

## Lista zadań nr 4

### 1. Obsługa plików tekstowych

```
2 package com.company;
3 import java.io.*;
4 public class Main {
5
6     public static void main(String[] args) {
7         try {
8             PrintWriter obslugaZapisu = new PrintWriter( fileName: "nazwa.txt");
9             obslugaZapisu.println("Nowa linia :-");
10            obslugaZapisu.println("Kolejna");
11            obslugaZapisu.close();
12            FileReader obslugaOdczytu = new FileReader( fileName: "nazwa.txt");
13            BufferedReader buforOdczytu = new BufferedReader(obslugaOdczytu);
14            String pojedynczyWiersz = null;
15            while ((pojedynczyWiersz = buforOdczytu.readLine()) != null) {
16                System.out.println(pojedynczyWiersz);
17            }
18            buforOdczytu.close();
19
20        } catch (IOException exc) {
21            exc.printStackTrace();
22            System.exit( status: 1);
23        }
24
25    }
26 }
```

8 – obiekt klasy `PrintWriter` – obsługującej zapis do pliku tekstowego (jako parametr konstruktora przekazujemy nazwę pliku)

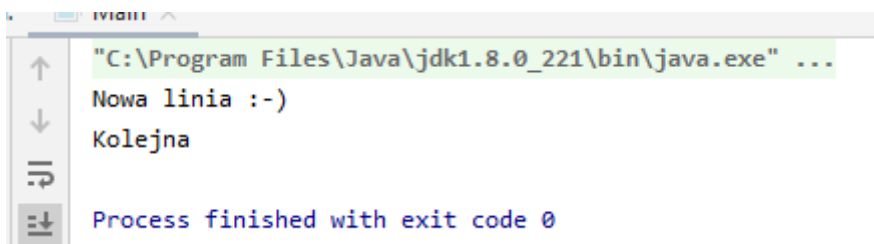
9,10 – dwa wiersze zapisywane kolejno do pliku.

11 – zamknięcie strumienia wyjściowego

12 – obiekt klasy `FileReader` – obsługującej odczyt z pliku tekstowego (jako parametr konstruktora przekazujemy nazwę pliku)

13 – bufor do którego trafią znaki odczytane z pliku

15 – pętla odczytująca kolejno wszystkie wiersze z bufora i wyświetlająca je w konsoli



```
"C:\Program Files\Java\jdk1.8.0_221\bin\java.exe" ...
Nowa linia :-)
Kolejna
Process finished with exit code 0
```

### 2. Obsługa plików binarnych

```

1  package com.company;
2  import java.io.*;
3
4  class TObserwacja {
5      String nazwa_obserwacji;
6      double[] pomiary;
7
8      public TObserwacja() {}
9
10     public TObserwacja(String nazwa, double[] tab_pomiarow) {
11         nazwa_obserwacji = nazwa;
12         pomiary = tab_pomiarow;
13     }
14
15     @ public void ZapiszDoStrumienia(DataOutputStream strumien)
16         throws IOException {
17         strumien.writeUTF(nazwa_obserwacji);
18         strumien.writeInt(pomiary.length);
19         for (int i=0; i<pomiary.length; i++) strumien.writeDouble(pomiary[i]);
20     }
21
22     @ public TObserwacja OdczytajZeStrumienia(DataInputStream strumien_wejscowy)
23         throws IOException {
24         nazwa_obserwacji = strumien_wejscowy.readUTF();
25         int n = strumien_wejscowy.readInt();
26         pomiary = new double[n];
27         for (int i=0; i<n; i++) pomiary[i] = strumien_wejscowy.readDouble();
28         return this;
29     }
30
31     public void WyświetlOpis() {
32         System.out.println(nazwa_obserwacji);
33         for (int i=0; i<pomiary.length; i++) System.out.print(pomiary[i] + " ");
34         System.out.println("");
35     }
36 }
37

```

4 – pomocnicza klasa TObserwacja (informacje przechowywane w obserwacjach będą zapisywane i odczytywane z pliku)

5,6 – pojedyncza obserwacja składa się z nazwy obserwacji oraz tabeli pomiarów (np. pomiarów temperatury)

10 – konstruktor parametrowy klasy

15 – metoda zapisująca dane obserwacji do strumienia binarnego przekazanego jako parametr wywołania (strumień musi być wcześniej otwarty)

