

# Informatika 2

Práca so súbormi



# Pojmy zavedené v 7. prednáške (1)

- generická trieda
  - syntax
- typový parameter
  - konvencie

# Pojmy zavedené v 8. prednáške (2)

- možné riešenia v jazykoch
  - jedna generická trieda -> n tried pri preklade
  - jedna generická trieda -> n tried za behu
  - jedna generická trieda -> jedna trieda pri preklade

# Pojmy zavedené v 8. prednáške (3)

- výhody využitia generickej triedy
  - netreba pretypovávať
  - úplná typová kontrola pri preklade
  - nie je možné vložiť položku iného typu

# Pojmy zavedené v 8. prednáške (4)

- type erasure/odstraňovanie typov
  - problémy
  - použitie generickej triedy bez typových parametrov
- generické triedy a polymorfizmus
  - obmedzenia
    - syntax

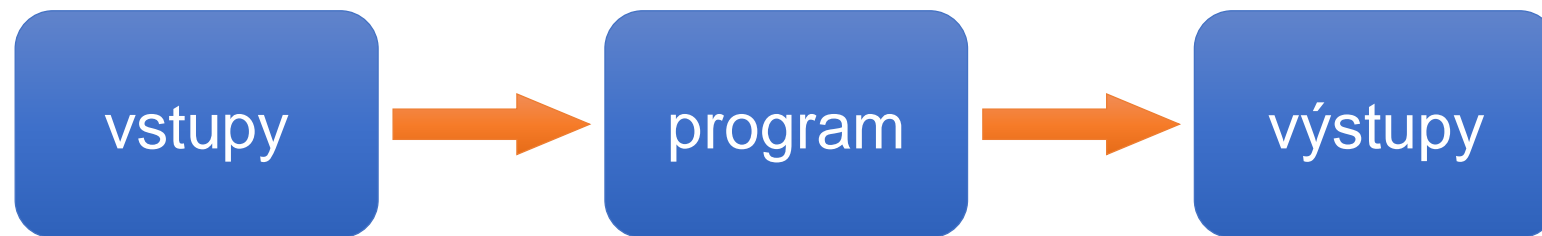
# Pojmy zavedené v 8. prednáške (5)

- iterátory a foreach
- generický interface
- generické metódy
  - automatické odvodzovanie typov
- divoké karty

# Vstupy a výstupy programov

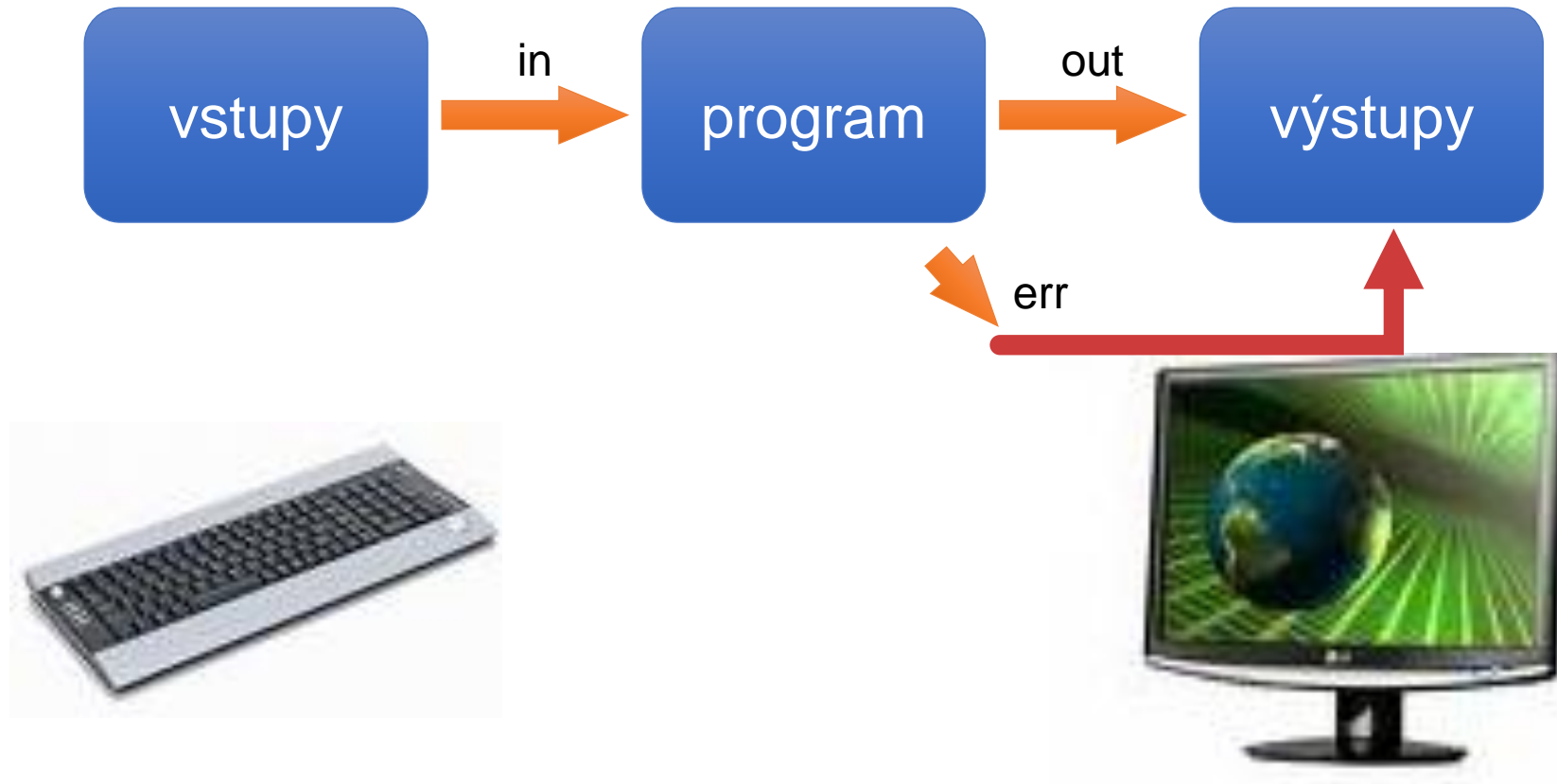
- programy – algoritmy
  - hromadnosť => nie na jediné použitie
  - jeden druh úlohy
  - rôzne začiatkové hodnoty
  - rôzne výsledky

