

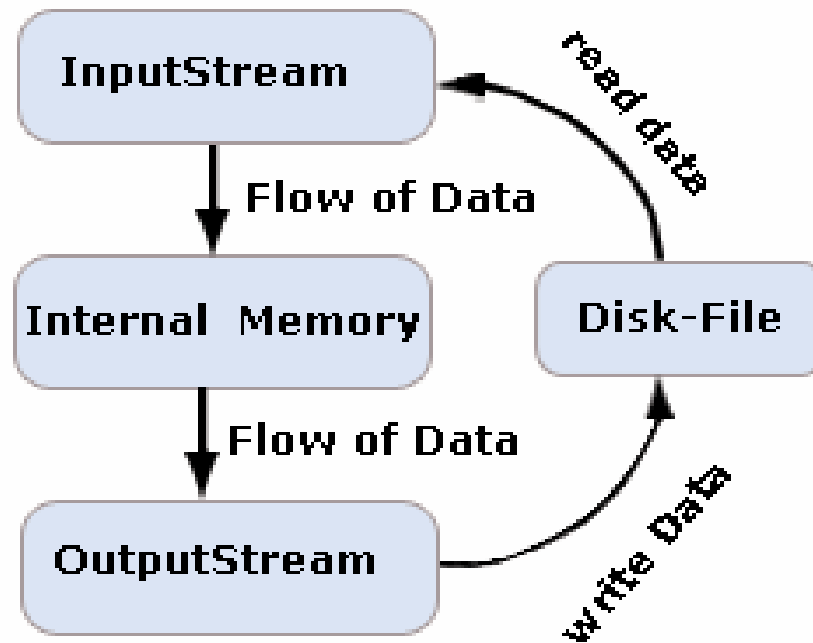
---

# 제11장 자바 입출력

---

Java\_source->src->ch11

# I/O Stream