



Міністерство освіти і науки України  
Національний технічний університет України  
“Київський політехнічний інститут імені Ігоря Сікорського”  
Факультет інформатики та обчислювальної техніки  
Кафедра інформаційні системи та технологій

Лабораторна робота №5

**Java**

Варіант 3 (103)

Виконав:

Студент групи IA-33  
Ничик О. О.

Перевірив:

Лесик В. О.

**Мета роботи:** створення додатків для роботи з файлами з використанням потоків вводу/виводу, опанування користувачької серіалізації, підключення до інтернет джерел та їх аналіз їх вмісту.

## Хід роботи

### Main.java

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

        Scanner sc = new Scanner(System.in);

        while (true) {
            System.out.println("n==== LAB 5 MENU ===");
            System.out.println("1. Рядок з найбільшою кількістю слів");
            System.out.println("2. Зберегти фігури в файл");
            System.out.println("3. Прочитати фігури з файлу");
            System.out.println("4. Шифрувати файл");
            System.out.println("5. Дешифрувати файл");
            System.out.println("6. Аналіз HTML тегів");
            System.out.println("0. Вихід");

            try {
                System.out.print("Ваш вибір: ");
                int choice = Integer.parseInt(sc.nextLine());

                switch (choice) {

                    case 1:
                        System.out.print("Введіть шлях до файлу: ");
                        String path = sc.nextLine();

                        List<String> lines = FileUtils.readLines(path);
                        System.out.println("Рядок: " +
                            StringUtils.findLineWithMostWords(lines));
                        break;

                    case 2:
                        List<Shape> shapes = Arrays.asList(
                            new Circle(r: 2),
                            new Rectangle(w: 3, h: 4)
                        );

                        System.out.print("Куди зберегти? ");
                        String savePath = sc.nextLine();
                        ShapeStorage.saveShapes(savePath, shapes);
                        break;
                }
            }
        }
    }
}
```

```
case 3:
    System.out.print(s: "Звідки читати? ");
    String loadPath = sc.nextLine();

    List<Shape> loaded = ShapeStorage.loadShapes(loadPath);
    if (loaded != null)
        for (Shape s : loaded)
            System.out.println(s);
    break;

case 4:
    System.out.print(s: "Вхідний файл: ");
    String inEnc = sc.nextLine();

    System.out.print(s: "Вихідний файл: ");
    String outEnc = sc.nextLine();

    System.out.print(s: "Ключ: ");
    int encKey = Integer.parseInt(sc.nextLine());

    try {
        FileInputStream fis = new FileInputStream(inEnc);
        FileOutputStream fos = new FileOutputStream(outEnc);
        EncryptingOutputStream eos = new EncryptingOutputStream(fos, encKey)
    } {
        fis.transferTo(eos);
    }

    System.out.println(x: "Файл зашифровано.");
    break;

case 5:
    System.out.print(s: "Вхідний файл: ");
    String inDec = sc.nextLine();

    System.out.print(s: "Вихідний файл: ");
    String outDec = sc.nextLine();

    System.out.print(s: "Ключ: ");
    int decKey = Integer.parseInt(sc.nextLine());
```

```

        try {
            FileInputStream fis = new FileInputStream(inEnc);
            FileOutputStream fos = new FileOutputStream(outEnc);
            EncryptingOutputStream eos = new EncryptingOutputStream(fos, encKey)
        } {
            fis.transferTo(eos);
        }

        System.out.println(x: "Файл зашифровано.");
        break;
    }

    case 5:
        System.out.print(s: "Вхідний файл: ");
        String inDec = sc.nextLine();

        System.out.print(s: "Вихідний файл: ");
        String outDec = sc.nextLine();

        System.out.print(s: "Ключ: ");
        int decKey = Integer.parseInt(sc.nextLine());

        try {
            FileInputStream fis = new FileInputStream(inDec);
            DecryptingInputStream dis = new DecryptingInputStream(fis, decKey);
            FileOutputStream fos = new FileOutputStream(outDec)
        } {
            dis.transferTo(fos);
        }

        System.out.println(x: "Файл дешифровано.");
        break;

```

```

    case 6:
        System.out.print(s: "URL: ");
        String url = sc.nextLine();

        Map<String, Integer> result = HtmlTagAnalyzer.analyze(url);

        System.out.println(x: "\nСортування за алфавітом:");
        HtmlTagAnalyzer.printSortedAlphabetically(result);

