Curs 10 POO

© Conf. univ dr. Crenguta M. Puchianu

- Exceptii. Gestiunea exceptiilor
- Operatii de intrare/iesire. Scrierea datelor
- Operatii de intrare/iesire. Citirea datelor
- Clasa Scanner
- Serializarea/deserializarea obiectelor

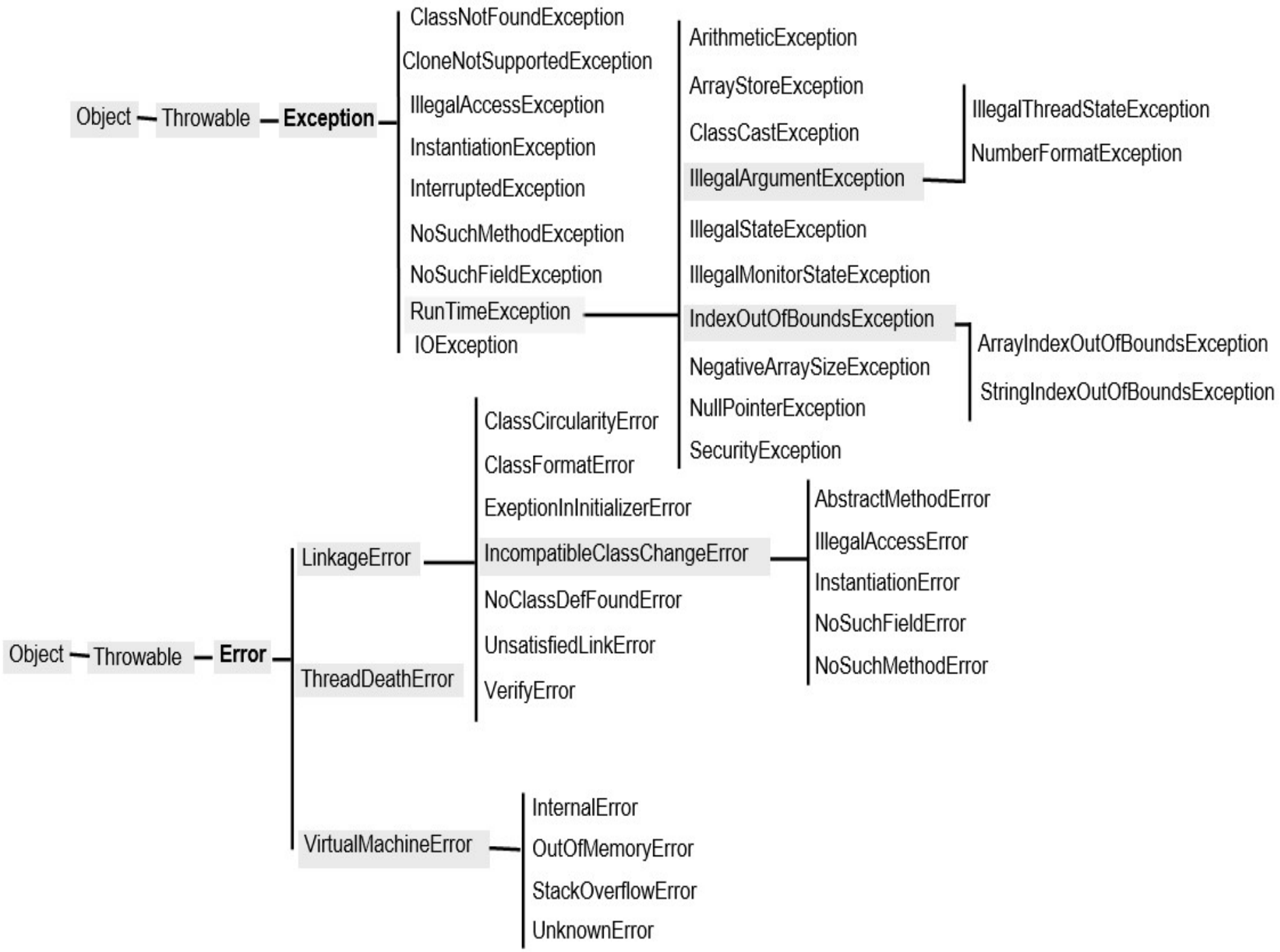
Gestiunea exceptiilor

- O *excepţie* este o situaţie excepţională care întrerupe execuţia unui program şi care poate fi cauzata de:
  - erori interne sau lipsa resurselor mediului de execuţie Java,
  - greseli de programare: cast greşit, referirea unui element al unui tablou care depăşeşte limita tabloului, referinţă la un obiect null, etc.