# Vstupy a výstupy programov

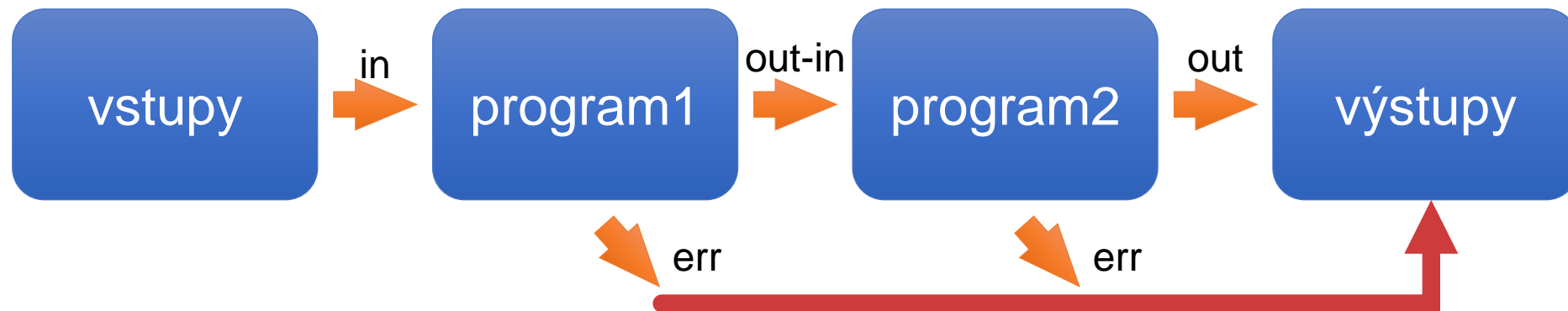




# Štandardný vstup a výstup



# Prepojenie viacerých programov



# Štandardný vstup a výstup

- Trieda System
- atribúty objektového typu:
  - in – štandardný vstup
  - out – štandardný výstup
  - err – štandardný chybový výstup

# Štandardný výstup

- výpis textu na obrazovku
- priame použitie v programe
- `System.out`, `System.err` – statický typ `PrintStream`
  - najčastejšie správy
    - `print(String text)`
    - `println()` – nový riadok
    - `println(String text)`

# Štandardný vstup

- čítanie textu (postupnosti znakov) z klávesnice
- čaká na ukončenie písania na klávesnici – enter
- kontrolný opis na obrazovku – JVM (+OS)
- `System.in` – statický typ `InputStream`
  - `System.in` sa väčšinou nevyužíva v programe priamo
  - využitie v programe pomocou triedy `Scanner`
  - `Scanner` – obaľuje `System.in`

# Scanner next a hasNext

- oddeľovače – skupina znakov (Whitespace): medzera, tabulátor, nový riadok
- rozdelenie vstupného riadku pomocou oddeľovačov na slová (tokeny)
- boolean hasNext()
  - existuje ešte slovo?
- String next()
  - prečíta nasledujúce slovo, všetky oddeľovače pred slovom vynechá

# Scanner

- ďalšie správy inštancii triedy Scanner
- boolean hasNextPrimitivnyTyp()
  - nasledujúce slovo
- PrimitivnyTyp nextPrimitivnyTyp()
  - InputMismatchException
- boolean hasNextLine() -
- String nextLine()

# Scanner – vytvorenie

- čítanie štandardného vstupu:

```
Scanner klavesnica = new Scanner(System.in);
```

