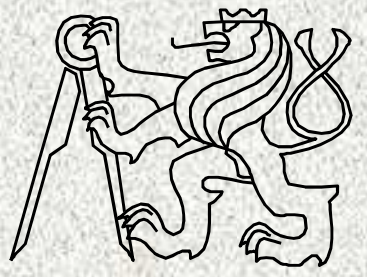
Soubory



BD6B36PJV 005

Fakulta elektrotechnická
České vysoké učení technické

Soubory

- Soubor je množina údajů uložená ve vnější paměti počítače, obvykle na disku
- Pro soubor jsou typické tyto operace:
    - otevření souboru, čtení údaje, zápis údaje, uzavření souboru
- Přístup k údajům (čtení nebo zápis) může být:
    - sekvenční nebo libovolný (adresovatelný)
- Soubory se sekvenčním přístupem umožňují pouze postupné (sekvenční) čtení nebo zápis údajů
- Soubory s libovolným přístupem umožňují adresovatelné čtení nebo zápis údaje (podobně jako pole)
- Způsob přístupu k údajům v souboru není zakódován v souboru, ale je dán programem
- Zde se budeme zabývat sekvenčními soubory, pro zájemce i adresovatelnými

Typy souborů

- Podle způsobu kódování informace v souboru známe:
  - Soubory textové
  - Soubory binární

- Textový soubor je posloupnost znaků členěná na řádky
  - každý znak je reprezentován jedním bytem
  - členění na řádky je závislé na platformě a obvykle je dáno jedním nebo dvěma řídicími znaky (*CR*, *CR LF ,* 0x0d 0x0c, "**\r\n"**)
  - Textový soubor je čitelný textovým editorem

- Binární soubor je posloupnost byteů a informace je kódována vnitřním kódem počítače. Do binárního souboru mohou být zapsány:
  - byte, jednoduché proměnné, pole, data celých objektů, …

- Důležité: Informace o typu souboru (textový, binární) ani o způsobu kódování informace není v souboru obsažena. Správnou interpretaci přečteného souboru musí zajistit uživatelský program

Práce se souborem

- Soubor je v běžném stavu chráněn proti změnám dat v souboru (je uzavřen)
- Před vlastní manipulací s daty v souboru (čtení/zápis) musíme soubor otevřít

