PROGRAMOWANIE OBIEKTOWE JAVA – LABORATORIUM

OPERACJE NA PLIKACH W JĘZYKU JAVA

Popularnymi operacjami wykonywanymi na plikach są odczyt z pliku oraz zapis do pliku. Czynności te wiążą się z otwarciem połączenia pomiędzy naszym programem a plikiem. Połączenie to nazywamy strumieniem. To właśnie strumieniem będą płynąć dane z pliku do programu (podczas odczytu) oraz z programu do pliku (podczas zapisu).

Klasy do obsługi operacji na plikach

```
//oczyt File odczyt danych z pliku
Scanner odczyt = new Scanner(new File("nazwa_pliku.txt"));
//zapis File
PrintWriter zapis = new PrintWriter("nazwa_pliku.txt");
//odczyt BufferedReader
new BufferedReader(new FileReader("nazwa_pliku.txt"));
```

Przykład 1. TextFile

```
package PO_UR.Lab11;

import java.io.*;
import java.util.Scanner;

public class TextFile {

    //oczyt File
    //odczyt danych z pliku
    //Scanner odczyt = new Scanner(new File("nazwa_pliku.txt"));
    public static void ReadFile() throws IOException {
        //tworzenie obiektu do przechowywywania danych w pliku
        //dostęp do pliku znak po znaku
        File file = new File("ala.txt");
        Scanner in = new Scanner(file);

        String zdanie = in.nextLine();
        System.out.println(zdanie);
    }

    //zapis File
    //PrintWriter zapis = new PrintWriter("nazwa_pliku.txt");
    public static void SaveFile() throws IOException{

        PrintWriter zapis = new PrintWriter("ala.txt");
        zapis.println("Ala ma kota, a kot ma Alę");
        zapis.close(); //zamkniecie strumienia
    }

    //zapis FileWriter
    public static void SaveFileWriter() throws IOException {
        //inicjalizacja zmiennych
        String filePath = "file.txt";
        int number = 123;
        FileWriter fileWriter = null; //klasa dpowiedzialna za zapis do
pliku tekstowego.

        try {
            fileWriter = new FileWriter(filePath); //tworzenie instancji
oraz przekazanie ściezki do pliku
            fileWriter.write(Integer.toString(number)); // zapis do pliku
```

```
wartości tekstowej liczby number
        } finally { //blok spr czy nasz filewriter zosta zainicjalizowany
            if (fileWriter != null) {
                fileWriter.close();
            }
        }
    }

    //odczyt BufferedReader //dostep do pliku linijka po linijce
    // new BufferedReader(new FileReader("ala.txt"));
    public static int ReadBufferFile() throws IOException {
        //inicjalizacja zmiennych
        String filePath = "file.txt";
        int number = 0;
        BufferedReader fileReader = null;

        try {
            fileReader = new BufferedReader(new FileReader(filePath));
//tworzenie istancji klasy
            //czytanie z pliku linijka po linijce,  parsowanie łańcucha
znaków i zapisanie go jako liczby typu int.
            // Metoda readLine zwróci null jeśli w pliku nie znajduje się
już więcej danych.
            String numberAsString = fileReader.readLine();
            number = Integer.parseInt(numberAsString);
        } finally {
            if (fileReader != null) {
                fileReader.close();
            }
        }
        return number;
    }

}
package PO_UR.Lab11;

import java.io.*;

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

        TextFile textFile = new TextFile();
        BinaryFile binaryFile = new BinaryFile();

        textFile.SaveFile();
        textFile.ReadFile();

        textFile.SaveFileWriter();
        System.out.println(textFile.ReadBufferFile());

    }
}
```

Przykład 2. BinraryFile

```
package PO_UR.Lab11;

import java.io.*;

public class BinaryFile {
```

```
    //pliki binarne zapis
    public static void SaveBinary()throws IOException {
        String filePath = "binarny.txt";
        int number = 1234567;
        DataOutputStream outputStream = null;

        //tworzenie instancji klasy, i przekazanie jej do konstrukotra,
FileOutputStream pozwala na zapis danych bajt po bajcie
        //DataOutputStream wykonuje zapis binarny z wykorzystaniem metody
writeInt
        try {
            outputStream = new DataOutputStream(new
FileOutputStream(filePath));
            outputStream.writeInt(number);
        } finally {
            if (outputStream != null) {
                outputStream.close();
            }
        }
    }

    public static int ReadBinray() throws IOException{
        String filePath = "binarny.txt";
        int number = 0;
        DataInputStream inputStream = null;
        //DataInputStream pozwala na czytanie większych kawałków pliku
zapisanego binarnie,
        // dzięki tej klasie możemy przeczytać liczbę typu int zapisaną
wcześniej w pliku.

        try {
            inputStream = new DataInputStream(new
FileInputStream(filePath));
            number = inputStream.readInt();
        } finally {
            if (inputStream != null) {
                inputStream.close();
            }
        }
        return number;
    }
}


package PO_UR.Lab11;

import java.io.*;

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

        BinaryFile binaryFile = new BinaryFile();

        binaryFile.SaveBinary();
        System.out.println(binaryFile.ReadBinray());
    }
}
```

Zadania do samodzielnego rozwiązania:

Zadanie 1.

Napisz program, który pobierze od użytkownika ścieżkę do pliku tekstowego oraz kilka linijek tekstu (dopóki użytkownik nie wprowadzi „-” jako linijki) i zapisze je do pliku tekstowego. Do wykonania tego zadania może Ci się przydać metoda System.lineSeparator() zwracająca znak nowej linii (jeśli chciałbyś oddzielić linie wprowadzone przez użytkownika).

Zadanie 2.

Napisz program, który pobierze od użytkownika ścieżkę do pliku i wyświetli zawartość pliku na ekranie wraz z informacją ile linii znajduje się w pliku.

Zadanie 3.

Napisz program, który poprosi od użytkownika o nazwę pliku wyjściowego oraz o podanie swojej daty urodzenia (osobno dzień, miesiąc i rok), a następnie zapisze te dane jako trzy osobne liczby binarnie.

Zadanie 4.

Napisz program, który pobierze od użytkownika ścieżkę do pliku binarnego z datą urodzenia, a następnie wyświetli ją na ekranie.