

Seminar 7. Operatii cu fisiere in Java

1 Partea Teoretica

Timp de lucru estimat: 60 minute

1.1 Introducere in operatii cu fisiere

In Java, operatiile cu fisiere permit citirea si scrierea datelor in fisiere de pe sistemul de fisiere. Aceste operatii sunt esentiale pentru stocarea persistenta a datelor.

1.2 Clasele importante pentru manipularea fisierelor

- **File**: Reprezinta un fisier sau un director din sistemul de fisiere.
- **FileReader** si **FileWriter**: Folosite pentru citirea si scrierea caracterelor in fisiere text.
- **BufferedReader** si **BufferedWriter**: Asigura citirea si scrierea eficienta prin intermediul unui buffer.
- **FileInputStream** si **FileOutputStream**: Folosite pentru citirea si scrierea octetilor in fisiere binare.
- **Scanner**: Poate fi utilizata pentru a citi date dintr-un fisier text.
- **PrintWriter**: Permite scrierea formatata in fisiere text.

1.3 Manipularea fisierelor text

1.3.1 Citirea din fisier folosind FileReader si BufferedReader

```
// Importurile necesare
import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

// Cod pentru citirea din fisier
try {
    FileReader fr = new FileReader("fisier.txt");
```

```

        BufferedReader br = new BufferedReader(fr);
        String linie;
        while ((linie = br.readLine()) != null) {
            System.out.println(linie);
        }
        br.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

```

1.3.2 Scrierea in fisier folosind FileWriter si BufferedWriter

```

// Importurile necesare
import java.io.FileWriter;
import java.io.BufferedWriter;
import java.io.IOException;

// Cod pentru scrierea in fisier
try {
    FileWriter fw = new FileWriter("fisier.txt");
    BufferedWriter bw = new BufferedWriter(fw);
    bw.write("Acesta este un text scris in fisier.");
    bw.newLine();
    bw.close();
} catch (IOException e) {
    e.printStackTrace();
}

```

1.3.3 Utilizarea clasei Scanner pentru citirea din fisier

```

// Importurile necesare
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

// Cod pentru citirea din fisier
try {
    File fisier = new File("fisier.txt");
    Scanner scanner = new Scanner(fisier);
    while (scanner.hasNextLine()) {
        String linie = scanner.nextLine();
        System.out.println(linie);
    }
    scanner.close();
} catch (FileNotFoundException e) {
    e.printStackTrace();
}

```

1.3.4 Utilizarea clasei PrintWriter pentru scrierea in fisier

```
// Importurile necesare
import java.io.PrintWriter;
import java.io.IOException;

// Cod pentru scrierea in fisier
try {
    PrintWriter pw = new PrintWriter("fisier.txt");
    pw.println("Scrierea unei linii in fisier.");
    pw.close();
} catch (IOException e) {
    e.printStackTrace();
}
```

1.4 Manipularea fisierelor binare

1.4.1 Citirea din fisier folosind FileInputStream

```
// Importurile necesare
import java.io.FileInputStream;
import java.io.IOException;

// Cod pentru citirea din fisier
try {
    FileInputStream fis = new FileInputStream("fisier.bin");
    int byteCitit;
    while ((byteCitit = fis.read()) != -1) {
        // Procesarea octetilor cititi
        System.out.print((char) byteCitit);
    }
    fis.close();
} catch (IOException e) {
    e.printStackTrace();
}
```

1.4.2 Scrierea in fisier folosind FileOutputStream

```
// Importurile necesare
import java.io.FileOutputStream;
import java.io.IOException;

// Cod pentru scrierea in fisier
try {
    FileOutputStream fos = new FileOutputStream("fisier.bin");
    String data = "Date binare de scris in fisier.";
}
```

```

        fos.write(data.getBytes());
        fos.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

```

1.5 Tratatamentul exceptiilor

Operatiile cu fisiere pot genera exceptii, cum ar fi `FileNotFoundException` sau `IOException`. Este important sa tratam aceste exceptii folosind blocurile `try-catch` pentru a evita erorile in timpul rularii programului.

2 Partea Practica

Timp de lucru estimat: 90 minute

2.1 Probleme Rezolvate

2.1.1 Problema 1: Citirea si afisarea continutului unui fisier text

Enunt: Scrieti un program care citeste continutul unui fisier text numit "input.txt" si afiseaza continutul pe ecran.