- čítanie ľubovoľného reťazca

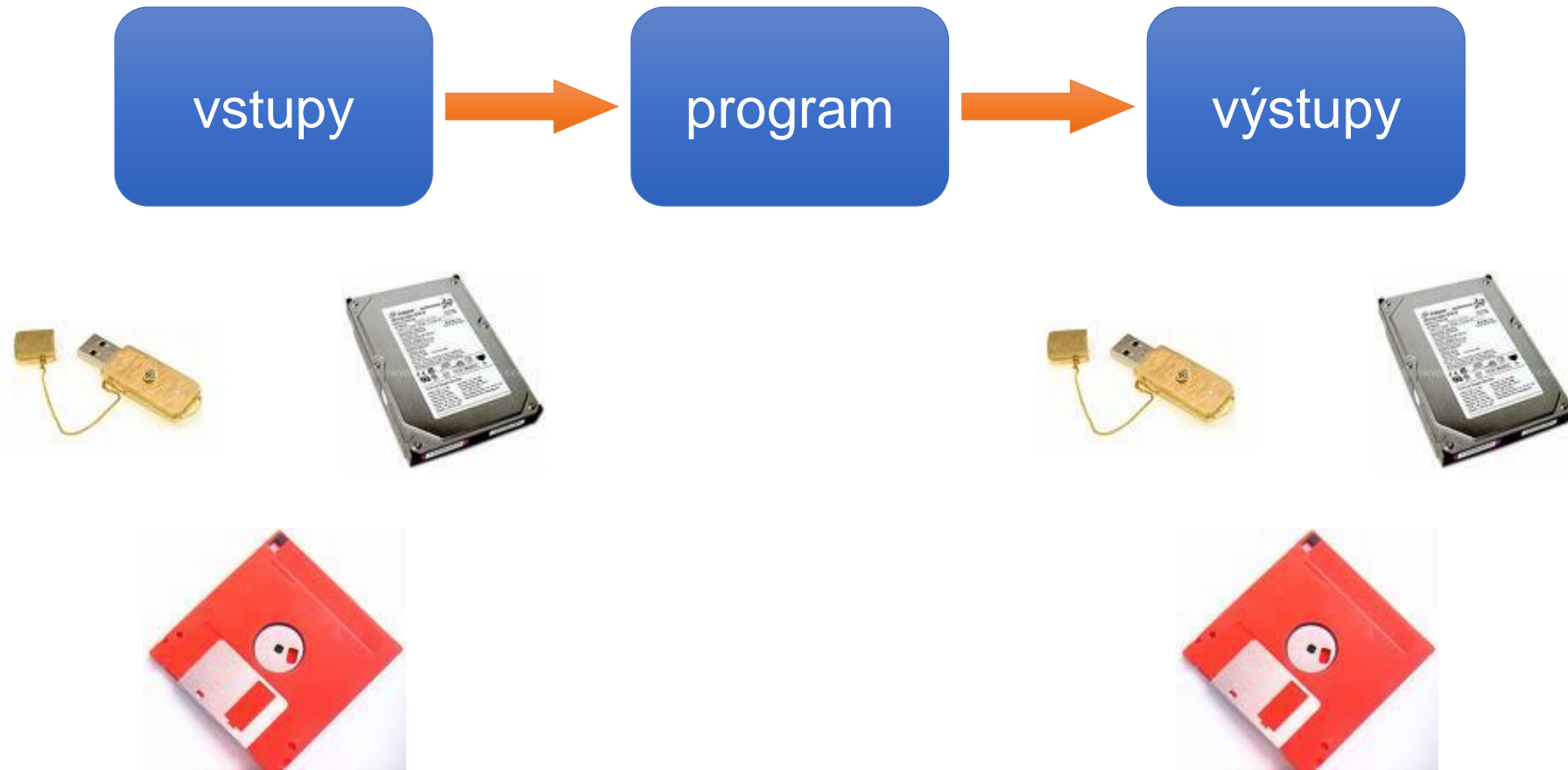
```
String riadok = ...;  
Scanner parser = new Scanner(riadok);
```



# Projekt KCalB

- katalóg CD a DVD
- objekty treba vždy nanovo vytvárať
- (alebo nechať navždy zapnutý program)
- úloha: ukladanie katalógu do súborov

# Súbory



# Trieda File (1)

- poskytuje základné informácie o súbore
  - názov súboru
  - čas zmeny
  - práva na súbor
  - existencia súboru na disku
  - absolútna cesta
  - ...

## Trieda File (2)

- operácie so súborom
  - vytvorenie
  - vymazanie
  - premenovanie
  - ...
- spolupráca programu v jazyku Java s OS
- nepracuje s obsahom súboru

# Práca so súbormi

- čítanie
  - otvorenie súboru na čítanie
  - postupné čítanie obsahu
  - zatvorenie súboru
- zápis
  - otvorenie súboru na zápis
  - postupný zápis nového obsahu
  - zatvorenie súboru

# Možnosti práce s obsahom

- textové súbory
- binárne súbory
- serializácia objektov

# Textové súbory

- čitateľné človekom
- najťažšie na strojové spracovanie
- príklady
  - textové súbory .txt
  - zdrojové kódy .java
  - ...

# Textové súbory v jazyku Java

- čítanie
  - trieda Scanner
  - Scanner(File subor)
- zápis
  - trieda PrintWriter
  - PrintWriter(File subor)
  - rozhranie podobné ako System.out
- ! nutnosť uzavretia súboru správou close()



# Metóda zapis v triede Katalog

```
public void zapis() throws IOException {  
    var subor = new File("Katalog.txt");  
    var zapisovac = new PrintWriter(subor);  
    for (var dielo : this.zoznam) {  
        dielo.zapis(zapisovac);  
    }  
    zapisovac.close();  
}
```