logic



# 서버와 클라이언트의 소켓 준비

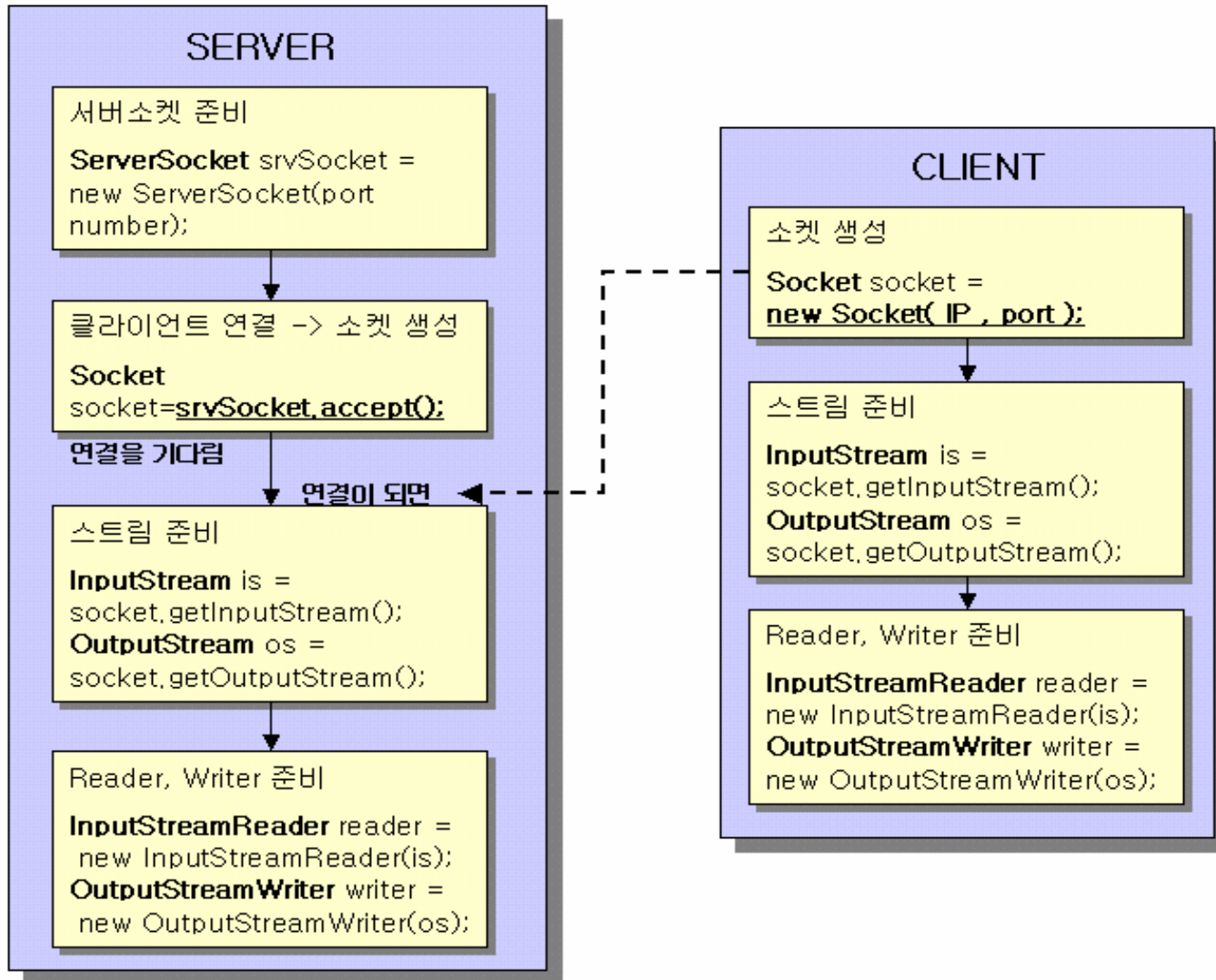
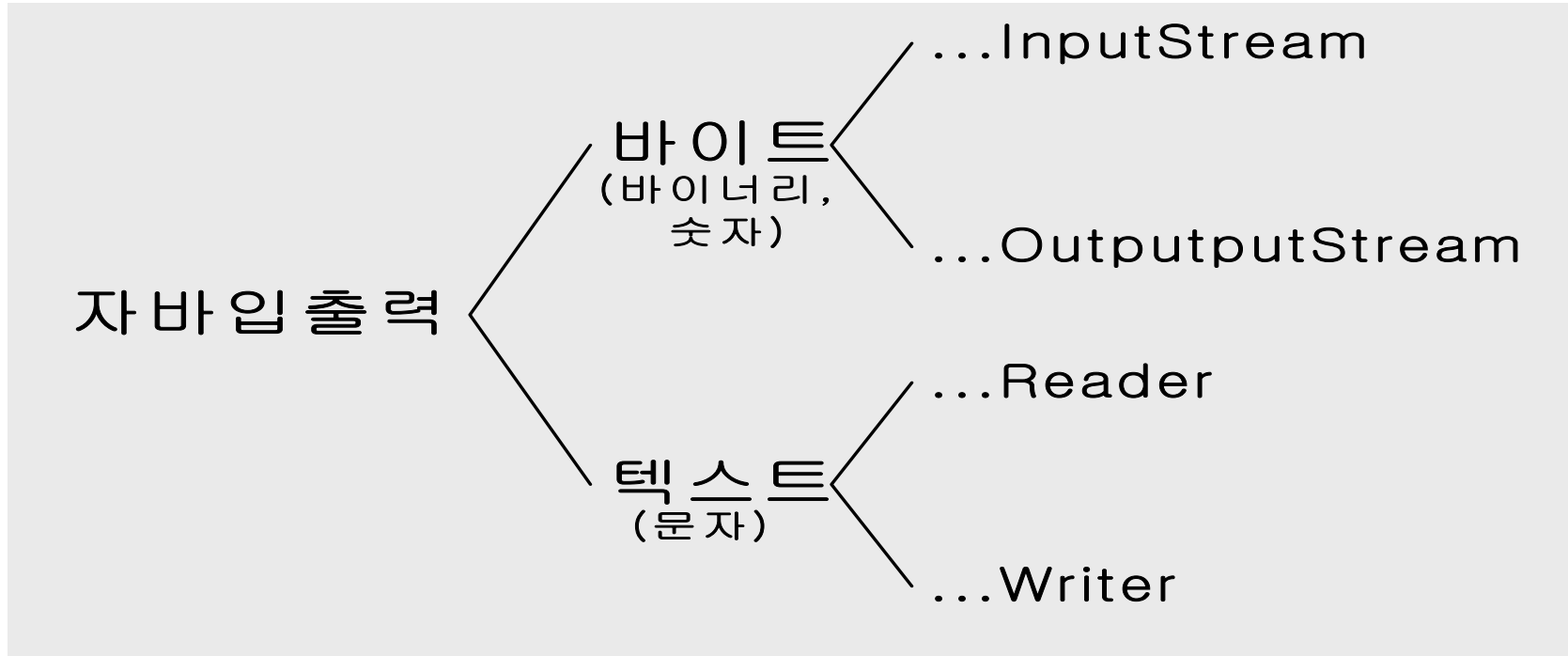