Cod Java:

```

import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

public class CitireFisierText {
    public static void main(String[] args) {
        try {
            FileReader fr = new FileReader("input.txt");
            BufferedReader br = new BufferedReader(fr);
            String linie;
            while ((linie = br.readLine()) != null) {
                System.out.println(linie);
            }
            br.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

2.1.2 Problema 2: Scrierea unor date introduse de utilizator intr-un fisier

Enunt: Scrieti un program care cere utilizatorului sa introduca linii de text de la tastatura si le scrie intr-un fisier numit "output.txt". Programul se opreste cand utilizatorul introduce "stop".

Cod Java:

```
import java.io.FileWriter;
import java.io.BufferedWriter;
import java.io.IOException;
import java.util.Scanner;

public class ScriereFisierText {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            FileWriter fw = new FileWriter("output.txt");
            BufferedWriter bw = new BufferedWriter(fw);
            System.out.println("Introduceti textul (scrieti 'stop'
                pentru a termina):");
            while (true) {
                String linie = scanner.nextLine();
                if (linie.equalsIgnoreCase("stop")) {
                    break;
                }
                bw.write(linie);
                bw.newLine();
            }
            bw.close();
            System.out.println("Datele au fost scrise in fisier.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

2.1.3 Problema 3: Copierea continutului unui fisier in alt fisier

Enunt: Scrieti un program care citeste continutul unui fisier numit "sursa.txt" si il copiaza intr-un fisier numit "destinatie.txt".

Cod Java:

```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
```

```

public class CopiereFisier {
    public static void main(String[] args) {
        try {
            FileReader fr = new FileReader("sursa.txt");
            BufferedReader br = new BufferedReader(fr);
            FileWriter fw = new FileWriter("destinatie.txt");
            BufferedWriter bw = new BufferedWriter(fw);
            String linie;
            while ((linie = br.readLine()) != null) {
                bw.write(linie);
                bw.newLine();
            }
            br.close();
            bw.close();
            System.out.println("Copierea s-a realizat cu succes.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

2.1.4 Problema 4: Numararea cuvintelor dintr-un fisier text

Enunt: Scrieti un program care citeste un fisier text numit "text.txt" si afiseaza numarul total de cuvinte din fisier.

Cod Java:

```

import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class NumarareCuvinte {
    public static void main(String[] args) {
        try {
            File fisier = new File("text.txt");
            Scanner scanner = new Scanner(fisier);
            int numarCuvinte = 0;
            while (scanner.hasNext()) {
                scanner.next();
                numarCuvinte++;
            }
            scanner.close();
            System.out.println("Numarul total de cuvinte este: " +
                numarCuvinte);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
}

```

```
}
```

2.1.5 Problema 5: Citirea numerelor dintr-un fisier si calcularea sumei

Enunt: Un fisier numit "numere.txt" contine pe fiecare linie un numar intreg. Scrieti un program care citeste numerele si calculeaza suma lor.

Cod Java:

```
import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

public class SumaNumerelorDinFisier {
    public static void main(String[] args) {
        try {
            FileReader fr = new FileReader("numere.txt");
            BufferedReader br = new BufferedReader(fr);
            String linie;
            int suma = 0;
            while ((linie = br.readLine()) != null) {
                int numar = Integer.parseInt(linie.trim());
                suma += numar;
            }
            br.close();
            System.out.println("Suma numerelor este: " + suma);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

2.2 Probleme Propuse

1. Scrieti un program care citeste un fisier text si afiseaza numarul de linii, numarul de cuvinte si numarul de caractere din fisier.

Solutie partiala:

- Folositi `BufferedReader` pentru a citi linie cu linie din fisier.
- Pentru fiecare linie, incrementati numarul de linii si adaugati la numarul total de caractere lungimea liniei.
- Folositi `String.split(" ")` pentru a separa cuvintele si a incrementa numarul de cuvinte.

Cod Java de exemplu:

```

import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

public class StatisticiFisierText {
    public static void main(String[] args) {
        String numeFisier = "fisier.txt";
        int numarLinii = 0;
        int numarCuvinte = 0;
        int numarCaractere = 0;

        try {
            FileReader fr = new FileReader(numeFisier);
            BufferedReader br = new BufferedReader(fr);
            String linie;

            while ((linie = br.readLine()) != null) {
                numarLinii++;

                // caractere (fara caracterele de linie noua)
                numarCaractere += linie.length();

                // cuvinte (separare dupa spatiu simplu)
                String[] cuvinte = linie.trim().split("\\s+");
                if (!linie.trim().isEmpty()) {
                    numarCuvinte += cuvinte.length;
                }
            }

            br.close();

            System.out.println("Numar linii: " + numarLinii);
            System.out.println("Numar cuvinte: " + numarCuvinte);
            System.out.println("Numar caractere: " +
                numarCaractere);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

2. Realizati un program care copiaza continutul unui fisier binar intr-un alt fisier binar.
Solutie partiala:

- Utilizati FileInputStream si FileOutputStream.
- Cititi octet cu octet din fisierul sursa si scrieti-l in fisierul destinatie.

Cod Java de exemplu:


```

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class CopiereFisierBinar {
    public static void main(String[] args) {
        String fisierSursa = "sursa.bin";
        String fisierDestinatie = "destinatie.bin";

        try {
            FileInputStream fis = new FileInputStream(fisierSursa);
            FileOutputStream fos = new
                FileOutputStream(fisierDestinatie);

            byte[] buffer = new byte[1024];
            int bytesCititi;

            while ((bytesCititi = fis.read(buffer)) != -1) {
                fos.write(buffer, 0, bytesCititi);
            }

            fis.close();
            fos.close();

            System.out.println("Copierea fisierului binar s-a
                realizat cu succes.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

3. Scrieti un program care citeste un fisier text si inlocuieste toate aparitiile unui cuvant dat cu un alt cuvant, salvand rezultatul intr-un fisier nou.

