

---

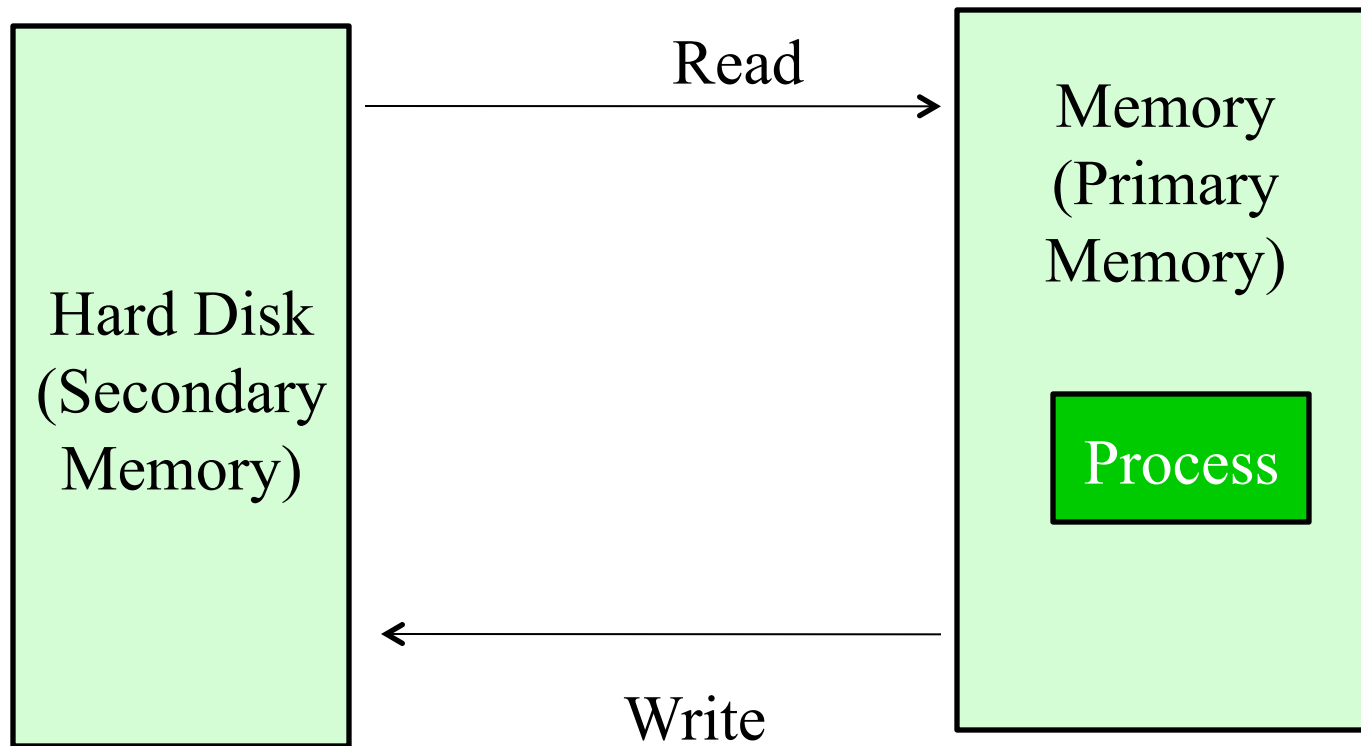
# File I/O, Streams, Scanner

IST 256

Application Programming for Information Systems

# File I/O

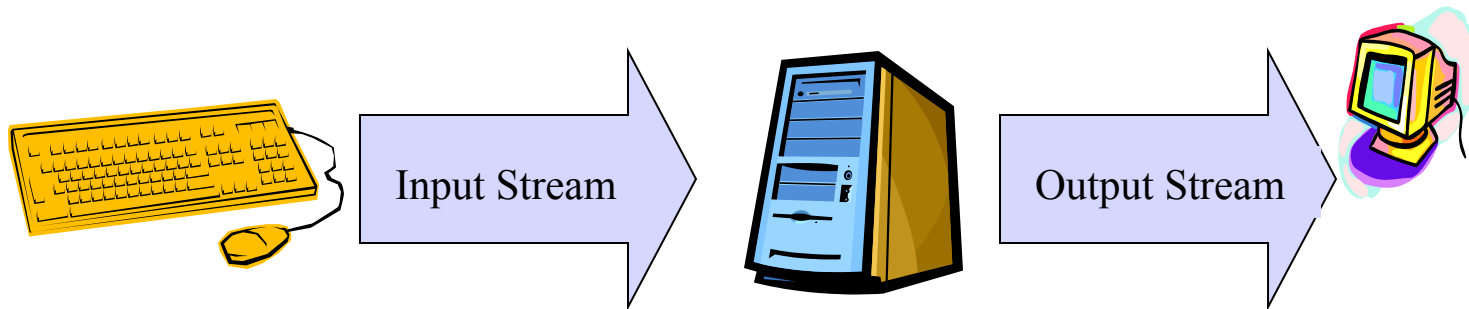
---



# I/O Streams

---

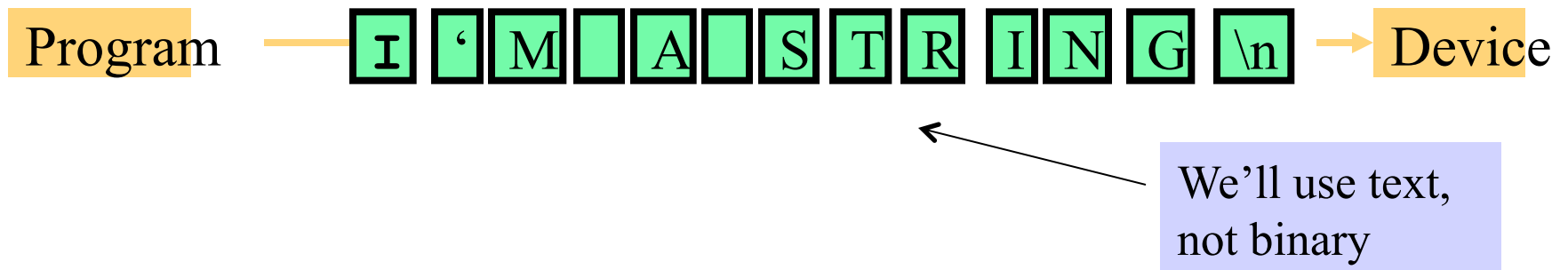
Default input/output streams



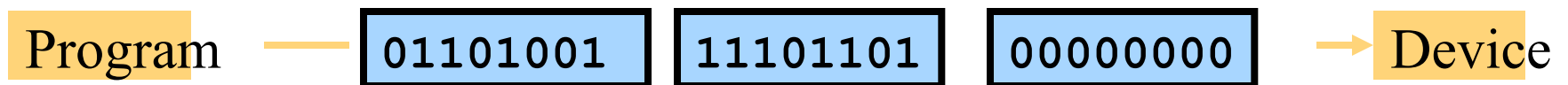
# Java - Streams

JAVA provides 2 types of streams

Text streams - containing 'characters'



Binary Streams - containing 8 – bit information



# Reading and Writing Files

---

Create a stream object and associate it with a disk-file



Give the stream object the desired functionality



while there is more information  
read(write) next data from(to) the stream



close the stream

# Reading Files

---

Code excerpt showing an Input Stream created as a `BufferedReader` and reading **one line of text** at a time:

```
BufferedReader instream
    = new BufferedReader(new FileReader("test.txt"));
while(instream.ready())
{
    String s = instream.readLine();
    System.out.println(s);
}
instream.close();
```

```
import java.io.*;
```

```
Place the code inside
try {...}
catch(IOException e){....}
wherever necessary
```

# Writing Files

---

Code excerpt showing an Output Stream created as a  
BufferedWriter, writing a string

```
import java.io.*;
```

```
String message = "example text";  
BufferedWriter outstream  
    = new BufferedWriter(new FileWriter("test_out.txt"));  
outstream.write(message);  
outstream.close();
```

```
Place the code inside  
try {...}  
catch(IOException e){....}  
wherever necessary
```

# Reading Tokens from a File

---

Code excerpt showing a Scanner wrapping an Input Stream, reading the **next String token** from the input (separated by white space)

```
BufferedReader instream
    = new BufferedReader(new FileReader("test.txt"));
Scanner scanner = new Scanner(instream);
while(scanner.hasNext())
{
    String message = scanner.next();
    System.out.println(message);
}
scanner.close();
```

```
import java.io.*;
import java.util.Scanner;
```

```
Place the code inside
try {...}
catch(IOException e){....}
wherever necessary
```