# Metóda zapis v triede AudiovizualneDielo

```
public abstract void zapis(PrintWriter zapisovac);
```

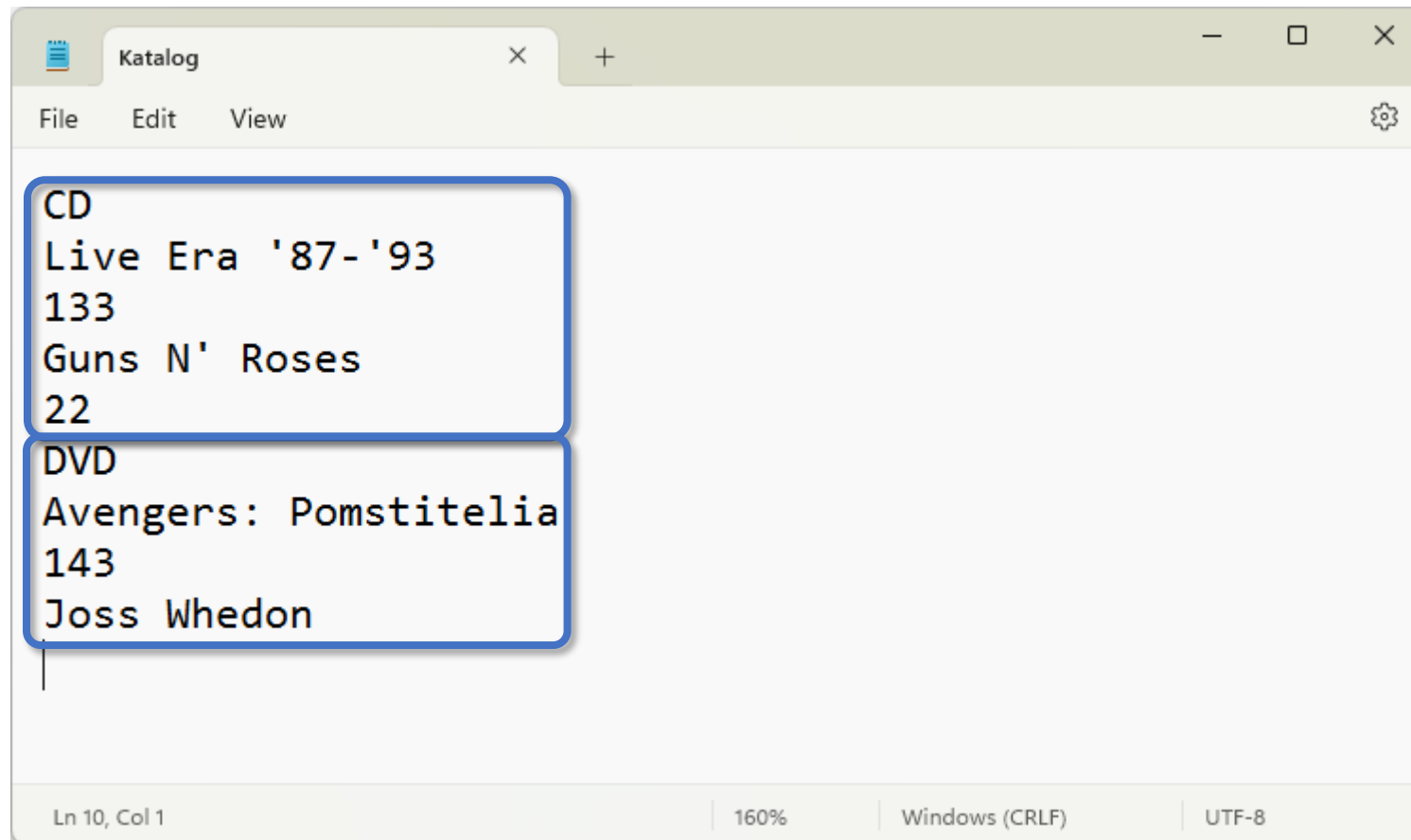
# Metóda zapis v triede CD

```
public void zapis(PrintWriter zapisovac) {  
    zapisovac.println("CD");  
    zapisovac.println(this.getTitul());  
    zapisovac.println(this.getCelkovyCas());  
    zapisovac.println(this.autor);  
    zapisovac.println(this.pocetSkladieb);  
}
```

# Metóda zapis v triede DVD

```
public void zapis(PrintWriter zapisovac) {  
    zapisovac.println("DVD");  
    zapisovac.println(this.getTitul());  
    zapisovac.println(this.getCelkovyCas());  
    zapisovac.println(this.reziser);  
}
```

# Výsledok v súbore



```
Katalog
File Edit View
CD
Live Era '87-'93
133
Guns N' Roses
22
DVD
Avengers: Pomstitelia
143
Joss Whedon
Ln 10, Col 1 | 160% | Windows (CRLF) | UTF-8
```

# Metóda nacitaj v triede Katalog

```
public void nacitaj() throws IOException {  
    this.zoznam.clear();  
    var subor = new File("Katalog.txt");  
    var citac = new Scanner(subor);  
    while (citac.hasNextLine()) {  
        ...  
    }  
    citac.close();  
}
```

# Metóda nacitaj v triede Katalog

```
var typ = citac.nextLine();  
if (typ.equals("CD")) {  
    this.zoznam.add(CD.nacitaj(citac));  
} else {  
    this.zoznam.add(DVD.nacitaj(citac));  
}
```

# Metóda nacitaj v triede CD

```
public static CD nacitaj(Scanner citac) {  
    return new CD(  
        citac.nextLine(),  
        citac.nextInt(),  
        citac.nextLine(),  
        citac.nextInt()  
    );  
}
```



# Metóda nacitaj v triede DVD

```
public static DVD nacitaj(Scanner citac) {  
    return new DVD(  
        citac.nextLine(),  
        citac.nextInt(),  
        citac.nextLine()  
    );  
}
```

# Problémy s textovými súbormi (1)

- príklad nebude fungovať
- problém s koncom riadku za číslom (133, 143)
- `nextLine` po `nextInt` prečíta prázdny reťazec
- možné riešenia
  - zapisovať čísla bez konca riadku – viac problémov ako úžitku
  - prečítať najskôr riadok a ten postupne čítať pomocou triedy `Scanner`
  - po čísle prečítať zvyšok riadku a zabudnúť

# Problémy s textovými súbormi (2)

- ďalší problém – viacriadkové komentáre
  - riešenie = DÚ
- problémy s formátovaním
  - zápis – bez problémov ľubovoľný formát
  - čítanie – komplikované

# Binárne súbory

- nečitateľné človekom, resp. čitateľné komplikovane
- formát rovnaký ako v pamäti – jednoducho spracovateľný počítačom
- pri dodržaní poradia zápisu a čítania dát (musia byť zhodné) nehrozí problém s formátovaním

# Binárne súbory v jazyku Java (1)

- čítanie
  - trieda `FileInputStream`
  - `FileInputStream(File subor)`
- zápis
  - trieda `FileOutputStream`
  - `FileOutputStream(File subor)`
- nutnosť uzavretia súboru správou `close()`

# Binárne súbory v jazyku Java (2)

- základný prístup k súboru
  - na prístup k súboru ich používa aj Scanner/PrintWriter
- InputStream
  - `int read(byte[] pole)`
  - načíta `pole.length` bajtov zo súboru do poľa
- OutputStream
  - `void write(byte[] pole)`
  - zapíše `pole.length` bajtov z poľa do súboru
- možnosť pracovať iba s `byte[]`
  - komplikované – nutnosť konvertovať dáta na `byte[]`

# Zjednodušenie práce s binárnymi súbormi

- čítanie
  - trieda `DataInputStream`
  - `DataInputStream(InputStream stream)`
- zápis
  - trieda `DataOutputStream`
  - `DataOutputStream(OutputStream stream)`
- nutnosť uzavretia súboru správou `close()`