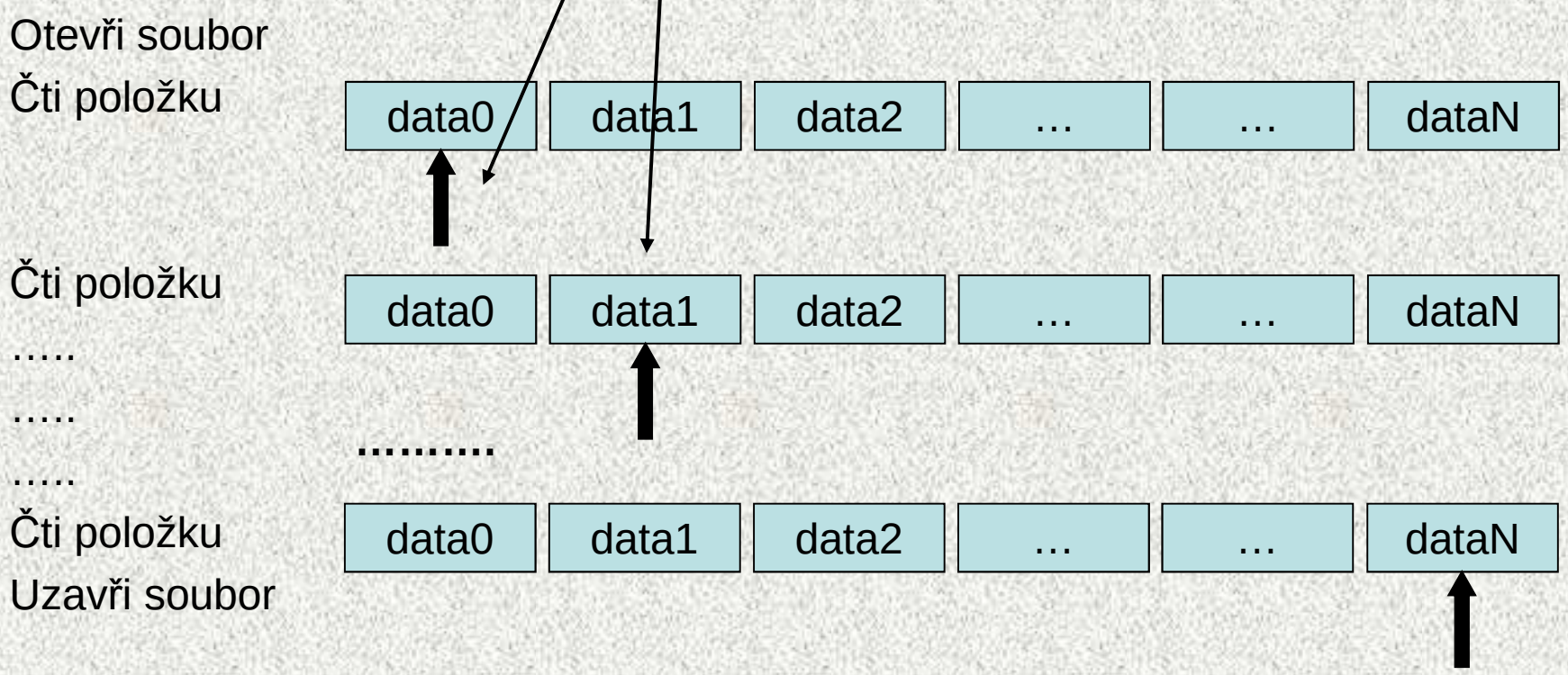
- Při práci s daty v souboru jsou typické tyto operace:
  1) otevření souboru,
  2) čtení nebo zápis údajů,
  3) uzavření souboru.

- Přístup k údajům v souboru (čtení nebo zápis) může být:
  - postupný (sekvenční)
  - přímý (nahodilý)

- Soubory se sekvenčním přístupem (sekvenční soubory) umožňují pouze postupné (sekvenční) čtení/zápis údajů
- Soubory s přímým přístupem umožňují nahodilé čtení/zápis údaje
- Důležité: způsob přístupu k údajům v souboru není zakódován v souboru, ale je dán programem a technickou podporou

