МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ОТЧЁТ

по лабораторной работе №3

Выполнил:
студент группы ПО-9
Дарашкевич Д.И.

Проверил:
Крощенко А.А.

Брест 2024

Цель работы: научиться создавать и использовать классы в программах на языке программирования Java.

Вариант 4

Задание 1:
Реализовать простой класс.
Требования к выполнению
• Реализовать пользовательский класс по варианту.
• Создать другой класс с методом main, в котором будут находится примеры использования
пользовательского класса.
Для каждого класса
• Создать поля классов
• Создать методы классов
• Добавьте необходимые get и set методы (по необходимости)
• Укажите соответствующие модификаторы видимости
• Добавьте конструкторы
• Переопределить методы toString() и equals()
4) Прямоугольник, заданный длинами двух сторон – Предусмотреть возможность определения площади и периметра, а также логические методы, определяющие, является ли прямоугольник квадратом и существует ли такой прямоугольник. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализовать метод equals, выполняющий сравнение объектов данного типа

Код программы:

Rectangle

```
public class Rectangle {

    private double length;
    private double width;

    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }


    public double calculateArea() {
        return length * width;
    }

    public double calculatePerimeter() {
        return 2 * (length + width);
    }

    public boolean isSquare() {
        return length == width;
    }

    public boolean isRectangleExist() {
        return length > 0 && width > 0;
    }


    public double getLength() {
        return length;
    }
```

```
    public void setLength(double length) {
        this.length = length;
    }

    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {
        this.width = width;
    }

    @Override
    public String toString() {
        return "Rectangle{length=" + length + ", width=" + width + '}';
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null || getClass() != obj.getClass()) {
            return false;
        }
        Rectangle rectangle = (Rectangle) obj;
        return Double.compare(rectangle.length, length) == 0 &&
                Double.compare(rectangle.width, width) == 0;
    }
}
```

Main

```
public class Main {
    public static void main(String[] args) {

        Rectangle rectangle1 = new Rectangle(5, 4);
        Rectangle rectangle2 = new Rectangle(8, 8);

        System.out.println("Прямоугольник 1:");
        System.out.println(rectangle1);
        System.out.println("Прямоугольник 2:");
        System.out.println(rectangle2);

        System.out.println("Площадь прямоугольника: " + rectangle1.calculateArea());
        System.out.println("Периметр прямоугольника: " + rectangle1.calculatePerimeter());
        System.out.println("Площадь прямоугольника: " + rectangle2.calculateArea());
        System.out.println("Периметр прямоугольника: " + rectangle2.calculatePerimeter());

        if (rectangle1.isSquare()) {
            System.out.println("Прямоугольник 1 является квадратом.");
        } else {
            System.out.println("Прямоугольник 1 не является квадратом.");
        }

        if (rectangle2.isSquare()) {
            System.out.println("Прямоугольник 2 является квадратом.");
        } else {
            System.out.println("Прямоугольник 2 не является квадратом.");
        }

        if (rectangle1.isRectangleExist()) {
            System.out.println("Прямоугольник 1 существует.");
        } else {
            System.out.println("Прямоугольник 1 не существует.");
        }
```

```
        if (rectangle2.isRectangleExist()) {
            System.out.println("Прямоугольник 2 существует.");
        } else {
            System.out.println("Прямоугольник 2 не существует.");
        }


        if (rectangle1.equals(rectangle2)) {
            System.out.println("Прямоугольники 1 и 2 равны.");
        } else {
            System.out.println("Прямоугольники 1 и 2 не равны.");
        }
    }
}
```

Входные данные:

Rectangle rectangle1 = new Rectangle(5, 4);

Rectangle rectangle2 = new Rectangle(8, 8);

Результат работы программы:

```
C:\Users\Legion\.jdks\openjdk-22.0.1\bin\j
Прямоугольник 1:
Rectangle{length=5.0, width=4.0}
Прямоугольник 2:
Rectangle{length=8.0, width=8.0}
Площадь прямоугольника: 20.0
Периметр прямоугольника: 18.0
Площадь прямоугольника: 64.0
Периметр прямоугольника: 32.0
Прямоугольник 1 не является квадратом.
Прямоугольник 2 является квадратом.
Прямоугольник 1 существует.
Прямоугольник 2 существует.
Прямоугольники 1 и 2 не равны.

Process finished with exit code 0
```

Задание 2:

Разработать автоматизированную систему на основе некоторой структуры данных, манипулирующей объектами пользовательского класса. Реализовать требуемые функции обработки данных

Требования к выполнению

• Задание посвящено написанию классов, решающих определенную задачу автоматизации;

• Данные для программы загружаются из файла (формат произволен). Файл создать и написать вручную.