# Trieda DataOutputStream

- `writePrimitivnyTyp(PrimitivnyTyp hodnota)`
  - zapíše hodnotu primitívneho typu do súboru
- `writeUTF(String retazec)`
  - zapíše reťazec do súboru



# Trieda DataInputStream

- *PrimitivnyTyp* read*PrimitivnyTyp*()
  - prečíta hodnotu primitívneho typu zo súboru
- String readUTF()
  - prečíta reťazec zo súboru

# Metóda zapis v triede Katalog

```
public void zapis() throws IOException {  
    var subor = new File("Katalog.bin");  
    var stream = new FileOutputStream(subor);  
    var zapisovac = new DataOutputStream(stream);  
    for (var dielo : this.zoznam) {  
        dielo.zapis(zapisovac);  
    }  
    zapisovac.close();  
}
```

# Metóda zapis v triede AudiovizualneDielo

```
public abstract void zapis(DataOutputStream zapisovac) throws IOException;
```

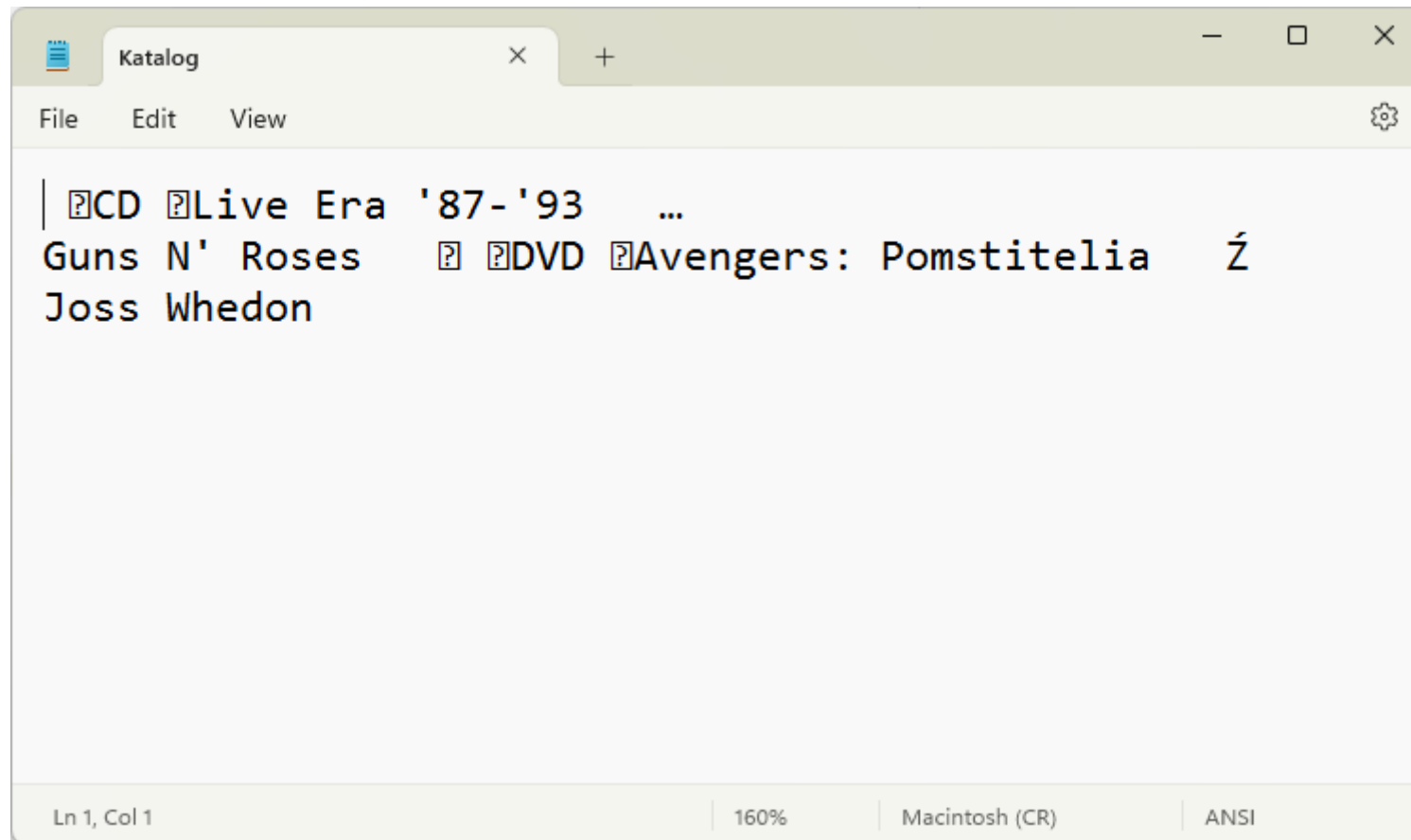
## Metóda zapis v triede CD

```
public void zapis(DataOutputStream zapisovac) throws IOException {  
    zapisovac.writeUTF("CD");  
    zapisovac.writeUTF(this.getTitul());  
    zapisovac.writeInt(this.getCelkovyCas());  
    zapisovac.writeUTF(this.autor);  
    zapisovac.writeInt(this.pocetSkladieb);  
}
```

# Metóda zapis v triede DVD

```
public void zapis(DataOutputStream zapisovac) throws IOException {  
    zapisovac.writeUTF("DVD");  
    zapisovac.writeUTF(this.getTitul());  
    zapisovac.writeInt(this.getCelkovyCas());  
    zapisovac.writeUTF(this.reziser);  
}
```

# Výsledok v súbore



```
| ?CD ?Live Era '87-'93 ...  
Guns N' Roses    ? ?DVD ?Avengers: Pomstitelia  Ž  
Joss Whedon
```