        System.out.println(x: "\nСортування за частотою:");
        HtmlTagAnalyzer.printSortedByCount(result);
        break;

    case 0:
        System.out.println(x: "Вихід.");
        return;

    default:
        System.out.println(x: "Невідомий пункт меню.");
    }

} catch (Exception e) {
    System.out.println("Помилка введення: " + e.getMessage());
}
}
}

```

## DecryptingInputStream.java

```
public class DecryptingInputStream extends FilterInputStream {

    private final int key;

    public DecryptingInputStream(InputStream in, int key) {
        super(in);
        this.key = key;
    }

    @Override
    public int read() throws IOException {
        int b = super.read();
        return (b == -1) ? -1 : b - key;
    }
}
```

## EncryptingOutputStream.java

```
public class EncryptingOutputStream extends FilterOutputStream {

    private final int key;

    public EncryptingOutputStream(OutputStream out, int key) {
        super(out);
        this.key = key;
    }

    @Override
    public void write(int b) throws IOException {
        super.write(b + key);
    }
}
```

## FileUtils.java

```
public class FileUtils {

    public static List<String> readLines(String path) {
        List<String> lines = new ArrayList<>();

        try (BufferedReader br = new BufferedReader(new FileReader(path))) {
            String line;
            while ((line = br.readLine()) != null)
                lines.add(line);

        } catch (IOException e) {
            System.out.println("Помилка читання файлу: " + e.getMessage());
        }

        return lines;
    }
}
```

## ShapeStorage.java

```
public class ShapeStorage {

    public static void saveShapes(String path, List<Shape> shapes) {
        try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(path))) {
            oos.writeObject(shapes);
            System.out.println("Фігури збережено.");
        } catch (IOException e) {
            System.out.println("Помилка запису: " + e.getMessage());
        }
    }

    public static List<Shape> loadShapes(String path) {
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(path))) {
            return (List<Shape>) ois.readObject();
        } catch (Exception e) {
            System.out.println("Помилка читання: " + e.getMessage());
        }
        return null;
    }
}
```

## HtmlTagAnalyzer.java

```
public class HtmlTagAnalyzer {

    public static Map<String, Integer> analyze(String url) {
        Map<String, Integer> freq = new HashMap<>();

        try {
            Document doc = Jsoup.connect(url).get();

            for (Element e : doc.getAllElements()) {
                String tag = e.tagName();
                freq.put(tag, freq.getOrDefault(tag, defaultValue: 0) + 1);
            }
        } catch (Exception e) {
            System.out.println("Помилка завантаження HTML: " + e.getMessage());
        }

        return freq;
    }

    public static void printSortedAlphabetically(Map<String, Integer> map) {
        map.entrySet().stream()
            .sorted(Map.Entry.comparingByKey())
            .forEach(e -> System.out.println(e.getKey() + ": " + e.getValue()));
    }

    public static void printSortedByCount(Map<String, Integer> map) {
        map.entrySet().stream()
            .sorted(Map.Entry.comparingByValue())
            .forEach(e -> System.out.println(e.getKey() + ": " + e.getValue()));
    }
}
```

## Circle.java

```
public class Circle extends Shape {
    private double r;

    public Circle(double r) {
        this.r = r;
    }

    @Override
    public double area() {
        return Math.PI * r * r;
    }

    @Override
    public String toString() {
        return "Circle r=" + r + ", area=" + area();
    }
}
```

## Rectangle.java

```
public class Rectangle extends Shape {
    private double w, h;

    public Rectangle(double w, double h) {
        this.w = w;
        this.h = h;
    }

    @Override
    public double area() {
        return w * h;
    }

    @Override
    public String toString() {
        return "Rectangle " + w + "x" + h + ", area=" + area();
    }
}
```

### Shape.java

```
public abstract class Shape implements Serializable {  
    public abstract double area();  
}
```

### StringUtils.java

```
public class StringUtils {  
  
    public static String findLineWithMostWords(List<String> lines) {  
        String best = "";  
        int max = 0;  
  
        for (String line : lines) {  
            int count = line.trim().split(regex: "\s+").length;  
            if (count > max) {  
                max = count;  
                best = line;  
            }  
        }  
  
        return best;  
    }  
}
```

<https://github.com/iamthegoose/java-fifth-semester/tree/lab5>

Висновок: В ході виконання даної лабораторної роботи я створював додатки для роботи з файлами з використанням потоків вводу/виводу, опанування користувачької серіалізації, підключення до інтернет джерел та їх аналіз їх вмісту.