17 – zapisujemy do strumienia nazwę obserwacji (jako tekst w formacie UTF)

18 – zapisujemy do strumienia ilość pomiarów, które będą zaraz zapisywane

19 – w pętli zapisujemy zawartość tabeli z pomiarami do strumienia

22 – metoda odczytująca dane o obserwacji ze strumienia przekazanego jako parametr wywołania

24 – odczytujemy nazwę obserwacji

25 – odczytujemy ilość pomiarów, które muszą być odczytane

27 – w pętli odczytujemy wartości wszystkich pomiarów (ich ilość odczytaliśmy wcześniej).

31 – metoda wyświetlająca kompletne informacje o obserwacji w konsoli

```

38 ▶ public class Main {
39
40 ▶ public static void main(String[] args) {
41     double[] pomiary_a = {1, 2, 3, 4, 5};
42     double[] pomiary_b = {7, 8, 9, 10, 11};
43
44     TObserwacja Obserwacja_1 = new TObserwacja( nazwa: "Dane obserwacji 1", pomiary_a);
45     TObserwacja Obserwacja_2 = new TObserwacja( nazwa: "Dane obserwacji 2", pomiary_b);
46
47     Obserwacja_1.WyswietlOpis();
48     Obserwacja_2.WyswietlOpis();
49
50     try {
51         DataOutputStream strumienWyjsciowy = new DataOutputStream(
52             new FileOutputStream( name: "dane.dat")
53         );
54         Obserwacja_1.ZapiszDoStrumienia(strumienWyjsciowy);
55         Obserwacja_2.ZapiszDoStrumienia(strumienWyjsciowy);
56         strumienWyjsciowy.close();
57
58         DataInputStream strumienWejsciowy = new DataInputStream(
59             new FileInputStream( name: "dane.dat")
60         );
61         TObserwacja Obserwacja_Odczyt_1 = new TObserwacja();
62         TObserwacja Obserwacja_Odczyt_2 = new TObserwacja();
63         Obserwacja_Odczyt_1.OdczytajZeStrumienia(strumienWejsciowy);
64         Obserwacja_Odczyt_2.OdczytajZeStrumienia(strumienWejsciowy);
65         Obserwacja_Odczyt_1.WyswietlOpis();
66         Obserwacja_Odczyt_2.WyswietlOpis();
67         strumienWejsciowy.close();
68     } catch (IOException exc) {
69         exc.printStackTrace();
70         System.exit( status: 1);
71     }
72 }
73
74 }

```

41, 42 – tablice pomocnicze do wypełnienia obserwacji

44, 45 – tworzymy dwa obiekty klasy TObserwacja

47, 48 – wyświetlamy testowo informacje o obu obserwacjach

51 – otwieramy strumień binarny do zapisu – będzie reprezentowany przez obiekt StrumienWyjsciowy

54, 55 – zapisujemy dane o obserwacjach do strumienia

58 – otwieramy strumień binarny do odczytu – będzie reprezentowany przez obiekt StrumienWejsciowy

61 – 66 – dwa nowe obiekty klasy TObserwacja, odczytują informacje zapisane wcześniej w pliku binarny i wyświetlają je w konsoli

```
"C:\Program Files\Java\jdk1.8.0_221\bin\java.exe" ...  
Dane obserwacji 1  
1.0 2.0 3.0 4.0 5.0  
Dane obserwacji 2  
7.0 8.0 9.0 10.0 11.0  
Dane obserwacji 1  
1.0 2.0 3.0 4.0 5.0  
Dane obserwacji 2  
7.0 8.0 9.0 10.0 11.0  
  
Process finished with exit code 0
```