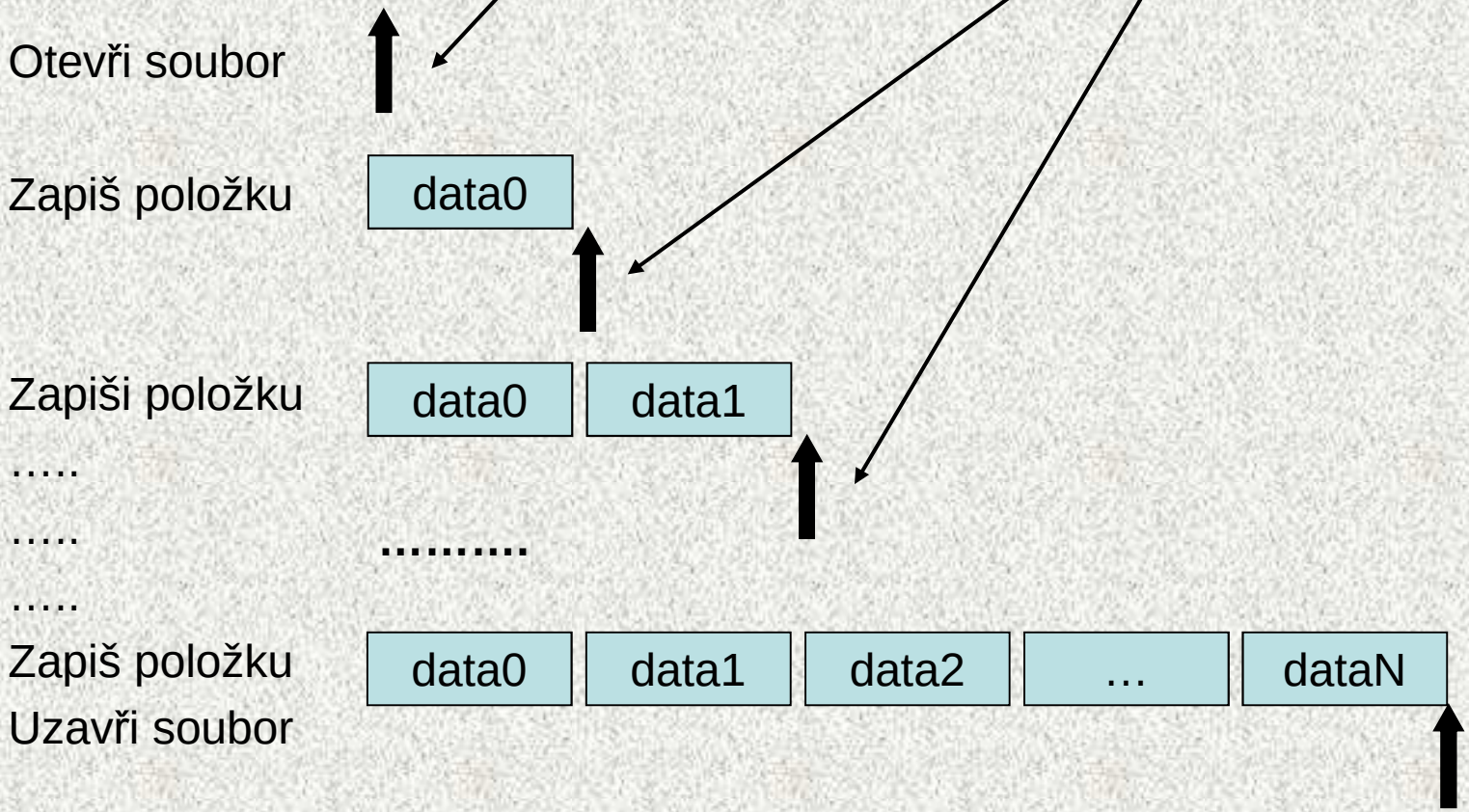
Postupný (sekvenční) přístup k datům v souboru

- Otevření souboru pro čtení nastaví souborové ukazovátko na začátek souboru
- Příkazem pro čtení souboru se přečte položka na kterou ukazuje souborové ukazovátko a to se pak automaticky posune na další položku
- Sekvenční čtení může probíhat až do dosažení konce souboru



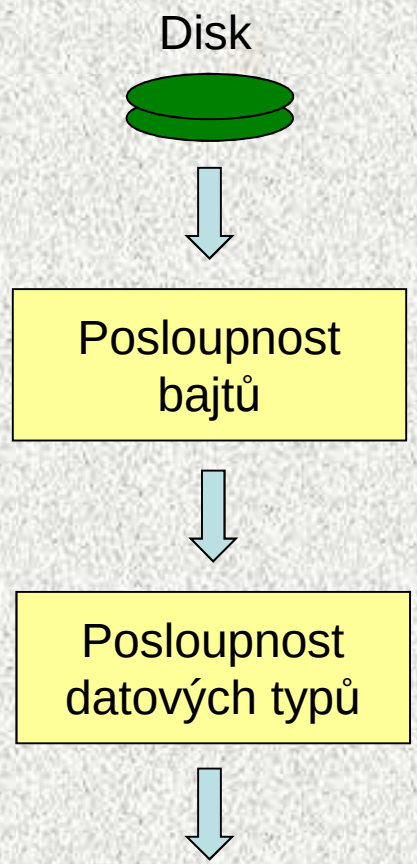
Postupný (sekvenční) přístup k datům v souboru