그림2-1-A) 서버와 클라이언트의 소켓 준비



## | 스트림

- 순서가 있고, 길이가 정해져 있지 않은 데이터 흐름
- 종류





# 11.1.1 스트림 개념

## I 바이트 스트림

- 바이너리(binary) 데이터와 숫자를 읽고, 쓰기 위해서 사용.
  - 예: 그림 파일, 동영상 파일, 응용프로그램 파일 등
- 바이트 단위로 입출력 작업





# 11.1.1 스트림 개념

## I 문자 스트림

- 문자 스트림은 문자 데이터를 읽고, 쓰기 위해서 사용.
- 인코딩을 자동으로 변환해줌





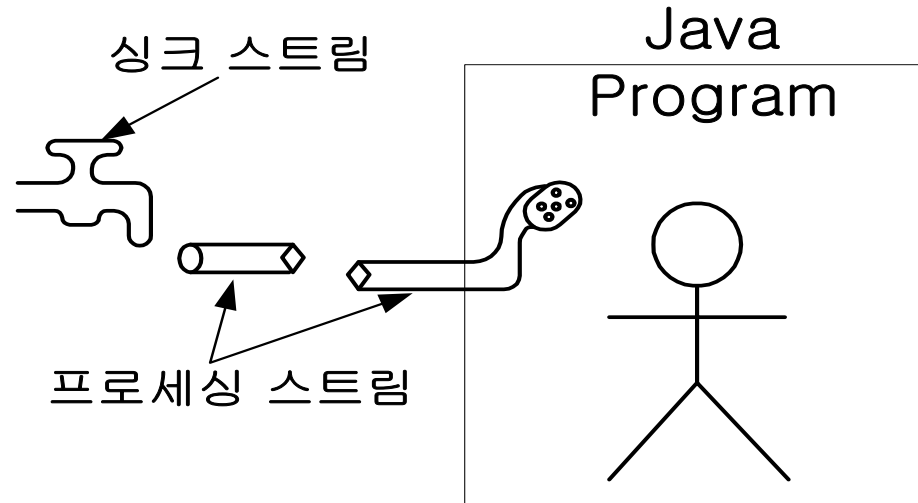
# 11.1.2 자바 입출력 클래스 분류와 상속 관계 Chap.11

## I 용도에 따른 분류

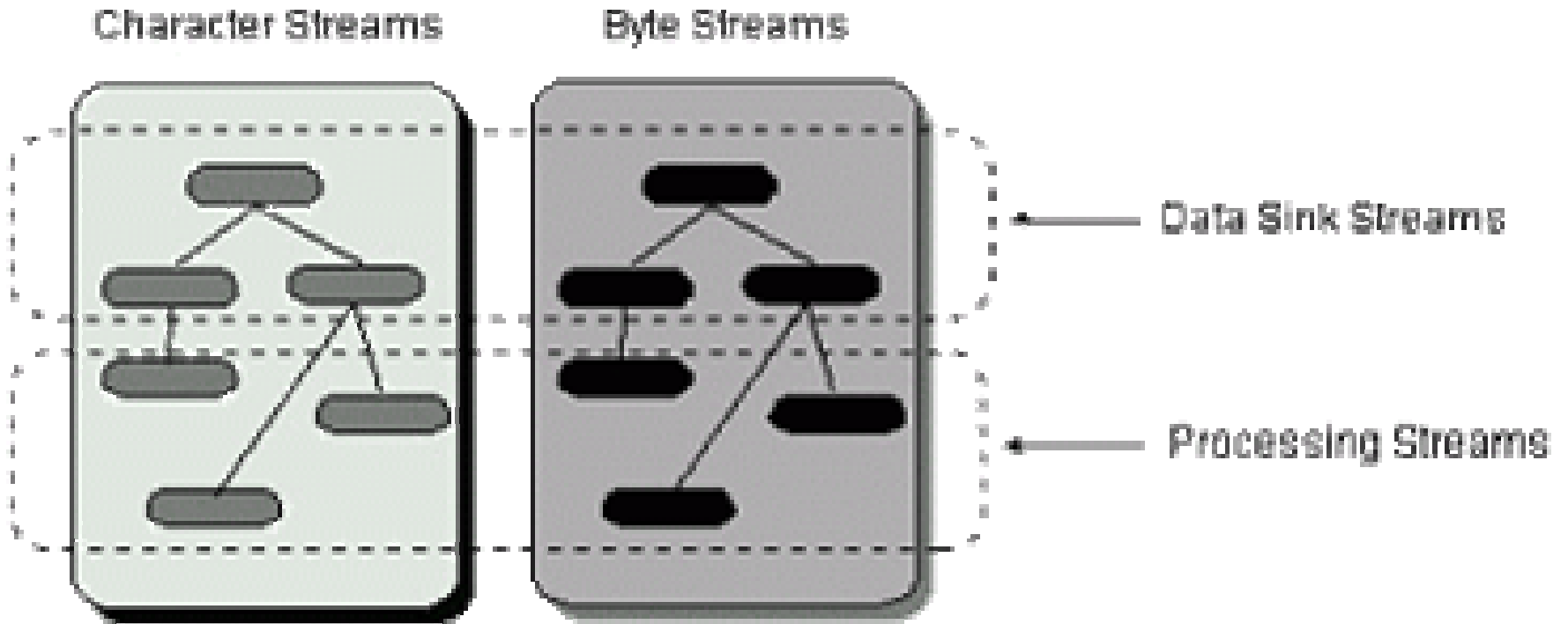
- 데이터 싱크 스트림
  - 데이터를 근원지에서 직접 읽거나, 목적지에 기록하는 클래스들
- 데이터 프로세싱 스트림
  - 다른 스트림을 이용해서 중간에서 어떤 작업을 수행하는 클래스들

## I 샤워의 비유

- 수도꼭지 -> 싱크 스트림
- 샤워기, 수도꼭지와 샤워기를 연결하는 것 -> 프로세싱 스트림



## 클래스들 분류







# 11.1.2 자바 입출력 클래스 분류와 상속 관계 Chap.11

## I 싱크스트림 클래스들

근원지/목적지	바이트 스트림	문자 스트림
메모리	ByteArrayInputStream, ByteArrayOutputStream	CharArrayReader, CharArrayWriter
	StringBufferInputStream	StringReader, StringWriter
파이프	PipedInputStream, PipedOutputStream	PipedReader, PipedWriter
파일	FileInputStream, FileOutputStream	FileReader, FileWriter

