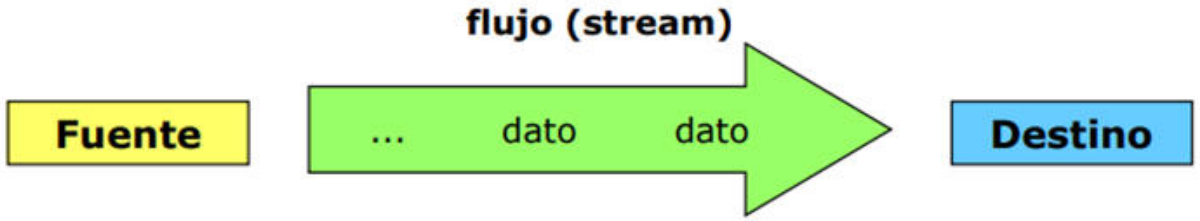
E/S con flujos (streams)

- En Java se define la abstracción de ***stream*** (flujo) para tratar la comunicación de información entre el programa y el exterior
  - Entre una fuente y un destino fluye una secuencia de datos
- Los flujos actúan como interfaz con el dispositivo o clase asociada
  - Operación independiente del tipo de datos y del dispositivo
  - Mayor flexibilidad (p.e. redirección, combinación)
  - Diversidad de dispositivos (fichero, pantalla, teclado, red, …)
  - Diversidad de formas de comunicación
    - Modo de acceso: secuencial, aleatorio
    - Información intercambiada: binaria, caracteres, líneas



Flujos estándar

- Como en Unix:
  - Entrada estándar - habitualmente el teclado
  - Salida estándar - habitualmente la consola
  - Salida de error - habitualmente la consola
- En Java se accede a la E/S estándar a través de campos estáticos de la clase ***java.lang.System***
  - ***System.in*** implementa la entrada estándar
  - ***System.out*** implementa la salida estándar
  - ***System.err*** implementa la salida de error

Flujos estándar

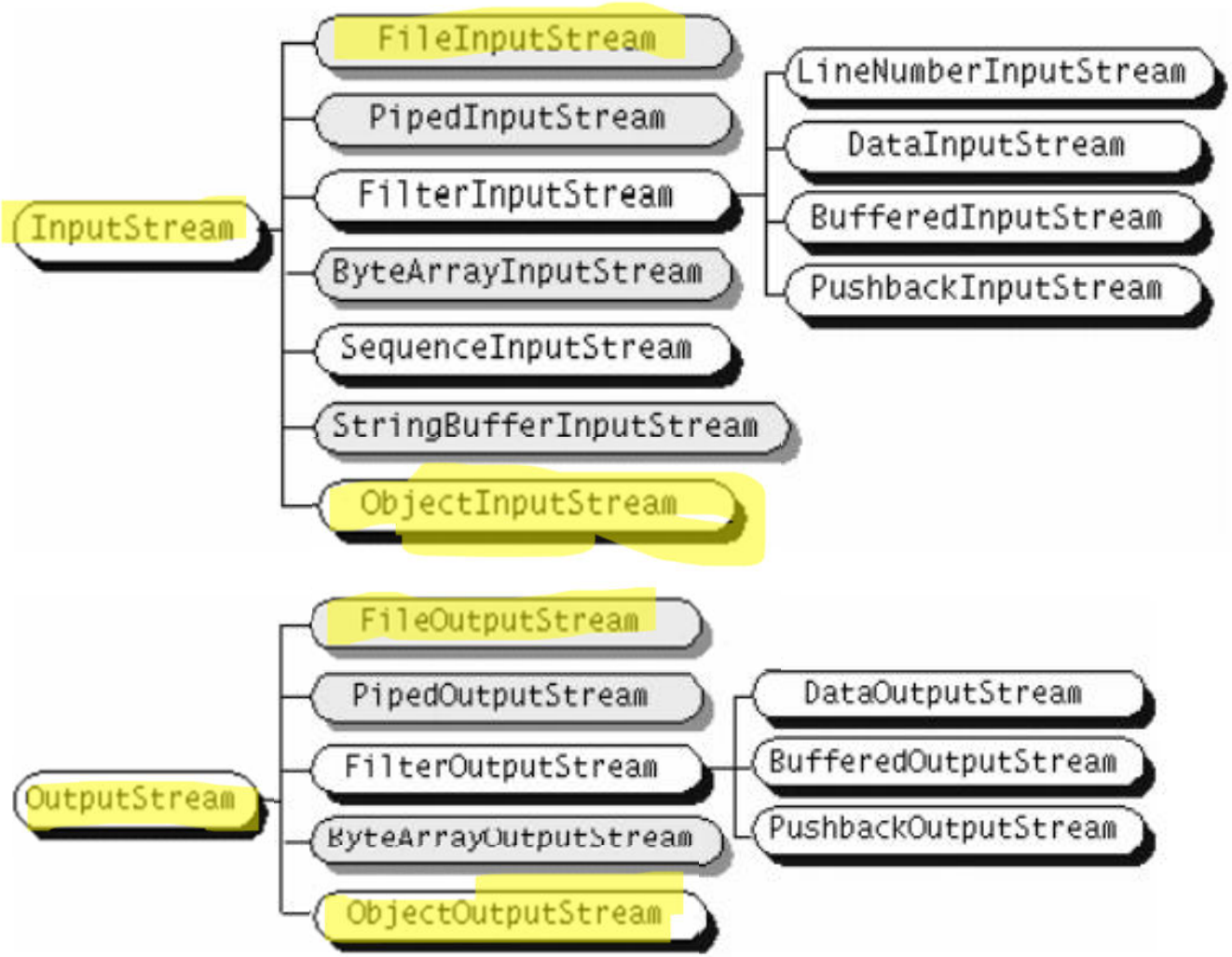
- ***System.in***
  - Instancia de la clase ***InputStream***: flujo de bytes de entrada
  - Metodos
    - ***read()*** permite leer un byte de la entrada como entero
    - ***skip(n)*** ignora n bytes de la entrada
    - ***available()*** número de bytes disponibles para leer en la entrada
- ***System.out***
  - Instancia de la clase ***PrintStream***: flujo de bytes de salida
  - Metodos para impresión de datos
    - ***print(), println()***
    - ***flush() vacía el buffer de salida escribiendo su contenido***
- ***System.err***
  - Funcionamiento similar a ***System.out***
  - ***Se utiliza para enviar mensajes de error*** (por ejemplo a un fichero de log o a la consola)