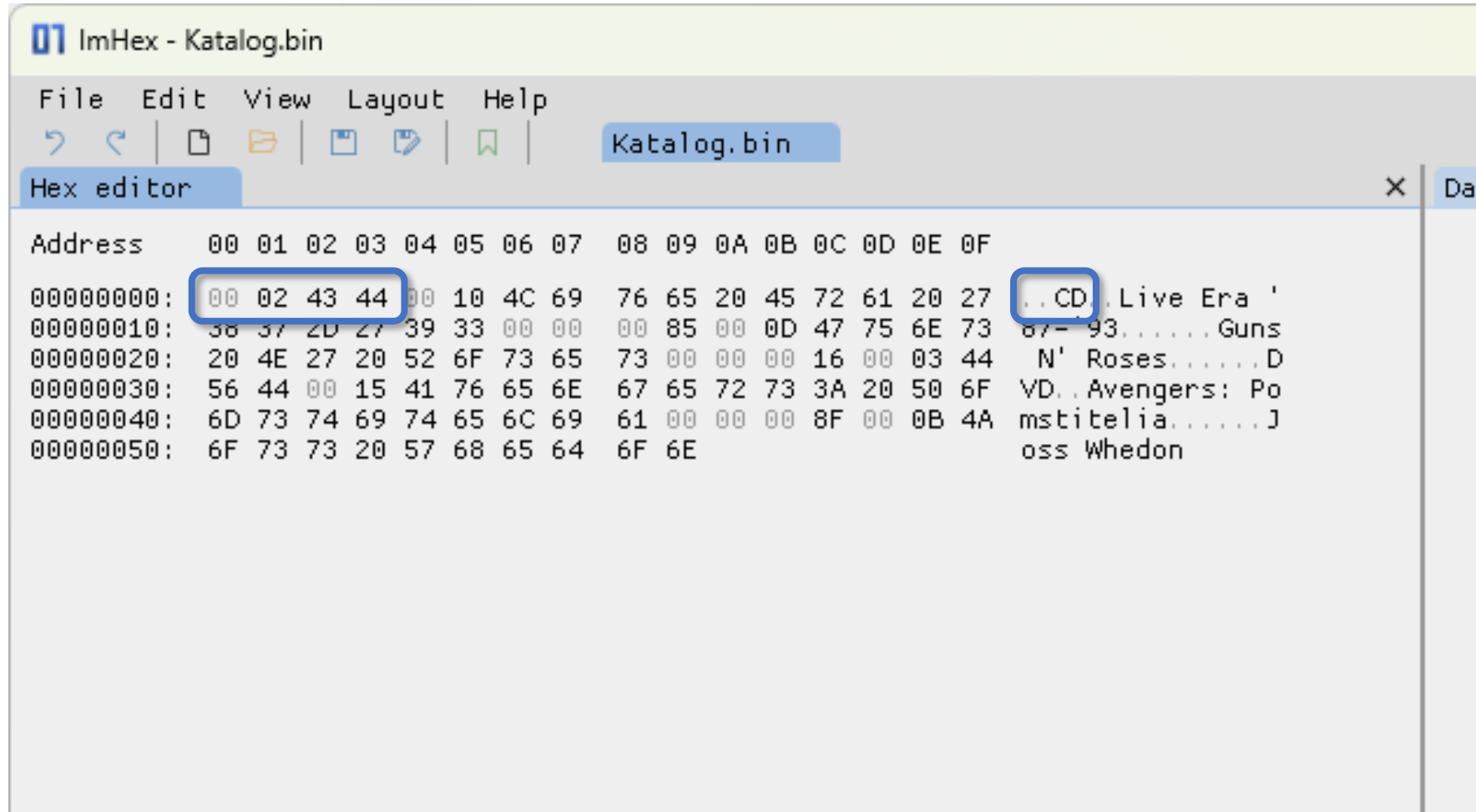
Ln 1, Col 1 | 160% | Macintosh (CR) | ANSI