Console

System.in/out





# 11.1.2 자바 입출력 클래스 분류와 상속 관계 Chap.11

## I 데이터 프로세싱 스트림 클래스

- 데이터 프로세싱 스트림은 항상 다른 스트림을 이용해서만 생성가능
- 데이터 프로세싱 스트림의 생성자는 다른 스트림을 매개 변수로 받게됨

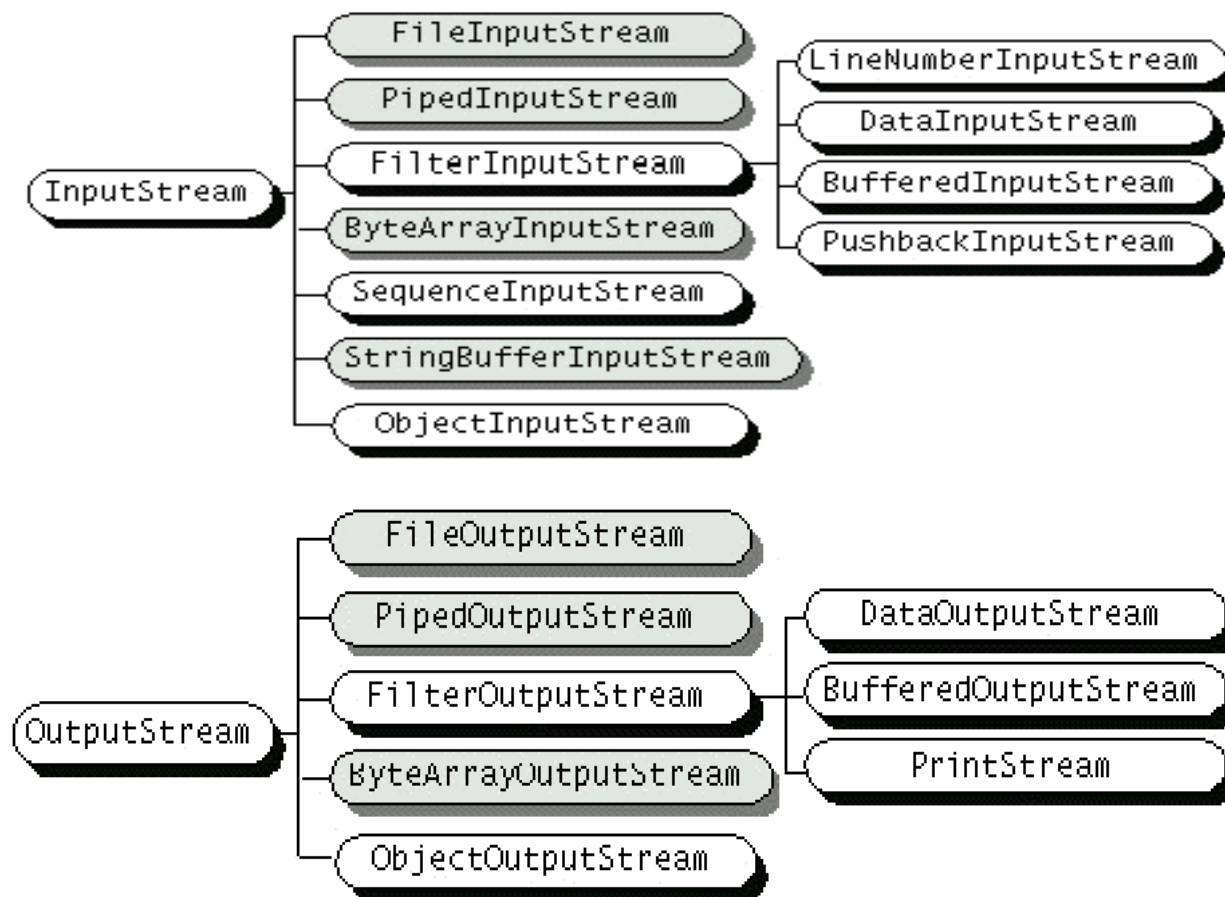
처 리	바이트 스트림	문자 스트림
버퍼링	BufferedInputStream, BufferedOutputStream	BufferedReader, BufferedWriter
필터링	FilterInputStream, FilterOutputStream	FilterReader, FilterWriter
바이트 스트림 -> 문자 스트림		InputStreamReader, OutputStreamWriter
여러 개의 스트림을 결합	SequenceInputStream	
객체 직렬화	ObjectInputStream, ObjectOutputStream	
자료 변환	DataInputStream, DataOutputStream	
프린트	PrintStream	PrintWriter



