Fájlkezelés

A program és a háttértár közötti kommunikációt (I/O műveleteket) a Java adatfolyamokon, (Streamen) keresztül valósítja meg. A fizikai adatmozgás mindig a memória (puffer) és a program között zajlik. A puffer létrehozása, feltöltése, ürítése és bezárása az operációs rendszer feladata (DMA). Amíg a fájlunk nyitva van, addig annak az elérése más programból korlátozott lehet.
Fájlkezelés során különböző hibák (kivételek) fordulhatnak elő, amelyek lekezelését meg kell oldanunk.
A fájlkezeléssel és hibakezeléssel kapcsolatos osztályok a java.io package-ben találhatóak.
Az állományok kezelése három részből áll: megnyitás, műveletek, lezárás.
A fájl megnyitásakor, egy csatornaobjektumot (vagy fájlobjektumot) hozunk létre, ahol megadjuk a fájl nevét a teljes elérési úttal együtt. Az elérési út elhagyása esetén a projekt főkönyvtárát használja a Java.

1. Karakteres fájlkezelés

Karakteres állományok írására a *FileWriter* osztály *write()* metódusát használjuk, ebben az esetben az *IOException* (általános I/O hiba) osztályt használata kötelező.
Az ilyen állományok olvasására a *FileReader* osztály *read()* metódusát használjuk, itt is szükséges az I*OException* osztály.

Példa:

Írjunk programot, amely egy string típusú változóban tárolt szöveget karakterenként kiír egy szöveges állományba. Majd az állomány tartalmát olvassuk vissza és írjuk ki a képernyőre.

Megbeszélés:

1. Fájlkezelés, szöveges fájlok, fájlok megnyitása, lezárása.
2. Kivételkezelés.
3. Sztringből egy karakter kivétele a *charAt()* metódus.
4. A *write()* metódus.
5. A *read()* metódus (egész típus, fájlvég = -1).
6. A „*try { try (FileWriter ki…*” egy a Java 7-től bevezetett megoldás, automatikusan gondoskodik a fájl lezárásáról.

Megoldás: 1

```
package file_char;
        import static java.lang.System.out;
        import static java.lang.System.err;
        import java.io.FileWriter;
        import java.io.FileReader;
        import java.io.IOException;
        public class File_char {
                public static void main(String[] args) {
                String szoveg = "Kipróbáljuk az ékezetes betűket:
                                \nÁáÉéÍíÓóÖöŐőÚúÜüŰű\n";
                try { try (FileWriter ki = new FileWriter("fájl.txt")) { //Fájl megnyitás)
                        int i = 0;
                        while(i < szoveg.length()) {
                        int c = szoveg.charAt(i++);             // A sztring i. karaktere
                        ki.write(c);                            // Egy karakter kiírása
```

```
                    }
                    ki.close();                    //Fájl lezárás
                    }
                }
                catch (IOException error){                    //Általános I/O hiba
                    err.println(error.getMessage());    //Hiba kiírása
                }
        try { try (FileReader be = new FileReader("fájl.txt")){  // Fájl megnyitás
                int c;
                while ((c = be.read()) != -1) {          //Egy karakter olvasás, -1 végjelig
                    out.print((char)c);          //Karakter kiírás a képernyőre
                }
                be.close();                    //Fájl lezárás
                }
        }
        catch (IOException error){              //Általános I/O hiba
                err.println(error.getMessage());    //Hiba kiírása
        }
        }
};
```

Feladat:

Olvassunk be a billentyűzetről neveket, amelyeket fájlba írunk, a kiírást az üres ENTER-rel fejezzük be. Ezután írjuk vissza a fájl tartalmát a képernyőre.