Exp: Persoana p; p.getNume()

- public class **Throwable**{

```
 public Throwable();
 public Throwable(String mesaj);
 public String getMessage();
 public String toString();
 public void printStackTrace();
 public void printStackTrace(java.io.PrintStream ps);
 public void printStackTrace(java.io.PrintWriter pw);
 public Throwable fillInStackTrace();
}
```


Object
Throwable
Exception
ClassNotFoundException
CloneNotSupportedException
IllegalAccessException
InstantiationException
InterruptedException
NoSuchMethodException
NoSuchFieldException
RunTimeException
IOException
ArithmeticException
ArrayStoreException
ClassCastException
IllegalArgumentException
IllegalThreadStateException
NumberFormatException
IllegalStateException
IllegalMonitorStateException
IndexOutOfBoundsException
ArrayIndexOutOfBoundsException
StringIndexOutOfBoundsException
NegativeArraySizeException
NullPointerException
SecurityException
Object
Throwable
Error
LinkageError
ClassCircularityError
ClassFormatError
ExeptionInInitializerError
IncompatibleClassChangeError
AbstractMethodError
IllegalAccessError
InstantiationError
NoSuchFieldError
NoSuchMethodError
NoClassDefFoundError
UnsatisfiedLinkError
VerifyError
ThreadDeathError
VirtualMachineError
InternalError
OutOfMemoryError
StackOverflowError
UnknownError

Tipuri de exceptii

Clasă excepţie	Motivul apariţiei
ArithmeticException	Erori matematice cum ar fi imparţirea la zero.
ArrayIndexOutOfBounds Exception	Indicele unui tablou este în afara tabloului.
ArrayStoreException	Se memorează într-un tablou o valoare de tip diferit de tipul de date al tabloului.
ClassNotFoundException	Se încearcă folosirea unei clase pe care compilatorul nu o cunoaşte. Clasa respectivă trebuie să fie importată sau scrisă de către programator.
FileNotFoundException	Se accesează un fişier care nu există în locaţia specificată de program.
IOException	Erori generale de I/O, cum ar fi incapacitatea de citi dintr-un fişier.
NullPointerException	Folosirea unei variabile referinta care nu refera niciun obiect.
NumberFormatException	O conversie eşuată între şiruri şi numere.
OutOfMemoryException	Memorie insuficientă creării unui obiect nou.
StackOverflowException	Depăşirea stivei de execuţie.
StringIndexOutOfBounds Exception	Programul incearcă să acceseze un caracter de pe o poziţie care nu există în şir.

Exemplu. Clasa Exceptie

```
1. public class Exceptie{
2. static void metoda1(int[] a){
3.   metoda2(a);
4. }
5. static void metoda2(int[] b){
6.   System.out.println(b[0]);
7. }
8. public static void main(String[] args){
9.  metoda1(null);
10. }
11.}
```

Executarea acestui program determină afişarea mesajului:

```
java.lang.NullPointerException
        at Exceptie.metoda2(Exceptie.java:6)
        at Exceptie.metoda1(Exceptie.java:3)
        at Exceptie.main(Exceptie.java:9)
```

Mesajul afişat conţine următoarele informaţii:

- tipul de exceptie aruncat,
- stiva apelurilor metodelor, fiecare cu clasa şi unitatea de compilare de care aparţine, împreună cu numărul liniei pe care se găseşte declaraţia metodei. Metoda din vârful stivei este cea care a aruncat excepţia respectivă.