# 11.1.2 자바 입출력 클래스 분류와 상속 관계 Chap.11

## I 바이트 스트림 상속 관계

- `InputStream` 과 `OutputStream`으로 부터 상속받음.
- ✓ 회색은 싱크 스트림 클래스

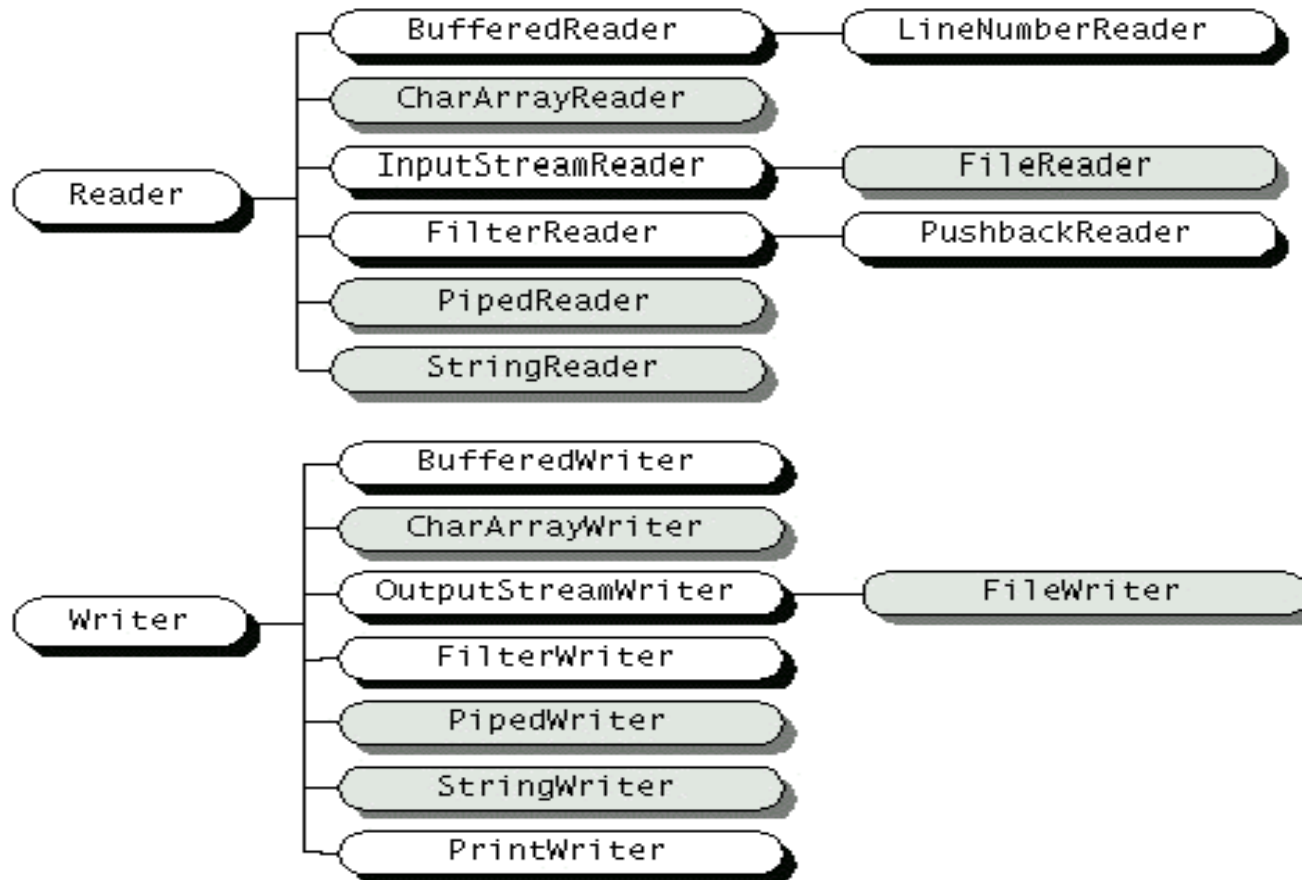




# 11.1.2 자바 입출력 클래스 분류와 상속 관계 Chap.11

## I 문자 스트림 상속 관계

- Reader와 Writer 클래스로부터 상속받는다.
- ✓ 회색은 싱크 스트림 클래스





# 11.2.1 InputStream/OutputStream 추상 클래스 Chap.11

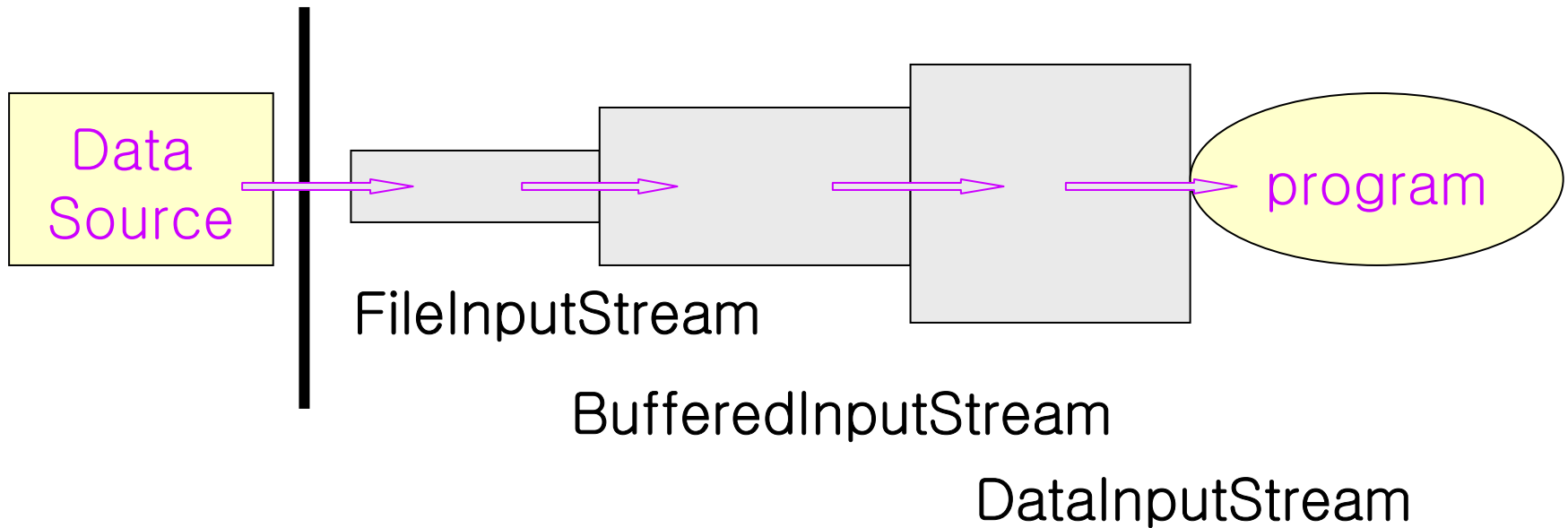
## I InputStream 클래스

- 바이트(byte) 입력 스트림을 위한 가장 상위 클래스.
- System.in
- 대표적인 메소드들
  - int read() - 입력 스트림에서 한 바이트를 읽어서 리턴.
  - int read(byte[] b, int off, int len) - 입력 스트림에서 len 개의 바이트를 읽어서 바이트 배열 b에 저장.
  - int read(byte[] b) - 배열의 크기만큼 읽는다. 읽은 바이트 수 리턴.