- Otevření souboru pro zápis nastaví souborové ukazovátko na začátek souboru
- Příkazem pro zápis se zapíše položka a souborové ukazovátko se pak automaticky posune na další prázdné místo
- Sekvenční zápis může probíhat až do „vyčerpání kapacity media“

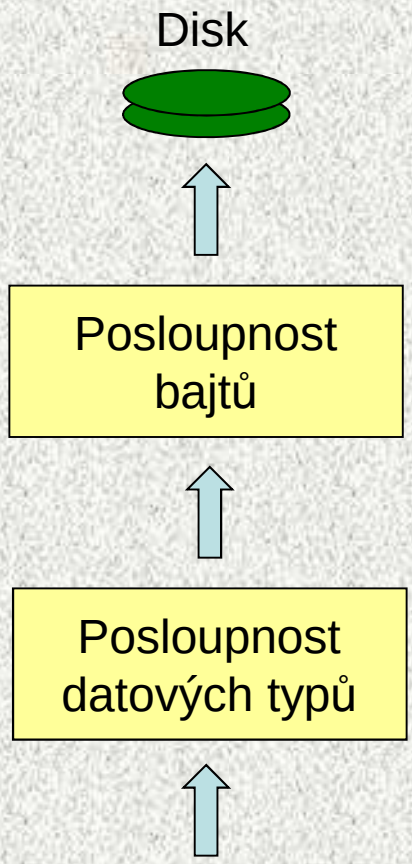


Přenos informace (v Javě) – dvě vrstvy

- Přenos informace z/do souboru je rozdělen do vrstev



Čtení ze souboru



Zápis do souboru

Soubory a proudy

- Java rozlišuje soubory (file) a proudy (stream)
  - soubory je množina údajů uložená ve vnější paměti počítače
  - proudy jsou nástroje k přenosu informací např. z/do souboru, ale také do/ze sítě, paměti, jiného programu, atd.
- informace může mít tvar znaků, bajtů, skupin bajtů (obrázky…), objektů,..
- přenos informace se děje dvoj/třívrstevně v proudech (streams):
  1. otevření přenosového proudu pro bajty či znaky
  2. otevření přenosového proudu pro datové typy Javy
  3. filtrace dat podle požadavků – bufferování, řádkování, …