Tratarea exceptiilor

- **Bloc try-catch**

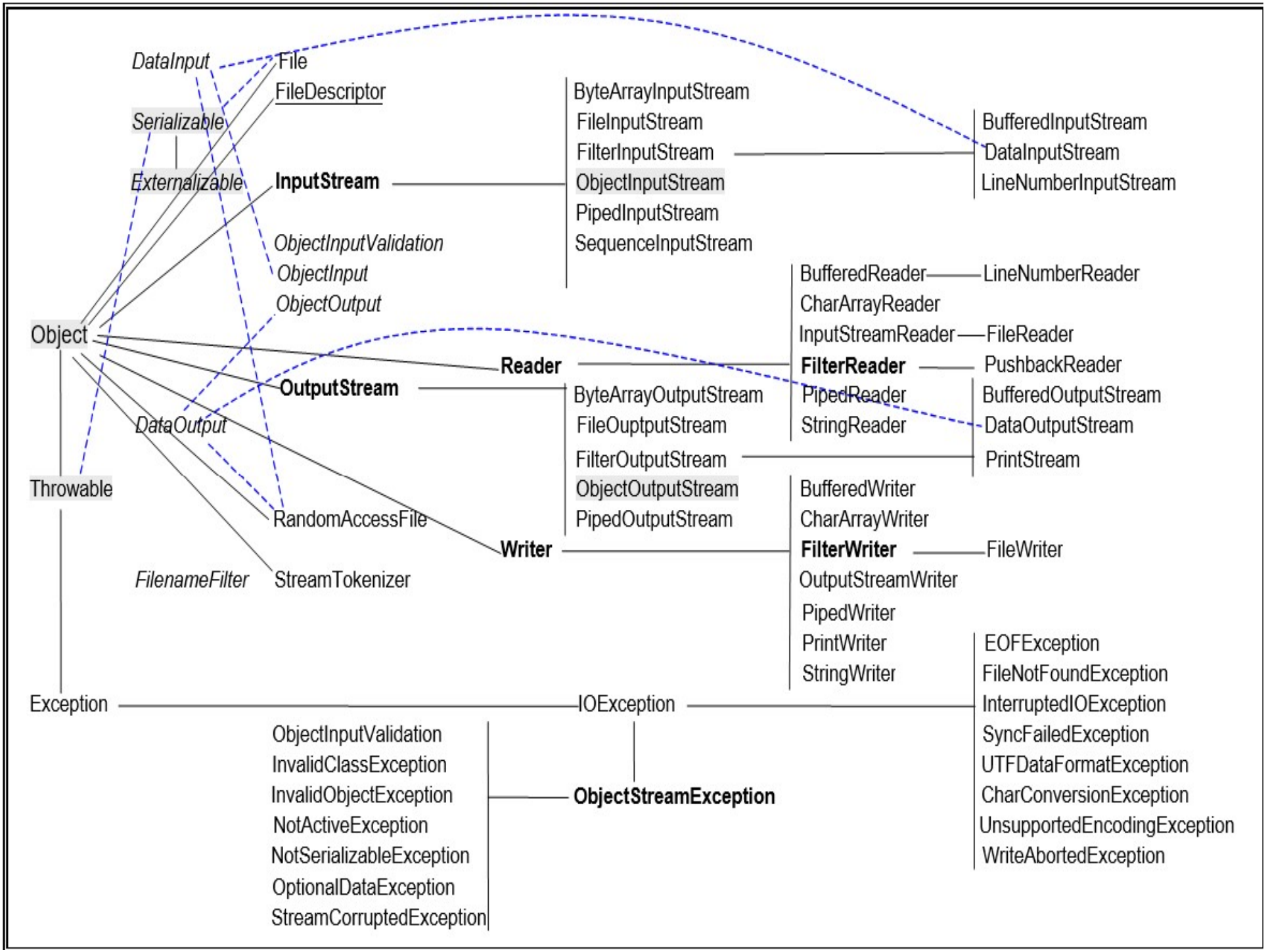
```
void metoda1(){
 try{
      //cod care poate produce o exceptie
 }catch(IOException|NumberFormatException ioe){//cod care gestioneaza
exceptia ioe}
  catch(Exception e){//cod care gestioneaza exceptia e}
[finally{
   //cod ce va fi executat indiferent daca apar sau nu exceptiile e sau ioe
  }]
}
```