### 3. Serializacja

```
1 package com.company;  
2  
3 import java.io.*;  
4 import java.util.*;  
5  
6 public class Main {  
7  
8     public static void main(String[] args) {  
9         Date data = new Date();  
10        int[] temperatury = { 25, 19, 22};  
11        String[] opisy = { "dzień 1", "dzień 2", "dzień 3" };  
12  
13        // Zapis  
14        try {  
15  
16            ObjectOutputStream strumienWyjsciowy = new ObjectOutputStream(  
17                new FileOutputStream( name: "test.ser")  
18            );  
19            strumienWyjsciowy.writeObject(data);  
20            strumienWyjsciowy.writeObject(opisy);  
21            strumienWyjsciowy.writeObject(temperatury);  
22            strumienWyjsciowy.close();  
23        } catch (IOException exc) {  
24            exc.printStackTrace();  
25            System.exit( status: 1);  
26        }  
27    }  
28 }
```

```

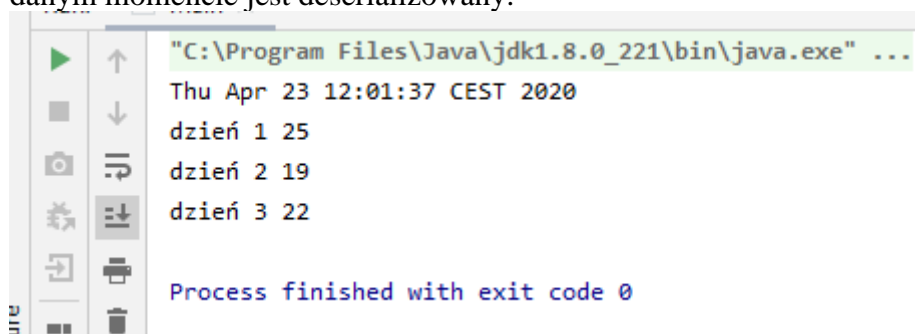
27
28
29     try {
30         ObjectInputStream strumienWejscowy = new ObjectInputStream(
31             new FileInputStream( name: "test.ser")
32         );
33         Date odczytData = (Date) strumienWejscowy.readObject();
34         String[] odczytOpis = (String[]) strumienWejscowy.readObject();
35         int[] odczytTemp = (int[]) strumienWejscowy.readObject();
36         strumienWejscowy.close();
37         System.out.println(String.valueOf(odczytData));
38         for (int i=0; i<odczytOpis.length; i++)
39             System.out.println(odczytOpis[i] + " " + odczytTemp[i]);
40     } catch(IOException exc) {
41         exc.printStackTrace();
42         System.exit( status: 1);
43     } catch(ClassNotFoundException exc) {
44         System.out.println("Nie można odnaleźć klasy obiektu");
45         System.exit( status: 1);
46     }
47
48
49 }
50

```

**9, 10, 11** – trzy różne obiekty – pojedyncza data, tablica liczb oraz tablica napisów – one będą serializowane

**19 – 21** – serializujemy kolejno wszystkie 3 obiekty w otwartym strumieniu wyjściowym. (wystarczy użyć metody writeObject());

**32 – 34** – korzystamy z metody readObject() aby deserializować obiekty odczytane ze strumienia. Oczywiście każdy odczytany zbiór bajtów, musi być rzutowany na typ, który w danym momencie jest deserializowany.



```

"C:\Program Files\Java\jdk1.8.0_221\bin\java.exe" ...
Thu Apr 23 12:01:37 CEST 2020
dzień 1 25
dzień 2 19
dzień 3 22
Process finished with exit code 0

```

### Zadanie 1 (20 pkt.)

Proszę przygotować aplikację, która będzie umożliwiała przeprowadzanie testów jednokrotnego wyboru. Aplikacja rozpoczyna pracę od poproszenia użytkownika o jakiś unikalny login (identyfikator), a następnie na kolejnych ekranach prezentuje pytania oraz pozwala wskazać odpowiedzi. Każde pytanie prezentowane jest osobno. Nie ma możliwości przechodzenia pomiędzy pytaniami. Kolejność pytań jak i kolejność odpowiedzi do wyboru

powinna zmieniać się losowo. Źródłem danych dla testu powinien być plik tekstowy w formacie:

```
NrPytania; TrescPytania; Odp1; Odp2; Odp3; Odp4; NrPrawidlowejOdpowiedzi
```

Wyniki przypisane poszczególnym osobom aplikacja przechowuje w pliku binarnym i pozwala je przeglądać użytkownikowi o identyfikatorze Admin.

Dodatkowo po każdej odpowiedzi aplikacja zapisuje swój stan przy wykorzystaniu serializacji i automatycznie odzyskuje go, jeśli poprzedniej sesji z programem użytkownik nie dokończył testu.

Do realizacji zadania wykorzystujemy interfejs dialogowy.