Solutie partiala:

- Cititi continutul fisierului intr-un `StringBuilder`.
- Folositi metoda `replaceAll()` pentru a inlocui cuvantul.
- Scrieti continutul modificat intr-un nou fisier.

Cod Java de exemplu:

```

import java.io.FileReader;
import java.io.BufferedReader;
import java.io.FileWriter;
import java.io.BufferedWriter;
import java.io.IOException;

```

```

public class InlocuireCuvantInFisier {
    public static void main(String[] args) {
        String fisierSursa = "input.txt";
        String fisierDestinatie = "rezultat.txt";
        String cuvantVechi = "vechi";
        String cuvantNou = "nou";

        StringBuilder continut = new StringBuilder();

        try {
            FileReader fr = new FileReader(fisierSursa);
            BufferedReader br = new BufferedReader(fr);
            String linie;

            while ((linie = br.readLine()) != null) {
                continut.append(linie).append("\n");
            }
            br.close();

            String textModificat =
                continut.toString().replaceAll(cuvantVechi,
                    cuvantNou);

            FileWriter fw = new FileWriter(fisierDestinatie);
            BufferedWriter bw = new BufferedWriter(fw);
            bw.write(textModificat);
            bw.close();

            System.out.println("Inlocuirea s-a realizat cu
                succes.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

4. Creati un program care citeste un fisier text si determina frecventa fiecarui cuvant din fisier.

Solutie partiala:

- Utilizati o structura de date precum `HashMap<String, Integer>` pentru a tine evidenta frecventei cuvintelor.
- Pentru fiecare cuvant citit, incrementati contorul din map.

Cod Java de exemplu:

```
import java.io.File;
```

```

import java.io.FileNotFoundException;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class FrecventaCuvinte {
    public static void main(String[] args) {
        String numeFisier = "text.txt";
        Map<String, Integer> frecvente = new HashMap<String,
            Integer>();

        try {
            File fisier = new File(numeFisier);
            Scanner scanner = new Scanner(fisier);

            while (scanner.hasNext()) {
                String cuvant = scanner.next().toLowerCase();
                cuvant = cuvant.replaceAll("[^a-z0-9]", "");
                if (cuvant.isEmpty()) {
                    continue;
                }
                if (frecvente.containsKey(cuvant)) {
                    frecvente.put(cuvant, frecvente.get(cuvant) +
                        1);
                } else {
                    frecvente.put(cuvant, 1);
                }
            }

            scanner.close();

            for (Map.Entry<String, Integer> entry :
                frecvente.entrySet()) {
                System.out.println(entry.getKey() + " : " +
                    entry.getValue());
            }
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
}

```

5. Scrieti un program care citeste numere reale dintr-un fisier si calculeaza media aritmetica a acestora.

Solutie partiala:

- Cititi fiecare numar si adaugati-l la suma totala.
- Tineti evidenta numarului de elemente pentru a calcula media.

Cod Java de exemplu:

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class MediaNumerelorReale {
    public static void main(String[] args) {
        String numeFisier = "numereReale.txt";
        double suma = 0.0;
        int count = 0;

        try {
            File fisier = new File(numeFisier);
            Scanner scanner = new Scanner(fisier);

            while (scanner.hasNextDouble()) {
                double valoare = scanner.nextDouble();
                suma += valoare;
                count++;
            }

            scanner.close();

            if (count > 0) {
                double media = suma / count;
                System.out.println("Media aritmetica este: " +
                    media);
            } else {
                System.out.println("Nu au fost gasite numere in
                    fisier.");
            }
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

6. Realizati un program care citeste un fisier text si afiseaza doar liniile care contin un anumit cuvant dat de la tastatura.

Solutie partiala:

- Cititi linie cu linie din fisier.
- Folositi metoda `contains()` pentru a verifica prezenta cuvantului in linie.

Cod Java de exemplu:

```
import java.io.FileReader;
import java.io.BufferedReader;
```

```

import java.io.IOException;
import java.util.Scanner;

public class LiniiCuCuvantDat {
    public static void main(String[] args) {
        String numeFisier = "fisier.txt";
        Scanner scanner = new Scanner(System.in);