# Hexadecimálne zobrazenie

```
ImHex - Katalog.bin
File Edit View Layout Help
Katalog.bin
Hex editor X Da

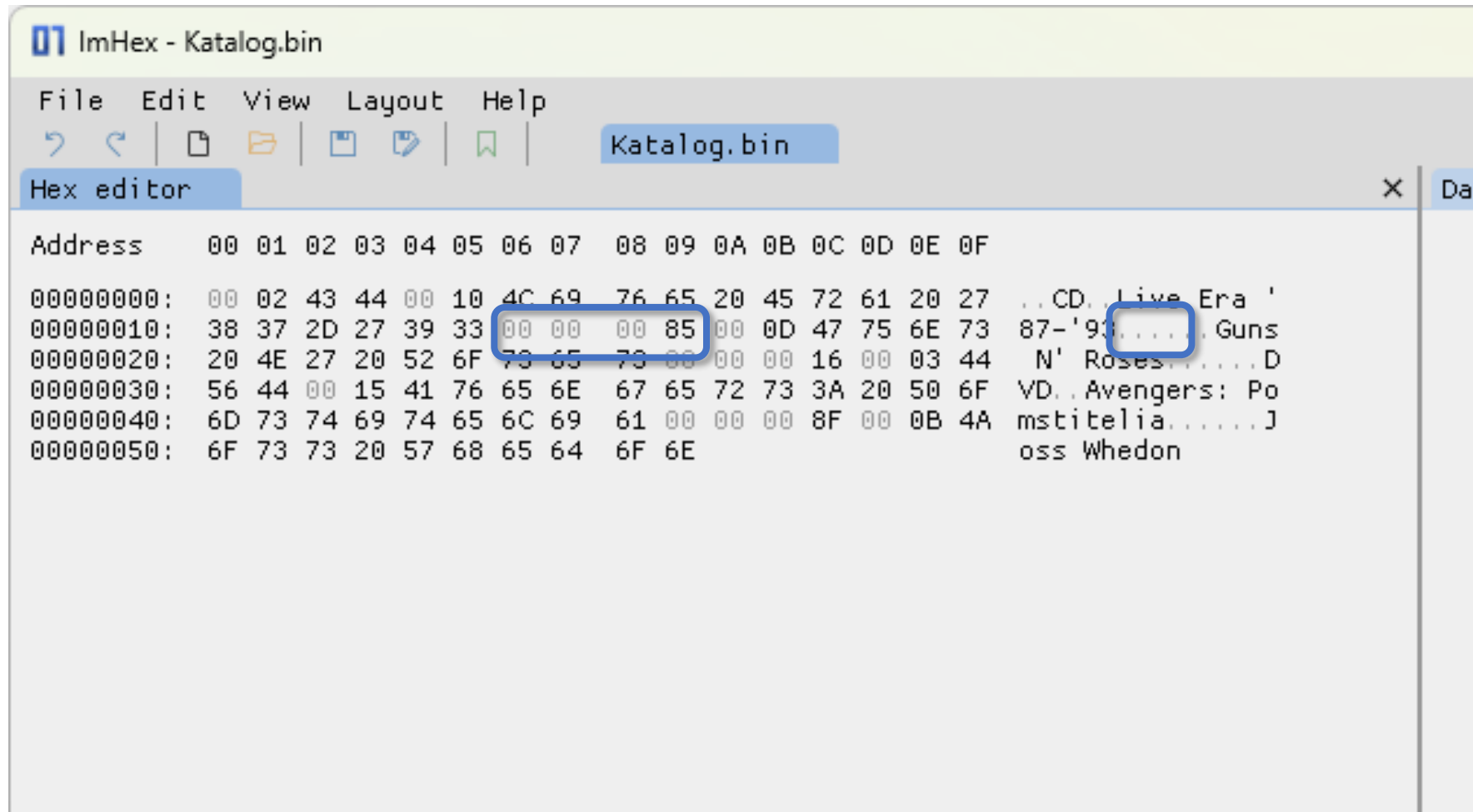
Address 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000: 00 02 43 44 00 10 4C 69 76 65 20 45 72 61 20 27 ..CD..Live Era '
00000010: 38 37 2D 27 39 33 00 00 00 85 00 0D 47 75 6E 73 87-'93.....Guns
00000020: 20 4E 27 20 52 6F 73 65 73 00 00 00 16 00 03 44 N' Roses.....D
00000030: 56 44 00 15 41 76 65 6E 67 65 72 73 3A 20 50 6F VD..Avengers: Po
00000040: 6D 73 74 69 74 65 6C 69 61 00 00 00 8F 00 0B 4A mstitelia.....J
00000050: 6F 73 73 20 57 68 65 64 6F 6E oss Whedon
```

# Hexadecimálne zobrazenie – writeUTF





# Hexadecimálne zobrazenie – writeInt



# Metóda nacitaj v triede Katalog

```
public void nacitaj() {  
    var koniecSuboru = false;  
    this.zoznam.clear();  
    var subor = new File("Katalog.bin");  
    var stream = new FileInputStream(subor);  
    var citac = new DataInputStream(stream);  
    while (!koniecSuboru) {  
        ...  
    }  
    citac.close();  
}
```

# Metóda nacistaj v triede Katalog

```
try {  
    var typ = citac.readUTF();  
    if (typ.equals("CD")) {  
        this.zoznam.add(CD.nacistaj(citac));  
    } else {  
        this.zoznam.add(DVD.nacistaj(citac));  
    }  
} catch (EOFException ex) {  
    koniecSuboru = true;  
}
```

# Metóda nacistaj v triede CD

```
public static CD nacistaj(DataInputStream citac) {  
    return new CD(  
        citac.readUTF(),  
        citac.readInt(),  
        citac.readUTF(),  
        citac.readInt()  
    );  
}
```

# Metóda nacitaj v triede DVD

```
public static DVD nacitaj(DataInputStream citac) {  
    return new DVD(  
        citac.readUTF(),  
        citac.readInt(),  
        citac.readUTF()  
    );  
}
```

# Objektové prúdy

- aplikácia
  - štruktúra spolupracujúcich objektov
  - v operačnej pamäti
  - ľubovoľne zložitá
  - prirodzene nelineárna
- ukladanie na vonkajšiu pamäť
  - postupnosť informácií – lineárna forma
- objektové streamy – serializácia
  - ukladanie stavu objektov na vonkajšiu pamäť
  - obnovenie stavu objektov z vonkajšej pamäte

# ObjectOutputStream

- uloží celý objekt = serializácia
- hodnoty všetkých atribútov
  - hodnoty primitívnych typov
  - objektové atribúty ako objekty – rekurzia
- rieši problém viacnásobných odkazov na objekt
- writeObject

# ObjectInputStream

- obnoví celý objekt = deserializácia
- hodnoty všetkých atribútov
- rieši problém viacnásobných odkazov na objekt
- readObject



# Podmienka

- trieda ukladaného/obnovovaného objektu implementuje interface Serializable
  - môže dediť od predka
- značkovací interface
  - žiadne metódy navyše – vnútorné riešenie Java
- triedy z balíčkov knižnice Java
  - bežne implementujú interface Serializable
  - nie je to 100% pravidlo

# Prvok kontajnera, chyba Serializable

```
Exception in thread "main" java.io.NotSerializableException: fri.kcaib.diela.CD
    at java.io.ObjectOutputStream.writeObject0 (ObjectOutputStream.java:1193)
    at java.io.ObjectOutputStream.writeObject (ObjectOutputStream.java:353)
    at java.util.ArrayList.writeObject (ArrayList.java:866)
    at jdk.internal.reflect.NativeMethodAccessorImpl.invoke0 (Native Method)
    at jdk.internal.reflect.NativeMethodAccessorImpl.invoke (NativeMethodAccessorImpl.java:64)
    at jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke (DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke (Method.java:564)
    at java.io.ObjectStreamClass.invokeWriteObject (ObjectStreamClass.java:1196)
    at java.io.ObjectOutputStream.writeSerialData (ObjectOutputStream.java:1523)
    at java.io.ObjectOutputStream.writeOrdinaryObject (ObjectOutputStream.java:1444)
    at java.io.ObjectOutputStream.writeObject0 (ObjectOutputStream.java:1187)
    at java.io.ObjectOutputStream.writeObject (ObjectOutputStream.java:353)
    at fri.kcaib.katalog.Katalog.zapis (Katalog.java:69)
    at fri.kcaib.Main.main (Main.java:33)
```