Megoldás:

```
package file_line_i.o;
import java.io.File;
import java.io.IOException;
import java.io.PrintStream;
import static java.lang.System.out;
import java.util.Scanner;
public class File_Line_IO {
	public static void main(String[] args) {
		String nev;
		Scanner billentyu = new Scanner (System.in, "ISO8859_2");
		try{ try (PrintStream file = new PrintStream("Fájl.txt"))
			{
			out.print("Kérek egy nevet: ");
			nev = billentyu.nextLine();
			while(nev.length()!=0){
				file.println(nev);
				out.print("Kérek egy nevet: ");
				nev = billentyu.nextLine();
			}
		}}
		catch (IOException error) {
			System.err.println("Hiba: " + error.getMessage());
		}
		try (Scanner file = new Scanner(new File("Fájl.txt"))) {
			while(file.hasNext()) {     // Van még adat?
				out.println(file.nextLine());
			}
			file.close();
		}
		catch (IOException error) {
			System.err.println("Hiba: " + error.getMessage());
		}
	}
}
```

Hallgatói feladat:

Egy tetszőleges szöveges fájl tartalmát írják ki a képernyőre. Vegyék figyelembe, hogy a Java alapértelmezésben UTF-8 kódolást használ.

Megoldás:

```
package file_char_1;
import java.io.FileReader;
import java.io.IOException;
import static java.lang.System.err;
import static java.lang.System.out;
public class File_char_1 {
    public static void main(String[] args) {
    try {
    try (FileReader be = new FileReader("fájl.txt")) {
        int c;
        out.println(be.getEncoding());      //Kiírjuk a fájl kódolását.
        while ((c = be.read()) != -1) {
            out.print((char)c);
        }
        be.close();
    }}
    catch (IOException error){
        err.println(error.getMessage());
    }
}
}
```

Egy hagyományos megoldás:

```
package file_line;
import java.io.File;
import java.io.IOException;
import java.util.Scanner;
public class File_Line {
    public static void main(String[] args) {
        try (Scanner out = new Scanner(new File("Fájl.txt"))) {
            while(out.hasNext()) {                  // Van még adat?
                System.out.println(out.nextLine());
            }
            out.close();
        }
        catch (IOException error) {
            System.err.println("Hiba: " + error.getMessage());
        }
    }
}
```

Példa:

Írjunk egy raktárnyilvántartó programot, amely segítségével szöveges fájlba tetszőleges mennyiségű adatot írunk ki. Adatok: egész szám (pl. raktári szám), szöveg (pl. árú neve). Az adatokat addig olvassuk, amig a számra nullát nem adunk. Ne használjunk ékezetes betűket!

Megbeszélés:

1. A nem alapértelmezett könyvtár használata.
2. Kivételkezelés.

Megoldás (kivétel kezelés nélkül):

```
package file_1_out;
import java.util.Scanner;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.io.PrintWriter;
public class File_1_out {
	public static void main(String[] args) throws IOException {
		Scanner billentyu = new Scanner (System.in);
		PrintWriter out = new PrintWriter("c:\\a\\text.txt");
		int i;
		String nev;
		System.out.print("kérek egy azonosítót: ");
		i=billentyu.nextInt();
		while (i != 0){
			System.out.print("kérek egy nevet: ");
			nev=billentyu.next();
			out.printf("%d - %s\n",i,nev);
			System.out.print("kérek egy azonosítót: ");
			i=billentyu.nextInt();
		}
		out.close();
		BufferedReader in = new BufferedReader(new FileReader("c:\\a\\text.txt"));
		String sor;
		while ((sor = in.readLine()) != null) {
			System.out.println(sor);
		}
		in.close();
	}
}
```

Megoldás (lekezeljük a kivételt):

```
package file_1_out;
import java.util.Scanner;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.io.PrintWriter;
public class File_1_out {
	public static void main(String[] args){
		Scanner billentyu = new Scanner (System.in);
		int i;
		String nev;
		String sor;
		try {
			PrintWriter out = new PrintWriter("c:text.txt");
			System.out.print("kérek egy azonosítót: ");
			i=billentyu.nextInt();
			while (i != 0){
				System.out.print("kérek egy nevet: ");
				nev=billentyu.next();
				out.printf("%d - %s\n",i,nev);
				System.out.print("kérek egy azonosítót: ");
				i=billentyu.nextInt();
			}
			out.close();
		}
		catch (IOException error) {
			System.err.println(error.getMessage());
		}
		try {
		BufferedReader in = new BufferedReader(new FileReader("c:text.txt"));
		while ((sor = in.readLine()) != null) {
				System.out.println(sor);
		}
		in.close();
		}
		catch (IOException error) {
			System.err.println(error.getMessage());
		}
	}
}
```

