFile4.txt");

File file1 = new File("MyFile1.txt");

String exist = "";

if (file1.exists()) {

exist = "Файл MyFile1.txt существует";

}

print("Существует ли файл MyFile1.txt: " + exist + "\nИмя файла: " + file.getName() + "\nПуть к файлу: " +

file.getAbsolutePath() + "\nВес: " + file.length() + " bytes\nТип файла: " + (file.isDirectory() ? "Папка" : "Файл"));

}

private static void task1\_3(){

createDirectory("one\_more\_folder");

print(Arrays.toString(getFiles()));

File[] files = getFolders();

print(Arrays.toString(files));

foldersCount(files);

deleteFile(".", "MyFile1.txt");

deleteFile("F:\\", "MyFile2.txt");

deleteFile("C:\\Steam\\", "MyFile3.txt");

deleteDirectory("Первая\\Вторая\\Третья");

deleteDirectory("Первая\\Вторая");

deleteDirectory("Первая");

deleteDirectory("one\_more\_folder");

}

private static void task2() throws IOException {

Scanner in = new Scanner(System.in);

String StringBuf = "";

print("Введите название файла: ");

StringBuf = in.nextLine();

StringBuf.trim();

print("Введите кол-во книг: ");

int IntBuf = in.nextInt();

FileManager F = new FileManager(StringBuf);

if(F.exist()){

print("Файл существует");

}else {

print("Файл не существует и будет создан.");

try {

F.create();

} catch (Exception ignored) {}

}

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

print("Вы вводите книгу №" + (i + 1));

print("Введите автора: ");

String author = in.nextLine();

print("Введите название: ");

String name = in.nextLine();

print("Введите id: ");

String id = in.nextLine();

print("Введите издание: ");

String pub = in.nextLine();

print("Введите кол-во страниц: ");

String count = in.nextLine();

print("Введите цену: ");

String cost = in.nextLine();

book Book = new book(author,name,id,pub,count,cost);

try {

F.write(Book);

ArrayList<book> Books = F.GetBooks();

print("Введите автора для поиска книги: ");

author = in.nextLine();

print("Введите название для поиска книги: ");

name = in.nextLine();

print(F.findId(author, name));

print("Введите издание для изменения: ");

pub = in.nextLine();

F.changePublisher(pub);

print("Введите автора для поиска всех книг: ");

author = in.nextLine();

F.oneAuthor(author);

}

catch (IOException ignored){}

}

}

private static void task3\_1(){

try {

Reader reader = new InputStreamReader(new FileInputStream("T1.txt"));

Writer writer = new OutputStreamWriter(new FileOutputStream("T2.txt"));

int x;

while((x = reader.read()) != -1) {

writer.write((char)x);

}

reader.close();

writer.close();

print("Перепись файлов завершена");

}

catch (Exception e){

print(String.valueOf(e));

}

}

private static void task3\_2(){

try {

Writer writer = new OutputStreamWriter(new FileOutputStream("A.txt"));

for (int i = 40; i < 552; i++) {

writer.write((char) i);

}

writer.close();

BufferedReader inb = new BufferedReader(new InputStreamReader(new FileInputStream("A.txt")), 128);

BufferedWriter outb = new BufferedWriter(new OutputStreamWriter(new FileOutputStream("B.txt")), 128);

char[] buf = new char[128];

for(int i = 0; i < 4; i++){

inb.read(buf);

outb.write(buf);

outb.newLine();

}

inb.close();

outb.close();

print("Данные переписаны");

}

catch (Exception e){

print(String.valueOf(e));

}

}

private static void task3\_3(){

try {

BufferedReader in = new BufferedReader(new InputStreamReader(new FileInputStream("A.txt"), "Cp1251"));

print(Charset.defaultCharset().name());

String str = in.readLine();

print(str);

in = new BufferedReader(new InputStreamReader(new FileInputStream("B.txt"), "UTF-8"));

str = in.readLine();

print(str);

}

catch (Exception e){

print(String.valueOf(e));

}

}

private static void task4(){

try {

String file1path = "C:\\Users\\dakfa\\Desktop\\st\\Java\\lab4\_2\\Serialization.bin";

File fileS = new File(file1path);

fileS.createNewFile();

ClassTextFile input = new ClassTextFile("Input.txt");

book fst = input.setBook();

System.out.println("Информация о книге: "+fst.getAuthor()+" "+fst.getName()+" "+fst.getNumber()

+" "+fst.getPublisher()+" "+fst.getCount()+" "+fst.getCost());