- **Declaratii throws**

Modificatori NumeTip NumeMetoda(**[***ParametriFormali***]**) **throws** *ListaNumeTip***{**

```
        //lista de instructiuni
```

}


DataInput
File
FileDescriptor
Serializable
Externalizable
InputStream
ObjectInputValidation
ObjectInput
ObjectOutput
Object
Reader
OutputStream
DataOutput
Throwable
RandomAccessFile
Writer
FilenameFilter
StreamTokenizer
ByteArrayInputStream
FileInputStream
FilterInputStream
ObjectInputStream
PipedInputStream
SequenceInputStream
BufferedInputStream
DataInputStream
LineNumberInputStream
BufferedReader
LineNumberReader
CharArrayReader
InputStreamReader
FileReader
FilterReader
PushbackReader
PipedReader
StringReader
ByteArrayOutputStream
FileOuptputStream
FilterOutputStream
ObjectOutputStream
PipedOutputStream
BufferedOutputStream
DataOutputStream
PrintStream
BufferedWriter
CharArrayWriter
FilterWriter
FileWriter
OutputStreamWriter
PipedWriter
PrintWriter
StringWriter
Exception
IOException
ObjectInputValidation
InvalidClassException
InvalidObjectException
NotActiveException
NotSerializableException
OptionalDataException
StreamCorruptedException
ObjectStreamException
EOFException
FileNotFoundException
InterruptedIOException
SyncFailedException
UTFDataFormatException
CharConversionException
UnsupportedEncodingException
WriteAbortedException

Operatii de intrare/iesire. Scrierea datelor

- **Clasa OutputStream**

Prototip metodă	Efectul apelării metodei
abstract void write(int n) throws IOException	Scrie valoarea lui n pe fluxul de ieşire.
void write(byte[] b) throws IOException	Scrie pe fluxul de ieşire octeţii tabloului b.
void write(byte[] b, int poz, int nr) throws IOException	Scrie nr octeţi pe fluxul de ieşire începând de la poziţia poz a tabloului b.
void flush() throws IOException	Goleşte fluxul de ieşire.
void close() throws IOException	Închide fluxul de ieşire şi eliberează resursa alocată.

- Clasa **FileOutputStream** (prin constructor specificam fisierul de scriere)
- Clasa **PrintStream**
- Clasa **BufferedOutputStream**

Scrierea datelor. Clasa Writer

Prototip metodă	Efectul apelării metodei
void write(int n) throws IOException	Scrie valoarea lui n pe fluxul de ieşire.
void write(char[] b) throws IOException	Scrie pe fluxul de ieşire caracterele din tabloul b transmis ca parametru.
abstract void write(char[] b, int poz, int nr) throws IOException	Scrie nr caractere pe fluxul de ieşire începând de la poziţia poz a tabloului b.
public void write(String str) throws IOException	Scrie şirul str pe fluxul de ieşire.
public void write(String str, int poz, int nr) throws IOException	Scrie pe fluxul de ieşire subşirul de lungime nr pe care îl extrage de la poziţia poz a şirului str.

- Subclase: **FileWriter, PrintWriter, BufferedWriter**
- Clasa **OutputStreamWriter** – creeaza un obiect Writer care face legatura cu OutputStream

Scrierea datelor. Clasa PrintWriter

Prototip metodă	Efectul apelării metodei
PrintWriter append(char c)	Adaugă caracterul c pe fluxul de ieşire
PrintWriter append(CharSequence cs)	Adaugă secvenţa de caractere transmisă ca parametru actual pe fluxul de ieşire
PrintWriter append(CharSequence cs, int pozIn, int pozSf)	Adaugă secvenţa de caractere dintre poziţiile transmise ca parametri actuali pe fluxul de ieşire
PrintWriter format(String format, Object... args)	Scrie un şir formatat la ieşire
PrintWriter printf(String format, Object... args)	Scrie un şir formatat la ieşire
void print(xxx val), unde xxx={int, long, float, double, boolean, char, char[], String, Object}	Scrie valoarea parametrului actual la ieşire
void println(xxx val), unde xxx={int, long, float, double, boolean, char, char[], String, Object}	Scrie valoarea parametrului actual la ieşire după care trece la linia următoare