4) Автоматизированная система в библиотеке

Составить программу, которая содержит текущую информацию о книгах в библиотеке. Сведения о книгах (Book) содержат:

• номер УДК;

• Фамилию и инициалы автора;

• Название;

• Год издания;
• Количество экземпляров в библиотеке;
• Количество страниц;
• Количество томов;
• ФИО читателя, взявшего книгу (при наличии);
• Срок сдачи книги (если была взята).
Программа должна обеспечивать:
• Формирование общего списка книг;
• Формирование списка книг, старше n лет;
• Формирование списка книг, взятых на чтение;
• Формирование списка книг, взятых на чтение с выводом личной информации о читателях;
• Формирование списка книг, которые задержаны читателем дольше указанного срока.

Код программы:

Book

```
import java.util.Date;
import java.util.HashMap;
class Book {
    private String udcCode;
    private String author;
    private String title;
    private int publicationYear;
    private int pagesCount;
    private int volumesCount;
    private int copiesCount;
    private HashMap<String, Date> borrowedBooks;
    public Book(String udkNumber, String author, String title, int publicationYear,
                int pageCount, int volumesCount, int copiesCount) {
        this.udcCode = udkNumber;
        this.author = author;
        this.title = title;
        this.publicationYear = publicationYear;
        this.pagesCount = pageCount;
        this.volumesCount = volumesCount;
        this.copiesCount = copiesCount;
        this.borrowedBooks = new HashMap<>();
    }
    public String getUdcCode() {
        return udcCode;
    }
    public void setUdcCode(String udcCode) {
        if (udcCode != null && !udcCode.isEmpty()){
            this.udcCode = udcCode;
        }
    }
    public String getAuthor() {
        return author;
    }
    public void setAuthor(String author) {
        if (author != null && !author.isEmpty()){
            this.author = author;
        }
    }
    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        if (title != null && !title.isEmpty()){
            this.title = title;
        }
```

```
}
public int getPublicationYear() {
    return publicationYear;
}
public void setPublicationYear(int publicationYear) {
    if (publicationYear > 0){
        this.publicationYear = publicationYear;
    }
}
public int getPagesCount() {
    return pagesCount;
}
public void setPagesCount(int pagesCount) {
    if (pagesCount > 0){
        this.pagesCount = pagesCount;
    }
}
public int getVolumesCount() {
    return volumesCount;
}
public void setVolumesCount(int volumesCount) {
    if (volumesCount > 0){
        this.volumesCount = volumesCount;
    }
}
public int getCopiesCount() {
    return copiesCount;
}
public void setCopiesCount(int copiesCount) {
    if (copiesCount > 0){
        this.copiesCount = copiesCount;
    }
}
@Override
public String toString() {
    StringBuilder readers = new StringBuilder();
    if (!borrowedBooks.isEmpty()){
        readers.append(", readers( ");
        borrowedBooks.forEach((readerName, deadline) -> {
            readers.append(readerName).append(", ").append(deadline).append(";");
        });
        readers.append(")");
    }
    return "book(" +
            "udkNumber = " + udcCode +
            ", author = " + author +
            ", title = " + title +
            ", publicationYear = " + publicationYear +
            ", pageCount = " + pagesCount +
            ", volumesCount = " + volumesCount +
            ", copiesCount = " + copiesCount +
            readers + ")";
}
public Book borrowBook(String reader, Date deadline){
    if (reader != null && deadline != null && copiesCount >= 1){
        copiesCount--;
        borrowedBooks.put(reader, deadline);
        return this;
    }
    else return null;
}
public void returnBook(String reader){
    borrowedBooks.remove(reader);
}
```

```
    public HashMap<String, Date> getBorrowedBooks(){
        return borrowedBooks;
    }
}
```