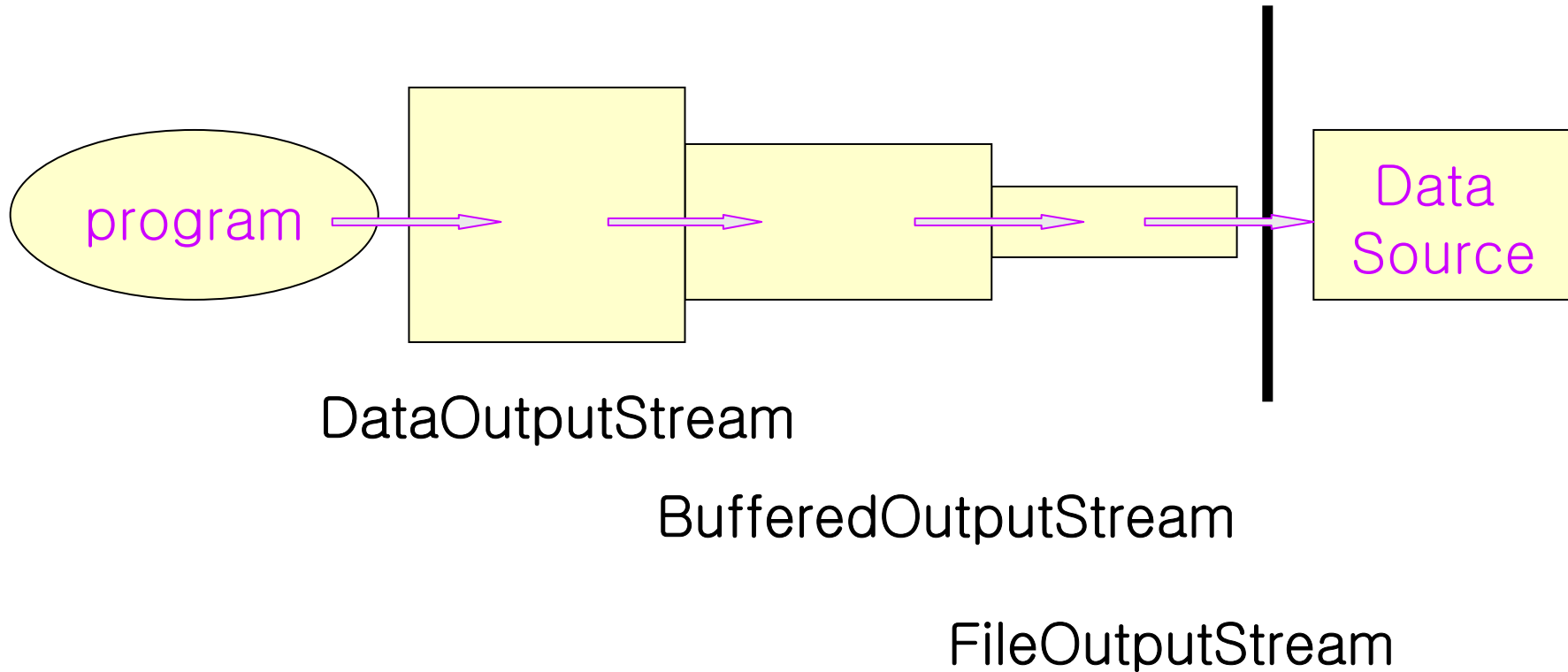
# Input Stream chain exam





# OutputStream chain exam

Chap.11





# 11.2.1 InputStream/OutputStream 추상 클래스 Chap.11

## I 예제: SimpleRead.java

```
8      try {
9          b = System.in.read();
10         while(b != -1) {
11             System.out.print((char)b);
12             count++;
13             b = System.in.read();
14         }
15     } catch(IOException e) {
16         System.out.println(e);
17     }
18     System.out.println();
19     System.out.println("total byte:" + count);
```

### I결과

```
C:\> java SimpleRead
```

```
Hello
```

```
Hello
```

```
한글
```

```
??±?
```

```
^Z
```

```
total bytes:13
```







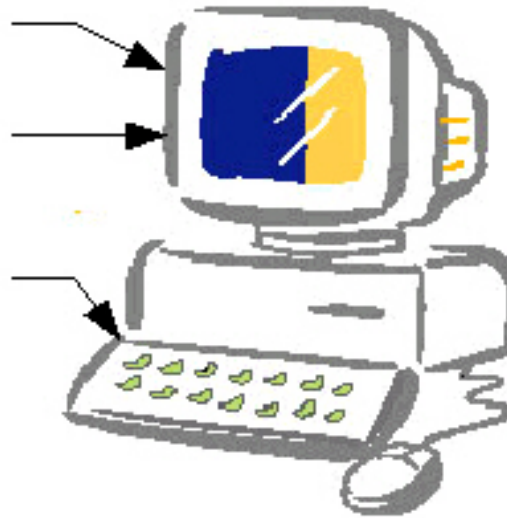
# 11.2.1 InputStream/OutputStream 추상 클래스 Chap.11

## | 콘솔

System.out

System.err

System.in



## | 리다이렉션

```
C:\W> java SimpleRead < SimpleRead.java
import java.io.IOException;
...
```

```
C:\W> java SimpleRead < SimpleRead.java > output.txt
C:\W> more output.txt
import java.io.IOException;
```





# 11.2.1 InputStream/OutputStream 추상 클래스 Chap.11

## | ConsoleReader 클래스

- 콘솔에서 데이터를 쉽게 읽기 위한 사용자 정의 클래스

## | 예제: ConsoleReader.java

```
3 public class ConsoleReader extends BufferedReader {
4     public ConsoleReader(Reader in) throws IOException {
5         super(in);
6     }
7
8     public ConsoleReader(InputStream in) throws IOException {
9         super(new InputStreamReader(in));
10    }
11
12    public int readInt() throws IOException {
13        return Integer.parseInt(readLine().trim());
```