Typy přenášených dat v souborech v Javě

Soubor **posloupnosti bytů**

Soubor **primitivních typů**

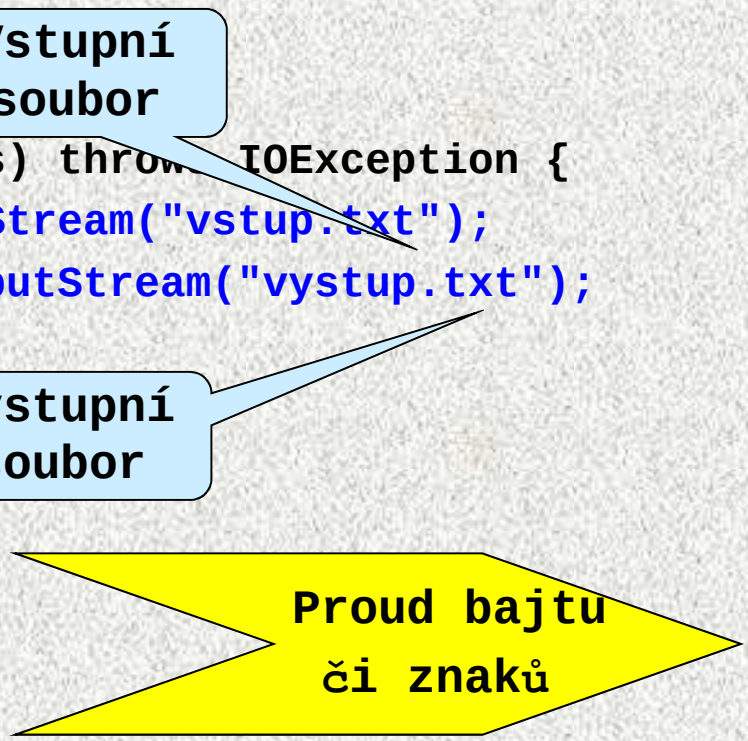
Soubor **řetězců**

Soubor **objektů**

1. Soubor jako **posloupnost bytů**

- V jazyku Java slouží pro práci se soubory třídy definované v balíku *java.io*
- Soubor jako posloupnost bytů reprezentují třídy:
  - ***FileInputStream*** vstupní soubor (bude se z něj číst)
  - ***FileOutputStream*** výstupní soubor (bude se do něj zapisovat)
- Příklad: kopie souboru

```
import java.io.*;
public class Kopie1 {
  public static void main(String[] args) throws IOException {
    FileInputStream in = new FileInputStream("vstup.txt");
    FileOutputStream out = new FileOutputStream("vystup.txt");
    int b = in.read();
    while (b!=-1) {
      out.write(b);
      b = in.read();
    }
    out.close();
    in.close();
  }
}
```



Soubor jako posloupnost bytů

- Komentář k příkazům
  - ► `FileInputStream in = new FileInputStream("vstup.txt");`
    - vytvořený objekt *in* reprezentuje vstupní soubor uložený na disku v aktuálním adresáři pod názvem *vstup.txt*; pokud takový soubor neexistuje, nastane chyba
  - ► `FileOutputStream out=new FileOutputStream("vystup.txt");`
    - vytvořený objekt *out* reprezentuje výstupní soubor, který bude uložen do aktuálního adresáře pod názvem *vystup.txt*; pokud by soubor nebylo možné vytvořit, nastane chyba

Metody:

- ► `b = in.read();`
  - ze souboru *in* se přečte jeden byte (číslo v rozsahu 0..255) a uloží do *b*; není-li v souboru žádný nepřečtený byte, výsledkem metody je –1
- ► `out.write(b);`
  - do souboru *out* se zapíše jeden byte s hodnotou *b*
- ► `out.close();`
- ► `in.close();`
  - uzavření souborů *in* a *out*

2. Soubor jako **posloupnost primitivních typů**

- Příklad: program, který vytvoří soubor obsahující 100 náhodných čísel typu *double* a pak soubor přečte a vypíše součet čísel

```
public class Cisla {
public static void main(String[] args)throws Exception {
DataOutputStream out = new DataOutputStream(
                              new FileOutputStream("temp.bin"));

for (int i=0; i<100; i++)
  out.writeDouble(Math.random());

out.close();

DataInputStream in = n                    utStream(
                              new FileInputStream("temp.bin"));

double soucet = 0;
while (in.available()>0)
   soucet = soucet + in.readDouble();
System.out.print(soucet);
}}
```

Výstupní soubor

Zavření souboru

Vstupní soubor

3. Soubor **primitivních typů a objektů**

- Soubor obsahující jak primitivní typy tak objekty reprezentují třídy **ObjectOutputStream a ObjectInputStream**
- Příklad: program, který vytvoří soubor obsahující dvě hodnoty typu *int* a dva řetězce, a pak soubor přečte a vypíše

```
public class CislaRetezce {
   public static void main(String[] args) throws Exception {
      ObjectOutputStream out = new ObjectOutputStream(
                               new FileOutputStream("temp.bin"));
      out.writeInt(1);
      out.writeInt(2);
      out.writeObject("prvni retez");
      out.writeObject("druhy retez");
      out.close();
      ObjectInputStream in = new ObjectInputStream(
                             new FileInputStream("temp.bin"));
      System.out.print(in.readInt()+" "+in.readInt());
      String s1 = (String)in.readObject();
      String s2 = (String)in.readObject();
      System.out.print (s1+" "+s2);
   }
}
```