Library

```
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.Map;
public class Library {
    private ArrayList<Book> books;
    public Library() {
        books = new ArrayList<>();
    }
    public void addBook(Book book){
        if (!books.contains(book))
            books.add(book);
    }
    public void removeBook(Book book){
        books.remove(book);
    }
    public Book borrowBook(Book book, String reader){
        if (books.contains(book)){
            int bookBorrowingDays = 30;
            Calendar calendar = Calendar.getInstance();
            calendar.setTime(new Date());
            calendar.add(Calendar.DAY_OF_YEAR, bookBorrowingDays);
            return book.borrowBook(reader, calendar.getTime());
        }
        else return null;
    }
    public void returnBookToLibrary(Book book, String reader){
        if (books.contains(book)){
            book.returnBook(reader);
        }
    }
    public ArrayList<Book> getAllBooks() {
        return books;
    }
    public ArrayList<Book> getBooksOlderThan(int year) {
        ArrayList<Book> result = new ArrayList<>();
        for (Book book : books)
            if (book.getPublicationYear() <
                    Calendar.getInstance().get(Calendar.YEAR) - year)
                result.add(book);
        return result;
    }
    public ArrayList<Book> getBooksOnLoan() {
        ArrayList<Book> result = new ArrayList<>();
        for (Book book : books) {
            if (!book.getBorrowedBooks().isEmpty()) {
                result.add(book);
            }
        }
        return result;
    }
    public ArrayList<Book> getOverdueBooks() {
        ArrayList<Book> result = new ArrayList<>();
        Date date = new Date();
        for (Book book : books)
            for (Map.Entry<String, Date> entry :
                    book.getBorrowedBooks().entrySet())
                if (entry.getValue().after(date)){
                    result.add(book);
```

```
                    break;
                }
        return result;
    }
    public Book getFirstBookByTitle(String title){
        for (Book book : books)
            if (book.getTitle().equals(title))
                return book;
        return null;
    }
}
```

Main

```
public class Main {
    public static void main(String[] args) throws Exception {
        Library library = new Library();
        Book book1 = new Book("978-0061120084", "J.K. Rowling",
                "Harry Potter and the Philosopher's Stone",
                2024, 223, 1, 1);
        Book book2 = new Book("978-0439554930", "J.K. Rowling",
                "Harry Potter and the Chamber of Secrets",
                1998, 251, 3, 2);
        Book book3 = new Book("978-0439064866", "J.K. Rowling",
                "Harry Potter and the Prisoner of Azkaban",
                1999, 317, 6, 1);
        Book book4 = new Book("978-0439136365", "J.K. Rowling",
                "Harry Potter and the Goblet of Fire",
                2000, 636, 5, 2);
        Book book5 = new Book("978-0439358071", "J.K. Rowling",
                "Harry Potter and the Order of the Phoenix",
                2003, 870, 2, 1);
        Book book6 = new Book("978-0439784542", "J.K. Rowling",
                "Harry Potter and the Half-Blood Prince",
                2005, 652, 4, 1);
        Book book7 = new Book("978-0545010221", "J.K. Rowling",
                "Harry Potter and the Deathly Hallows",
                2024, 759, 3, 2);
        library.addBook(book1);
        library.addBook(book2);
        library.addBook(book3);
        library.addBook(book4);
        library.addBook(book5);
        library.addBook(book6);
        library.addBook(book7);

        System.out.println("All books:");
        for (Book bk : library.getAllBooks()){
            System.out.println(bk);
        }

        System.out.println("\nBooks on loan:");
        library.borrowBook(book1, "John");
        library.borrowBook(book3, "Emma");
        library.borrowBook(book6, "David");
        for (Book bk : library.getBooksOnLoan()){
            System.out.println(bk);
        }

        System.out.println("\nBooks older than 5 years:");
        for (Book bk : library.getBooksOlderThan(5)){
            System.out.println(bk);
        }
    }
}
```

Входные данные:

```
Book book1 = new Book("978-0061120084", "J.K. Rowling",
                "Harry Potter and the Philosopher's Stone",
                1997, 223, 1, 1);
        Book book2 = new Book("978-0439554930", "J.K. Rowling",
                "Harry Potter and the Chamber of Secrets",
                1998, 251, 3, 2);
        Book book3 = new Book("978-0439064866", "J.K. Rowling",
                "Harry Potter and the Prisoner of Azkaban",
                1999, 317, 6, 1);
        Book book4 = new Book("978-0439136365", "J.K. Rowling",
                "Harry Potter and the Goblet of Fire",
                2000, 636, 5, 2);
        Book book5 = new Book("978-0439358071", "J.K. Rowling",
                "Harry Potter and the Order of the Phoenix",
                2003, 870, 2, 1);
        Book book6 = new Book("978-0439784542", "J.K. Rowling",
                "Harry Potter and the Half-Blood Prince",
                2005, 652, 4, 1);
        Book book7 = new Book("978-0545010221", "J.K. Rowling",
                "Harry Potter and the Deathly Hallows",
                2007, 759, 3, 2);
```