# 11.2.1 InputStream/OutputStream 추상 클래스 Chap.11

## I 예제: ConsoleReader.java(계속)

```
16 public String readString() throws IOException {
17     return readLine().trim();
18 }
19
20 public char readChar() throws IOException {
21     String line = readLine().trim();
22     return line.charAt(0);
23 }
24
25 public double readDouble() throws IOException {
26     Double d = new Double(readLine().trim());
27     return d.doubleValue();
```





# 11.2.1 InputStream/OutputStream 추상 클래스 Chap.11

## I 예제: ConsoleReaderTest.java

```
11      ConsoleReader in = new ConsoleReader(System.in);
12      System.out.print("정수를 입력해주세요:");
13      ivalue = in.readInt();
14      System.out.println("value = " + ivalue);
15      System.out.print("\n실수를 입력해주세요:");
16      dvalue = in.readDouble();
17      System.out.println("value = " + dvalue);
18      System.out.print("\n문자를 입력해주세요:");
19      cvalue = in.readChar();
20      System.out.println("value = " + cvalue);
21      System.out.print("\n문자열을 입력해주세요:");
22      svalue = in.readString();
23      System.out.println("value = " + svalue);
```



# 11.2.1 InputStream/OutputStream 추상 클래스 Chap.11

## I Copier 클래스

- 입력 스트림에서 읽은 데이터를 출력 스트림으로 출력하는 사용자 정의 클래스

## I 예제: Copier.java

```
3 public class Copier {
4     protected InputStream in;
5     protected OutputStream out;
6     protected byte data[];
7
8     public Copier(InputStream in, OutputStream out, int sz) {
9         this.in = in;
10        this.out = out;
11        data = new byte[sz];
```



# 11.2.1 InputStream/OutputStream 추상 클래스 Chap.11

## I 예제: Copier.java(계속)

```
14     public Copier(InputStream in, OutputStream out) {
15         this(in, out, 1024);
16     }
17
18     public void copy() throws IOException {
19         int n = 0;
20         while((n = in.read(data)) != -1) {
21             out.write(data, 0, n);
22         }
23         in.close();
24         out.close();
```





## I FileInputStream

- 파일에서 바이트 단위로 내용을 읽기 위해 사용
- 생성자
  - FileInputStream(File file)
  - FileInputStream(String name)

## I FileOutputStream

- 파일로 바이트 단위로 내용을 출력하기 위해 사용
- 생성자
  - FileOutputStream(File file)
  - FileOutputStream(String name, boolean append)
  - FileOutputStream(String name)





# 11.2.2 FileInputStream/FileOutputStream

## I 예제: ConsoleCopy.java

```
4 public static void main(String args[
5     if(args.length != 1) {
6         System.out.println(
7             "usage: java ConsoleCopy
8         System.exit(1);
9     }
10    try {
11        Copier cp = new Copier(System.in,
12            new FileOutputStream(args[0]));
13        cp.copy();
```

### I결과

```
C:\> java ConsoleCopy abc.txt
한글입니다.
^Z

C:\> more abc.txt
한글입니다.
```





# 11.2.2 FileInputStream/FileOutputStream

## I 예제: More.java

### I 결과

```
C:\> java More More.java
import java.io.*;

public class More {
.....
```

```
4 public static void main(String args[]) {
5     if(args.length != 1) {
6         System.out.println(
7             "usage: java More <filename>");
8         System.exit(1);
9     }
10    try {
11        Copier cp = new Copier(new FileInputStream(args[0]),
12                                System.out);
13        cp.copy();
```





# 11.2.2 FileInputStream/FileOutputStream

## I 예제: FileCopy.java

```
5     if(args.length != 2) {
6         System.out.print("usage: java
7         System.out.println("<filename
8         System.exit(1);
9     }
10    try {
11        Copier cp = new Copier(new FileInputStream(args[0]),
12                                new FileOutputStream(args[1]));
13        cp.copy();
14        System.out.println("파일 복사가 완료되었습니다.");
```

### I결과

C:\> java FileCopy abc.txt bc.txt

파일 복사가 완료되었습니다.

C:\> more bc.txt

한글입니다.





# 11.2.3 FileReader/FileWriter

## | FileReader

- 파일에서 문자 단위로 내용을 읽을 때 사용
- 생성자
  - `FileReader(File file)`
  - `FileReader(String fileName)`

## | FileWriter

- 파일에 문자 데이터를 출력할 때 사용
- 생성자
  - `FileWriter(File file)`
  - `FileWriter(String fileName, boolean append)`
  - `FileWriter(String fileName)`





# 11.2.3 FileReader/FileWriter

## I 예제: ReadFileData.java

```
10     try {
11         ConsoleReader in = new ConsoleReader(
12             new FileReader("data.txt"));
13         ivalue = in.readInt();
14         System.out.println("value = " + ivalue);
15         dvalue = in.readDouble();
16         System.out.println("value = " + dvalue);
17         cvalue = in.readChar();
18         System.out.println("value = " + cvalue);
19         svalue = in.readString();
20         System.out.println("value = " + svalue);
```





# 11.2.4 InputStreamReader/OutputStreamWriter Chap.11

## I InputStreamReader

- 입력 스트림을 Reader 타입으로 변환
- 생성자
  - InputStreamReader(InputStream in, String enc)
  - InputStreamReader(InputStream in)

## I OutputStreamWriter

- 출력 스트림을 Writer 클래스로 변환
- 생성자
  - OutputStreamWriter(OutputStream out, String enc)
  - OutputStreamWriter(OutputStream out)





# 11.2.5 PrintStream/PrintWriter

Chap.11

## I PrintStream

- 데이터 출력(예: `System.out`, `System.err`)
- 생성자
  - `PrintStream(OutputStream out, boolean autoFlush)`
  - `PrintStream(OutputStream out)`