Citirea datelor. Clasa InputStream

Prototip metodă	Efectul apelării metodei
abstract int read() throws IOException	Returnează un octet citit de pe fluxul de intrare. Dacă se detectează sfârşitul fluxului, atunci metoda întoarce valoarea –1.
int read(byte[] b) throws IOException	Citeşte octeţi de pe fluxul de intrare şi îi memorează în tabloul b. Dacă b este mai mic decât numărul de octeţi citiţi, se lansează o excepţie. Metoda returnează numărul de octeţi citiţi efectiv.
int read(byte[] b, int poz, int nr) throws IOException	Citeşte maxim nr octeţi de pe fluxul de intrare şi îi memorează în tabloul b, de la poziţia poz. Dacă b este mai mic decât numărul de octeţi citiţi, se lansează o excepţie. Metoda returnează numărul de octeţi citiţi efectiv.
long skip(long n) throws IOException	Ignoră maxim n octeţi din fluxul de intrare şi întoarce numărul de octeţi cu care s-a avansat în flux.

Clasa Reader

Prototip metodă	Efectul apelării metodei
int read() throws IOException	Citeşte un caracter de pe fluxul de intrare şi returnează codul său Unicode. Dacă se detectează sfârşitul fluxului, metoda întoarce –1.
int read(char[] b) throws IOException	Citeşte caractere de pe fluxul de intrare şi îi memorează în tabloul b. Dacă b este mai mic decât numărul de caractere citite, se lansează o excepţie. Metoda returnează numărul de caractere citite efectiv.
boolean ready() throws IOException	Verifica daca fluxul este pregatit pentru citire (nu se blocheaza). Daca da, metoda returneaza true.

- Subclase: **BufferedReader**, **FileReader**

Clasa Scanner

Prototip metodă	Efectul apelării metodei
Scanner(InputStream sursa) Scanner(File sursa) Scanner(String sursa)	Creeaza un obiect Scanner ce produce valori din sursa transmisa ca parametru al constructorului
boolean hasNext() boolean hasNextInt() boolean hasNextDouble()	Returneaza true daca urmatorul token de pe fluxul de intrare este un String, int sau double
String next() int nextInt() float nextFloat() double nextDouble()	Returneaza urmatorul String, int, float sau double de pe fluxul de intrare
String nextLine()	Muta cursorul pe urmatoarea linie si returneaza linia anterioara
void close()	Inchide fluxul de citire

Serializarea obiectelor

- **Serializarea** este un mecanism prin care se salvează starea curentă a unui obiect ca un flux de octeţi. Acest flux poate fi salvat mai departe într-un fişier sau transmis în reţea şi poate fi reconstruit ulterior din fişier sau la destinaţie.

Solutia 1:

- Pas 1: Clasa trebuie sa implementeze interfaţa Serializable
- Pas 2: ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStream (numeFisier));
- Pas 3: oos.writeObject(a);

Solutia 2:

- Pas 1: XMLEncoder en=new XMLEncoder(new FileOutputStream ("document.xml"));
- Pas 2: en.writeObject(a);

Deserializarea obiectelor

- **Deserializarea** este mecanismul invers serializării şi are ca scop crearea unui obiect în memoria calculatorului pe care rulează programul. Starea obiectului este încărcată de exemplu din fişierul în care a fost salvată.

Solutia 1:

- Pas 1: ObjectInputStream ois=new ObjectInputStream(new FileInputStream("fisier.txt"));

- Pas 2: ois.readObject(); //returneaza Object

Solutia 2:

- Pas 1: XMLDecoder dec=new XMLDecoder(new FileInputStream ("document.xml"));
- Pas 2: dec.readObject(); // returneaza Object