Zavření souboru

Výstupní soubor

Vstupní soubor

Proud bajtu či znaků

Proud datových typů

Soubor primitivních typů a objektů, poznámka

Uvedenými metodami lze zapisovat a číst pouze tzv. serializovatelné objekty (patří mezi ně implicitně např. řetězce a pole primitivních typů), jinak je třeba „serializovat“ (implementovat rozhraní **`Serializable !`**)

4. Ukládání objektů do souboru

```
class ProObjekt implements Serializable {
  int i;
  String jmeno;
  int telefon;
  boolean pohlavi;
  double vaha;
  public ProObjekt(int j) {
    i = j;
    jmeno = "JMENO-" + j;
    telefon = 111 + j;
    pohlavi = i%2==0;
    vaha = 25 - i;
  }
public String toString() {
return (i+"\t"+ jmeno+"\t"+ telefon+"\t"+ pohlavi+"\t"+
   vaha);}
```

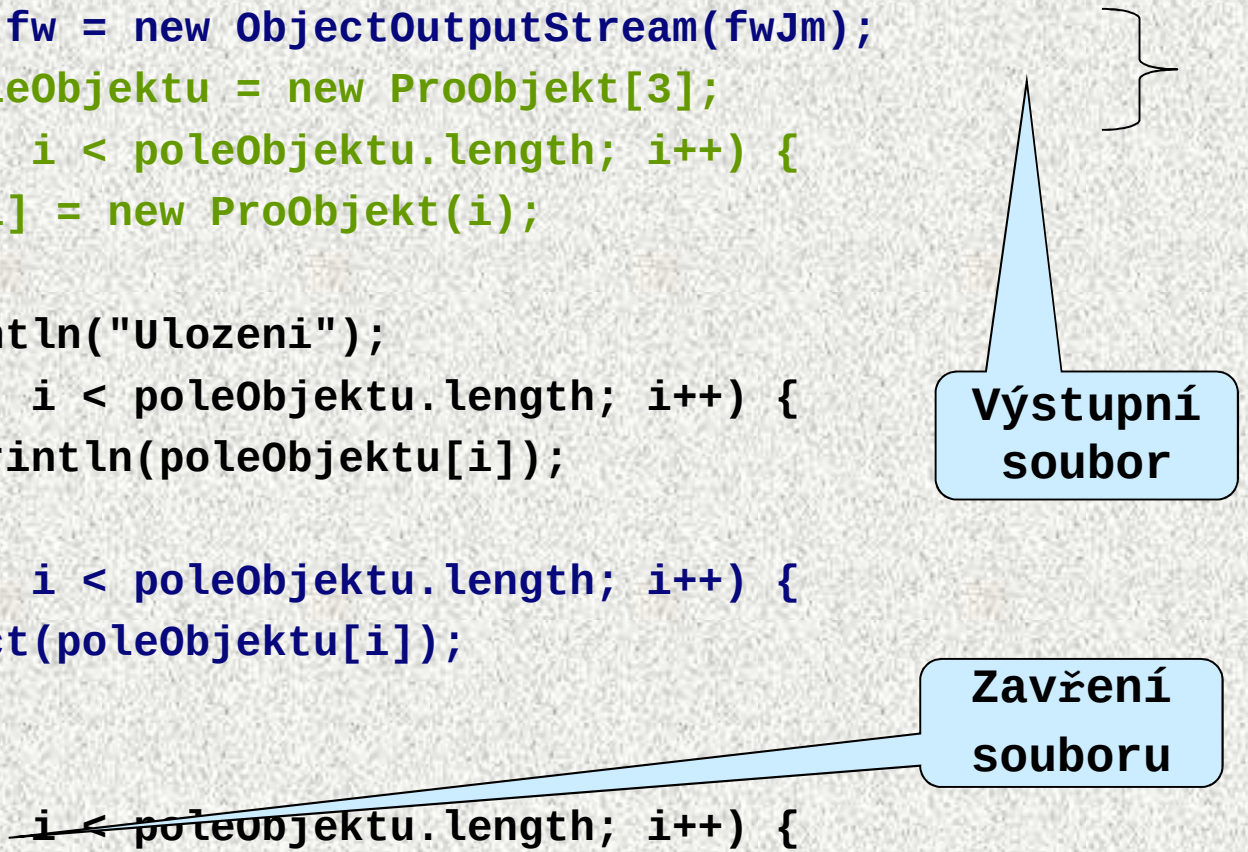
Datové složky

Konstruktor

Jak zobrazit objekt

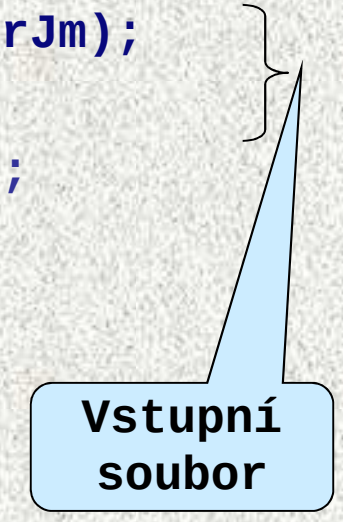
Ukládání objektů do souboru

```
public static void main(String[] args) throws IOException,
                                       ClassNotFoundException {
FileOutputStream fwJm = new FileOutputStream("objekty.bin");
ObjectOutputStream fw = new ObjectOutputStream(fwJm);
    ProObjekt[] poleObjektu = new ProObjekt[3];
    for (int i = 0; i < poleObjektu.length; i++) {
      poleObjektu[i] = new ProObjekt(i);
    }
    System.out.println("Ulozeni");
    for (int i = 0; i < poleObjektu.length; i++) {
      System.out.println(poleObjektu[i]);
    }
    for (int i = 0; i < poleObjektu.length; i++) {
      fw.writeObject(poleObjektu[i]);
    }
    fwJm.close();