## I PrintWriter

- 문자 데이터 출력
- 생성자
  - `PrintWriter(OutputStream out, boolean autoFlush)`
  - `PrintWriter(OutputStream out)`
  - `PrintWriter(Writer out, boolean autoFlush)`
  - `PrintWriter(Writer out)`





# 11.2.5 PrintStream/PrintWriter

## | ConsoleWriter

- 출력을 보다 편리하게 할 수 있는 사용자 정의 클래스

## | 예제: ConsoleWriter.java

```
4 public class ConsoleWriter extends PrintWriter {
5     public ConsoleWriter(OutputStream out) {
6         super(out, true);
7     }
8     ...
9     public ConsoleWriter(Writer out) {
10        super(out, true);
11    }
12    ...
13    public void printWon(long value) {
14        String format = "W#,###";
15        DecimalFormat df = new DecimalFormat(format);
16        print(df.format(value));
17    }
18 }
```



# 11.2.5 PrintStream/PrintWriter

## I 예제: ConsoleWriter.java(계속)

```
24     public void printDollar(float value) {
25         String format = "$#,###.00";
26         DecimalFormat df = new DecimalFormat(format);
27         print(df.format(value));
...
35     public void print(double value, int digit) {
36         String format = "#.";
37         for(int i=0; i < digit; i++) {
38             format += "#";
39         }
40         DecimalFormat df = new DecimalFormat(format);
41         print(df.format(value));
42         flush();
```





# 11.2.5 PrintStream/PrintWriter

## I 예제: ConsoleWriterTest.java

```
5      ConsoleWriter out = new ConsoleWriter(System.out);
6      double d = 123.456789;
7      int money = 1234567;
8
9      out.println(d);
10     out.println(d, 2);
11     out.printlnWon(money);
12     out.printlnDollar(money);
```

### I결과

```
C:\> java ConsoleWriterTest
123.456789
123.46
W1,234,567
$1,234,567.00
```





## 11.2.6 BufferedInputStream/BufferedOutputStream Chap.11

### I BufferedInputStream

- 바이트 입력에서 성능을 높이기 위해서 버퍼를 사용하는 클래스
- 생성자
  - BufferedInputStream(InputStream in, int size)
  - BufferedInputStream(InputStream in)

### I BufferedOutputStream

- 바이트 출력에서 성능을 높이기 위해서 버퍼를 사용하는 클래스
- 생성자
  - BufferedOutputStream(OutputStream out, int size)
  - BufferedOutputStream(OutputStream out)





# 11.2.7 BufferedReader/BufferedWriter Chap.11

## | BufferedReader

- 버퍼를 사용하는 Reader
- 생성자 및 중요 메소드
  - `BufferedReader(Reader in, int sz)`
  - `BufferedReader(Reader in)`
  - `String readLine()`

## | BufferedWriter

- 버퍼를 사용하는 Writer
- 생성자
  - `BufferedWriter(Writer out, int sz)`
  - `BufferedWriter(Writer out)`





# 11.2.7 BufferedReader/BufferedWriter Chap.11

## I 예제: ConfigurationReader.java

```
28     BufferedReader br =
29         new BufferedReader(new FileReader(file) );
30
31     while((rl=br.readLine()) != null) {
32         rl = rl.trim();
33         if(rl.charAt(0) == comment )    {
34             continue;
35         } else {
36             StringTokenizer st =
37                 new StringTokenizer(rl,delm,false);
38             String key = st.nextToken();
39             String value = st.nextToken();
40             ht.put(key, value);
```



# 11.2.7 BufferedReader/BufferedWriter Chap.11

## I 예제: ConfigurationReader.java(계속)

```
46     public String getValue(String name) {
47         return (String)ht.get(name);
48     }
49
50     public static void main(String args[])
51         ConfigurationReader cr =
52         new ConfigurationReader("server.conf");
53     try {
54         cr.parse();
55         System.out.println(cr.getValue("port"));
56         System.out.println(cr.getValue("server"));
57         System.out.println(cr.getValue("admin"));
```

### I결과

```
C:\> java ConfigurationReader
7690
it.mokpo.ac.kr
admin@it.mokpo.ac.kr
```





## I DataInputStream

- 데이터 타입별로 읽을 때 사용
- 생성자 및 중요 메소드
  - DataInputStream(InputStream in)  
char readChar()  
double readDouble()  
int readInt()

## I DataOutputStream

- 데이터 타입별로 출력
- 생성자 및 중요 메소드
  - DataOutputStream(OutputStream out)  
void writeChar(int v)  
void writeInt(int v)





## 11.2.8 DataInputStream/DataOutputStream

Chap.11

### I 예제: DataIOTest.java

```
12     public void write() throws IOException {
13         DataOutputStream out = new
14             DataOutputStream(new FileOutputStream(file));
15         int value[] = Fibonacci.calculate(n);
16         for(int i=0; i <= n; i++) {
17             out.writeInt(value[i]);
18             //System.out.println(value[i]);
19         }
20         out.flush();
21         out.close();
```





## 11.2.8 DataInputStream/DataOutputStream

### I 예제: DataIOTest.java(계속)

```
24     public void read() throws IOException {
25         DataInputStream in =
26             new DataInputStream(new FileInputStream(file));
27         int n = in.available()/4;
28         //System.out.println("# of data = " + n);
29         for(int i=0; i < n; i++) {
30             int v = in.readInt();
31             System.out.println("f["+i+"] = " + v);
32         }
33         in.close();
34     }
```