Utilización de los flujos

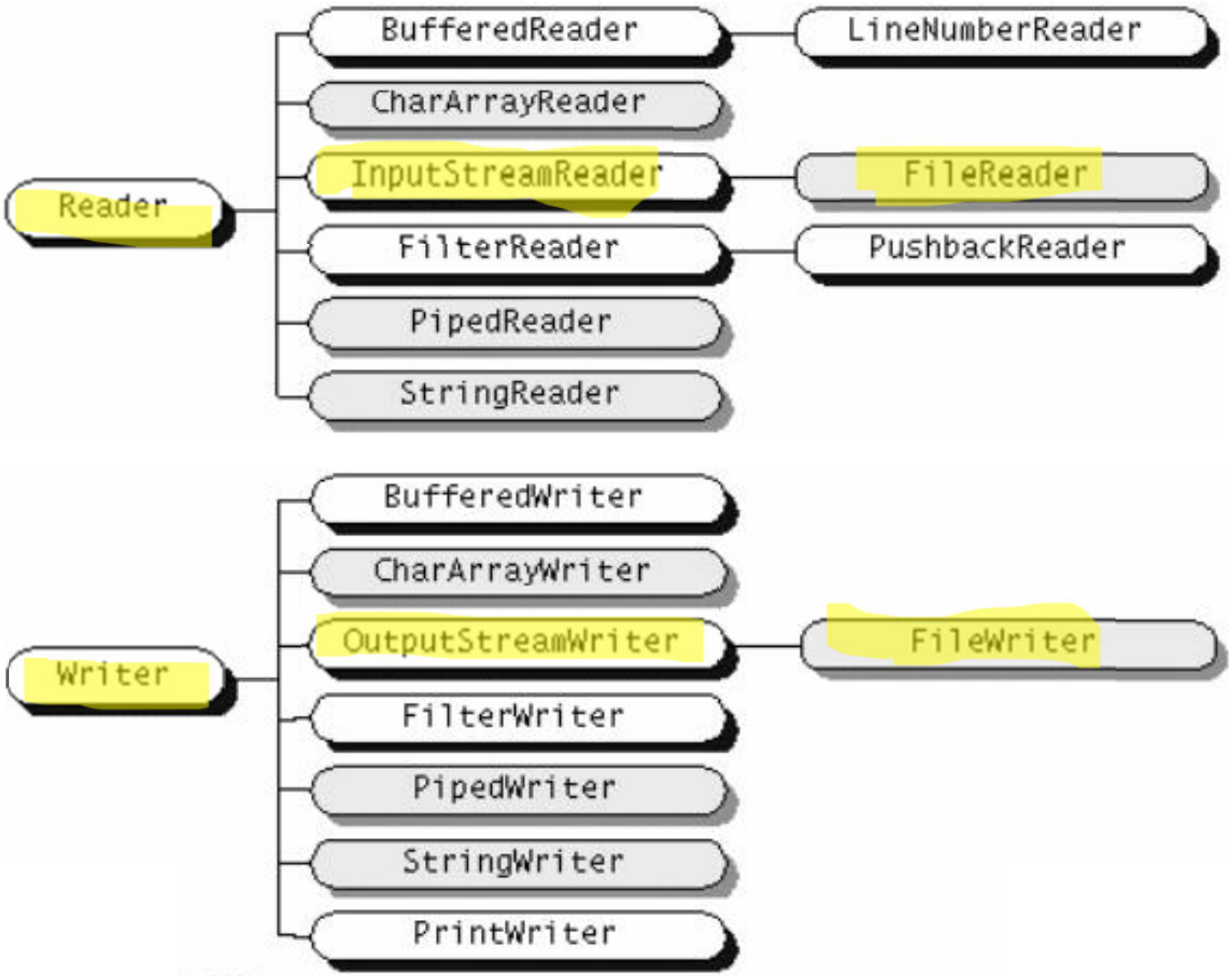
- Lectura
  1. Abrir un flujo a una fuente de datos (creación del objeto stream)
     - Teclado
     - Fichero
     - Socket remoto
  2. Mientras existan datos disponibles
     - Leer datos
  3. Cerrar el flujo (método close)
- Escritura
  1. Abrir un flujo a una fuente de datos (creación del objeto stream)
     - Pantalla
     - Fichero
     - Socket local
  2. Mientras existan datos disponibles
     - Escribir datos
  3. Cerrar el flujo (método close)