    for (int i = 0; i < poleObjektu.length; i++) {
      poleObjektu[i] = null;}
```



Ukládání objektů do souboru

```
FileInputStream frJm = new FileInputStream("objekty.bin");
    ObjectInputStream fr = new ObjectInputStream(frJm);
    for (int i = 0; i < poleObjektu.length; i++) {
      poleObjektu[i] = (ProObjekt) fr.readObject();
    }
    fwJm.close();
    System.out.println("Cteni");
    for (int i = 0; i < poleObjektu.length; i++) {
      System.out.println(poleObjektu[i]);
    }
  }
}
```




```
Ulozeni
0       JMENO-0  111     true    25.0
1       JMENO-1  112     false   24.0
2       JMENO-2  113     true    23.0
Cteni
0       JMENO-0  111     true    25.0
1       JMENO-1  112     false   24.0
2       JMENO-2  113     true    23.0
```

Třídy pro práce se soubory v Javě

Třída (viz knihovna java.io.*)	Použití
File	Adresáře a vlastnosti souborů
Binární soubory, postupný přístup	
FileOutputStream	Zápis bajtů
FileInputStream	Čtení bajtů
DataOutputStream	Zápis jednoduchých typů
DataInputStream	Čtení jednoduchých typů
ObjectOutputStream	Zápis dat objektů
ObjectInputStream	Čtení dat objektů
Textové soubory, sekvenční přístup	
FileWriter	Zápis do textového souboru
FileReader	Čtení textového souboru
BufferedReader	Čtení textového souboru
Binární soubory, přímý (nahodilý) přístup	
RandomAccessFile	

- Při práci se souborem se nepoužívá jeho fyzické jméno (tj. jméno které zobrazuje souborový manažer – např. “Průzkumník”. Fyzickému jménu se přiřadí jméno (objekt) logické a s tím se dále pracuje viz třída File.