# Príklad KCalB – podmienka

```
import java.io.Serializable;

public abstract class AudiovizualneDielo implements Serializable {
    ...
}
```

# Metóda zapis v triede Katalog

```
public void zapis() throws IOException {  
    var subor = new File("Katalog.obj");  
    var stream = new FileOutputStream(subor);  
    var zapisovac = new ObjectOutputStream(stream);  
    zapisovac.writeObject(this.zoznam);  
    zapisovac.close();  
}
```

# Výsledok v súbore

```
ImHex - Katalog.obj
File Edit View Layout Help
Katalog.obj
Hex editor
Address 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000: AC ED 00 05 73 72 00 13 6A 61 76 61 2E 75 74 69 ....sr..java.util
00000010: 6C 2E 41 72 72 61 79 4C 69 73 74 78 81 D2 1D 99 l.ArrayListx...
00000020: C7 61 9D 03 00 01 49 00 04 73 69 7A 65 78 70 00 .a....I..sizep.
00000030: 00 00 02 77 04 00 00 00 02 73 72 00 12 66 72 69 ...w....sr..fri
00000040: 2E 6B 63 61 69 62 2E 64 69 65 6C 61 2E 43 44 47 .kcaib.diel.CDG
00000050: 90 31 70 05 07 79 04 02 00 02 49 00 0D 70 6F 63 .1p..y...I..poc
00000060: 65 74 53 6B 6C 61 64 69 65 62 4C 00 05 61 75 74 etSkladiebL..aut
00000070: 6F 72 74 00 12 4C 6A 61 76 61 2F 6C 61 6E 67 2F ort..Ljava/lang/
00000080: 53 74 72 69 6E 67 3B 78 72 00 22 66 72 69 2E 6B String;xr."fri.k
00000090: 63 61 69 62 2E 64 69 65 6C 61 2E 41 75 64 69 6F caib.diel.Audio
000000A0: 76 69 7A 75 61 6C 6E 65 44 69 65 6C 6F A2 A9 B4 vizualneDielo...
000000B0: 73 C2 B7 61 C2 02 00 03 49 00 0A 63 65 6C 6B 6F s..a...I..celko
000000C0: 76 79 43 61 73 4C 00 08 6B 6F 6D 65 6E 74 61 72 vyCasL..komentar
000000D0: 74 00 19 4C 6A 61 76 61 2F 6C 61 6E 67 2F 53 74 t..Ljava/lang/St
000000E0: 72 69 6E 67 42 75 69 6C 64 65 72 3B 4C 00 05 74 ringBuilder;L..t
000000F0: 69 74 75 6C 71 00 7E 00 03 78 70 00 00 00 85 73 itulq..xp...s
00000100: 72 00 17 6A 61 76 61 2E 6C 61 6E 67 2E 53 74 72 r..java.lang.Str
00000110: 69 6F 67 42 75 69 6C 64 65 72 3C 05 6B 44 5A 4C ingBuilder; 71
```

## Metóda nacitaj v triede Katalog

```
public void nacitaj() throws IOException, ClassNotFoundException {  
    var subor = new File("Katalog.bin");  
    var stream = new FileInputStream(subor);  
    var citac = new ObjectInputStream(stream);  
    this.zoznam = (ArrayList<AudiovizualneDielo>) citac.readObject();  
    citac.close();  
}
```

# Serializácia/Deserializácia

- POZOR: inštancia ide deserializovať len v prípade, že neboli v zdrojovom kóde triedy vykonané žiadne zásadné zmeny (upravený zoznam atribútov/metód/konštruktorov)
- serializáciu/deserializáciu používať len na nedôležité dáta
  - dáta, ktoré je možné opäť stiahnuť z internetu
  - dáta, ktoré je možné opäť vypočítať
  - dáta, ktoré nie sú pre používateľa dôležité
  - ...

# Resource

- dátové súbory šírené s programom
- prekladač automaticky vkladá do jar súborov
  - netreba samostatne šíriť
  - na jeho dostupnosť sa dá spoľahnúť
- iba na čítanie



# Otvorenie resource v jazyku Java

- správa `ClassLoader.getResourceAsStream`
- vracia `InputStream`
- je možné čítať pomocou
  - `Scanner` – ako textový súbor
  - `DataInputStream` – ako binárny súbor
  - `ObjectInputStream` – ako objektový prúd

```
var subor = ClassLoader.getResourceAsStream("mapa.txt");  
var citac = new Scanner(subor);  
  
...
```

# BONUS: Využitie resource pomocou ShapesGE

- súbor sbge.ini sa automaticky hľadá v resource
  - nemusí byť teda prikladaný samostatne, stačí ho mať v jar
- obrázky sa vedia načítavať z resource
  - štandardne sa čítajú zo súboru
  - treba prepnúť v konfigurácii

```
[Shapes]  
ImageSource = resource
```

# try-with-resources (1)

- súbor treba vždy zavrieť
  - správa close()
  - treba používať try-finally, aby sme si boli zatvorením istý

```
var subor = new File("Katalog.txt");  
var zapisovac = new PrintWriter(subor);  
  
try {  
    ...  
} finally {  
    zapisovac.close();  
}
```

## try-with-resources (2)

- príkaz try-with-resources
  - nemýliť s resource
- automaticky zatvára súbor (Autoclosable.close())
- môže – nemusí obsahovať catch bloky

```
var subor = new File("Katalog.txt");  
  
try (var zapisovac = new PrintWriter(subor)) {  
    ...  
}
```