Результат работы программы:

```
C:\Users\Legion\.jdks\openjdk-22.0.1\bin\java.exe "-javaagent:D:\IntelliJ IDEA 2024.1.1\lib\idea_rt.jar=60213:D:\IntelliJ IDEA 2024.1.1\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
 -classpath "C:\Users\Legion\Desktop\6 семестр\СПП\lab3\2\out\production\2" Main
All books:
book(udkNumber = 978-0061120084, author = J.K. Rowling, title = Harry Potter and the Philosopher's Stone, publicationYear = 2024, pageCount = 223, volumesCount = 1, copiesCount = 1)
book(udkNumber = 978-0439554930, author = J.K. Rowling, title = Harry Potter and the Chamber of Secrets, publicationYear = 1998, pageCount = 251, volumesCount = 3, copiesCount = 2)
book(udkNumber = 978-0439064866, author = J.K. Rowling, title = Harry Potter and the Prisoner of Azkaban, publicationYear = 1999, pageCount = 317, volumesCount = 6, copiesCount = 1)
book(udkNumber = 978-0439136365, author = J.K. Rowling, title = Harry Potter and the Goblet of Fire, publicationYear = 2000, pageCount = 636, volumesCount = 5, copiesCount = 2)
book(udkNumber = 978-0439358071, author = J.K. Rowling, title = Harry Potter and the Order of the Phoenix, publicationYear = 2003, pageCount = 870, volumesCount = 2, copiesCount = 1)
book(udkNumber = 978-0439784542, author = J.K. Rowling, title = Harry Potter and the Half-Blood Prince, publicationYear = 2005, pageCount = 652, volumesCount = 4, copiesCount = 1)
book(udkNumber = 978-0545010221, author = J.K. Rowling, title = Harry Potter and the Deathly Hallows, publicationYear = 2024, pageCount = 759, volumesCount = 3, copiesCount = 2)

Books on loan:
book(udkNumber = 978-0061120084, author = J.K. Rowling, title = Harry Potter and the Philosopher's Stone, publicationYear = 2024, pageCount = 223, volumesCount = 1, copiesCount = 0, readers( John, Thu Jun 06 20:16:39 MSK
  2024;))
book(udkNumber = 978-0439064866, author = J.K. Rowling, title = Harry Potter and the Prisoner of Azkaban, publicationYear = 1999, pageCount = 317, volumesCount = 6, copiesCount = 0, readers( Emma, Thu Jun 06 20:16:39 MSK
  2024;))
book(udkNumber = 978-0439784542, author = J.K. Rowling, title = Harry Potter and the Half-Blood Prince, publicationYear = 2005, pageCount = 652, volumesCount = 4, copiesCount = 0, readers( David, Thu Jun 06 20:16:39 MSK
 2024;))

Books older than 5 years:
book(udkNumber = 978-0439554930, author = J.K. Rowling, title = Harry Potter and the Chamber of Secrets, publicationYear = 1998, pageCount = 251, volumesCount = 3, copiesCount = 2)
book(udkNumber = 978-0439064866, author = J.K. Rowling, title = Harry Potter and the Prisoner of Azkaban, publicationYear = 1999, pageCount = 317, volumesCount = 6, copiesCount = 0, readers( Emma, Thu Jun 06 20:16:39 MSK
  2024;))
book(udkNumber = 978-0439136365, author = J.K. Rowling, title = Harry Potter and the Goblet of Fire, publicationYear = 2000, pageCount = 636, volumesCount = 5, copiesCount = 2)
book(udkNumber = 978-0439358071, author = J.K. Rowling, title = Harry Potter and the Order of the Phoenix, publicationYear = 2003, pageCount = 870, volumesCount = 2, copiesCount = 1)
book(udkNumber = 978-0439784542, author = J.K. Rowling, title = Harry Potter and the Half-Blood Prince, publicationYear = 2005, pageCount = 652, volumesCount = 4, copiesCount = 0, readers( David, Thu Jun 06 20:16:39 MSK
 2024;))

Process finished with exit code 0
```

Вывод: в ходе выполнения данной лабораторной работы я научился создавать и использовать классы в программах на языке программирования Java.