Serialization scnd = new Serialization("C:\\Users\\dakfa\\Desktop\\st\\Java\\lab4\_2\\Serialization.bin");

scnd.outBook();

scnd.inpBook();

Serialization trd = new Serialization("C:\\Users\\dakfa\\Desktop\\st\\Java\\lab4\_2\\Serialization.bin");

trd.addToColl();

print("Коллекция после добавления объектов:");

trd.printColl();

print("Коллекция после записи/чтения:");

trd.outColl();

trd.inpColl();

trd.printColl();

} catch (IOException ignored) {}

}

private static void print(String txt){

System.out.println(txt);

}

}

FileManager

package com.company;

import java.io.\*;

import java.io.File;

import java.util.ArrayList;

class FileManager {

private String Name;

private File file;

public FileManager(String Name) throws IOException {

this.Name = Name;

file = new File(this.Name);

}

public void create()throws IOException {

file.createNewFile();

}

public boolean exist(){

return file.canRead();

}

public void write(book book) throws IOException{

if(exist()){

String res = "";

DataInputStream InpStream = new DataInputStream(new FileInputStream(file.getAbsolutePath()));

try{

res = InpStream.readUTF();

}

catch (Exception ignored){}

res = stringBook(book) + "\n" + res;

InpStream.close();

DataOutputStream OutStream = new DataOutputStream(new FileOutputStream(file.getAbsolutePath()));

OutStream.writeUTF(res);

OutStream.flush();

OutStream.close();

}

else{

throw new IOException("Файла не существует");

}

}

private String read() throws IOException{

String res = "";

byte ress[];

if(exist()){

DataInputStream InpStream = new DataInputStream(new FileInputStream(file.getAbsolutePath()));

res = InpStream.readUTF();

System.out.println();

InpStream.close();

}

return res;

}

public String stringBook(book book){

String res = "";

res += book.getAuthor() + ":" + book.getName() + ":" + book.getNumber() + ":" + book.getPublisher() + ":" + book.getCount() + ":" + book.getCost();

return res;

}

public ArrayList<book> GetBooks() throws IOException{

ArrayList<book> ArrBooks = new ArrayList<>();

String[] books = read().split("\n");

for (String book : books) {

//ArrBooks.add(new book(Book));

}

return ArrBooks;

}

public ArrayList<book> oneAuthor(String author) throws IOException{

ArrayList<book> Books = GetBooks();

ArrayList<book> resBooks = new ArrayList<>();

for (com.company.book book : Books) {

if ((book.getAuthor().compareTo(author) == 0)) {

resBooks.add(book);

}

}

return resBooks;

}

public String findId(String author, String name) throws IOException{

String res = "";

String Books = randomAccess();

String[] ArrBooks = Books.split("\n");

for (String arrBook : ArrBooks) {

String[] books = arrBook.split(":");

if (books[0].compareTo(author) == 0 && books[1].compareTo(name) == 0) {

return "Инвентарный номер: " + books[2];

}

}

return res;

}

private String randomAccess() throws IOException{

String nameFile = "RandomAccess.txt";

RandomAccessFile randomFile = new RandomAccessFile(nameFile, "r");

String res = "";

try {

randomFile.seek(0);

String buff;

while ((buff = randomFile.readLine()) != null){

res += buff;

}

}

catch (EOFException ignored){}

return res;

}

public void changePublisher(String publisher) throws IOException{

String Books = randomAccess();

String[] ArrBooks = Books.split("\n");

for (String arrBook : ArrBooks) {

String[] books = arrBook.split(":");

if (books[3].compareTo(publisher) == 0) {

//ArrBooks.setPublisher(publisher);

}

}

}

}

ClassTextFile

package com.company;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.util.Scanner;

public class ClassTextFile {

private String file1 = "Input.txt";

ClassTextFile(String file1) {

try {

if (!new File(file1).createNewFile())

System.out.println("File already exists!");

else

this.file1 = file1;

} catch (IOException ignored) {}

}

book setBook() throws FileNotFoundException {

Scanner in = new Scanner(System.in);

System.out.print("Введите автора: ");

String author = in.nextLine();

System.out.print("Введите название: ");

String name = in.nextLine();

System.out.print("Введите id: ");

String id = in.nextLine();

System.out.print("Введите издание: ");

String pub = in.nextLine();

System.out.print("Введите кол-во страниц: ");

String count = in.nextLine();

System.out.print("Введите цену: ");

String cost = in.nextLine();

return new book(author, name, id, pub, count, cost);

}

}