Feladatok:

1. Írjon programot, amely metódus segítségével beolvassa e program forrásállományát (.java fájl) és meghatározza a betű karakterek számát. A fájl megnyitását a main metódusban végezze el. A metódus visszaadott értéke a karakterek száma legyen, amit a main ír ki.

2. Írjon programot, amely metódus segítségével beolvassa e program forrásállományát (.java fájl) és közben megszámolja a sorokat. A fájl megnyitását a main metódusban végezze el. A metódus visszaadott értéke a sorok száma legyen! Az eredményt, a függvényértéknek megfelelően a main írja ki.

3. Írjon programot, amelyben a main metódus soronként beolvassa e program forrásállományát (.java fájl), egy metódus segítségével megszámolja az egy sorban lévő karakterek számát. A metódus paramétere a sort tartalmazó string, a visszaadott érték a karakterek száma! Az eredményeket, a függvényértékeknek megfelelően a main írja ki.

4. Készítsen programot, amelyben a main metódus sorokat olvas be a billentyűzetről, majd egy metódus segítségével kiírja azokat sorszámmal ellátva egy szöveges állományba (a metódus paraméterei: fájlobjektum, sorszám és a sort tartalmazó string). Az adatmegadás végét szabadon választhatja meg.

5. Készítsen programot, amely egy metódus segítségével sorokat olvas be a billentyűzetről (a metódus paramétere a sort tartalmazó string), majd egy másik metódus segítségével kiírja azokat egy szöveges állományba (a függvény paraméterei: fájlobjektum és a sort tartalmazó string). Az adatmegadás végét szabadon választhatja meg.

6. Írjon programot, amelyben a main metódus beolvassa e program forrásállományát soronként, egy metódus segítségével megszámolja a sorban lévő szóközök számát. A metódus paramétere a sort tartalmazó string, a visszaadott érték a szóközök száma! Az eredményt, a függvényértéknek megfelelően a main metódus írja ki, amely tartalmazza a sorszámot és az adott sorban található szóközök számát.

7. Írjon programot, amelyben a main metódus beolvas a billentyűzetről valós számokat (valamilyen végjelig), és azokat egy szöveges fájlban tárolja. Majd egy metódus segítségével visszaolvassa a fájl tartalmát és kiszámítja a tárolt értékek átlagát. A metódus paramétere a fájlobjektum, a visszaadott érték az átlag, amit a main ír ki.

8. Írjon programot, amelyben a main metódus beolvassa e program forrásállományát soronként, egy metódus segítségével megszámolja a sorban lévő számkarakterek számát. A metódus paramétere a sort tartalmazó string, a visszaadott érték a karakterek száma, amit sorszámmal együtt a main kiír.

9. Készítsen programot, amely a main metódus segítségével sorokat olvas be a billentyűzetről (tetszőleges végjelig), majd meghatározza a sorok hosszát (a sorokban lévő karakterek számát), egy metódus segítségével kiírja a sorokat a sorok hosszával együtt egy szöveges állományba (a metódus paraméterei: fájlobjektum, a sort tartalmazó sting és a sorhossz).

10. Készítsen programot, amely a main metódusban, egymás után bekéri több téglatest három méretét (tetszőleges végjelig). Metódus segítségével számítsa ki a téglatestek térfogatát (a metódus paraméterei a három méret, a visszaadott érték a térfogat). A téglatestek méreteit és a térfogatát a main kiírja egy szöveges állományba.