        System.out.print("Introduceti cuvantul cautat: ");
        String cuvant = scanner.nextLine();

        try {
            FileReader fr = new FileReader(numeFisier);
            BufferedReader br = new BufferedReader(fr);
            String linie;
            int nrLinie = 0;

            while ((linie = br.readLine()) != null) {
                nrLinie++;
                if (linie.contains(cuvant)) {
                    System.out.println("Linia " + nrLinie + ": " +
                        linie);
                }
            }

            br.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

7. Scrieti un program care imbină două fișiere text într-un singur fișier, alternând liniile din fiecare fișier.

Soluție parțială:

- Cititi simultan din ambele fișiere folosind doi `BufferedReader`.
- Scrieti alternativ câte o linie din fiecare fișier în fișierul rezultat.

Cod Java de exemplu:

```

import java.io.FileReader;
import java.io.BufferedReader;
import java.io.FileWriter;
import java.io.BufferedWriter;
import java.io.IOException;

public class ImbinareFișiere {

```

```

public static void main(String[] args) {
    String fisier1 = "fisier1.txt";
    String fisier2 = "fisier2.txt";
    String fisierRezultat = "rezultat.txt";

    try {
        FileReader fr1 = new FileReader(fisier1);
        FileReader fr2 = new FileReader(fisier2);
        BufferedReader br1 = new BufferedReader(fr1);
        BufferedReader br2 = new BufferedReader(fr2);

        FileWriter fw = new FileWriter(fisierRezultat);
        BufferedWriter bw = new BufferedWriter(fw);

        String linie1, linie2;
        boolean continua = true;

        while (continua) {
            linie1 = br1.readLine();
            linie2 = br2.readLine();

            if (linie1 == null && linie2 == null) {
                continua = false;
            } else {
                if (linie1 != null) {
                    bw.write(linie1);
                    bw.newLine();
                }
                if (linie2 != null) {
                    bw.write(linie2);
                    bw.newLine();
                }
            }
        }

        br1.close();
        br2.close();
        bw.close();

        System.out.println("Imbinarea fisierelor s-a realizat
            cu succes.");
    } catch (IOException e) {
        e.printStackTrace();
    }
}
}

```

8. Creati un program care citeste un fisier si afiseaza continutul acestuia in ordine inversa

(de la ultima linie la prima).

Solutie partiala:

- Cititi toate liniile si stocati-le intr-un vector.
- Parcurgeti lista in ordine inversa si afisati fiecare linie.

Cod Java de exemplu:

```
import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class AfisareInversa {
    public static void main(String[] args) {
        String numeFisier = "fisier.txt";
        List<String> linii = new ArrayList<String>();

        try {
            FileReader fr = new FileReader(numeFisier);
            BufferedReader br = new BufferedReader(fr);
            String linie;

            while ((linie = br.readLine()) != null) {
                linii.add(linie);
            }
            br.close();

            for (int i = linii.size() - 1; i >= 0; i--) {
                System.out.println(linii.get(i));
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

9. Scrieti un program care citeste un fisier binar si afiseaza continutul acestuia in format hexadecimal.

Solutie partiala:

- Cititi octet cu octet din fisier.
- Folositi `String.format("%02X", byte)` pentru a afisa in format hexadecimal.

Cod Java de exemplu:

```

import java.io.FileInputStream;
import java.io.IOException;

public class AfisareHex {
    public static void main(String[] args) {
        String numeFisier = "fisier.bin";

        try {
            FileInputStream fis = new FileInputStream(numeFisier);
            int b;
            while ((b = fis.read()) != -1) {
                System.out.print(String.format("%02X ", b));
            }
            fis.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

10. Realizati un program care verifica daca un fisier text este palindrom (continutul sau citit de la inceput la sfarsit este acelasi cu continutul citit de la sfarsit la inceput).

Solutie partiala:

- Cititi intreg continutul fisierului intr-un `String`.
- Eliminati spatiile si semnele de punctuatie pentru o comparatie corecta.
- Comparati sirul cu inversul sau folosind `StringBuilder.reverse()`.

Cod Java de exemplu:

```

import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

public class PalindromFisierText {
    public static void main(String[] args) {
        String numeFisier = "palindrom.txt";
        StringBuilder continut = new StringBuilder();

        try {
            FileReader fr = new FileReader(numeFisier);
            BufferedReader br = new BufferedReader(fr);
            String linie;

            while ((linie = br.readLine()) != null) {
                continut.append(linie);
            }
        }
    }
}

```



```

        br.close();

        String text = continut.toString().toLowerCase();
        text = text.replaceAll("[^a-z0-9]", "");

        String invers = new
            StringBuilder(text).reverse().toString();

        if (text.equals(invers)) {
            System.out.println("Fisierul este palindrom.");
        } else {
            System.out.println("Fisierul NU este palindrom.");
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}

```