- ***Nota: para los flujos estándar ya se encarga el sistema de abrirlos y cerrarlos***
- Un fallo en cualquier punto produce la excepción `IOException`

Jerarquía de flujos de bytes

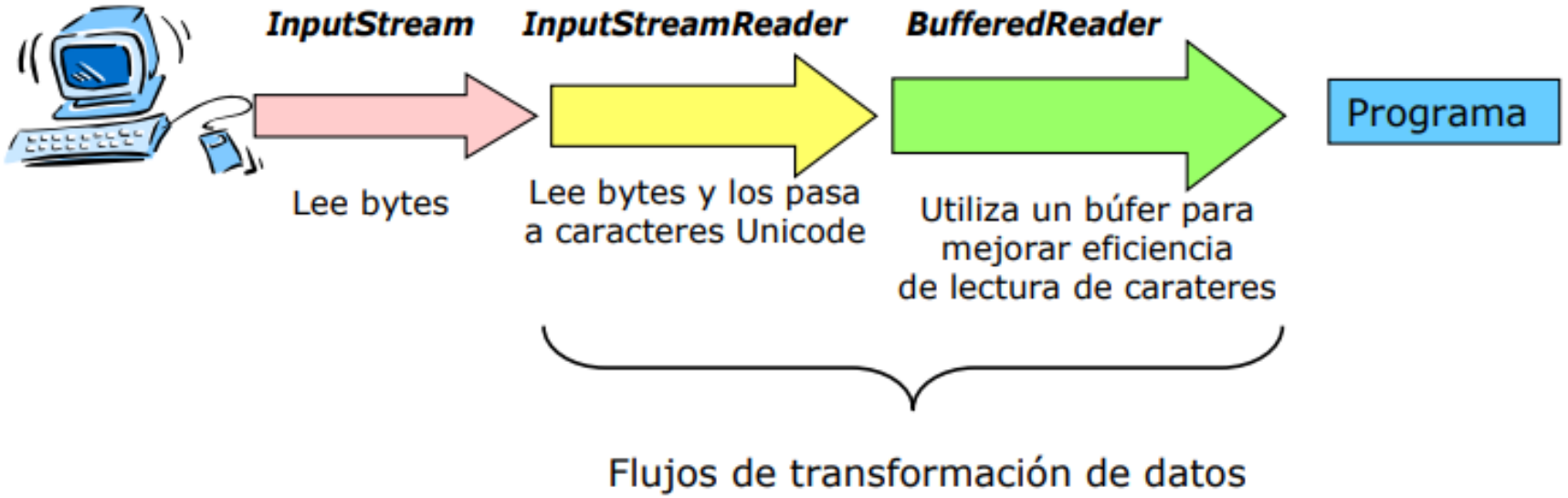


Jerarquía de flujos de caracteres



Combinación de flujos

- Los flujos se pueden combinar para obtener la funcionalidad deseada



Resumen

- La E/S en Java sigue el mismo modelo que en Unix:
  - Abrir, usar, cerrar flujo
  - Flujos estándar: ***System.in, System.out*** y ***System.err***
- Dos tipos de clases de E/S:
  - Readers y Writers para texto
    - Basados en el tipo char
  - Streams (InputStream y OutputStream) para datos binarios
    - Basados en el tipo byte
- Los flujos de E/S se pueden combinar para facilitar su uso
- La E/S suele ser propensa a errores
  - Implica interacción con el entorno exterior
  - Excepción ***IOException***