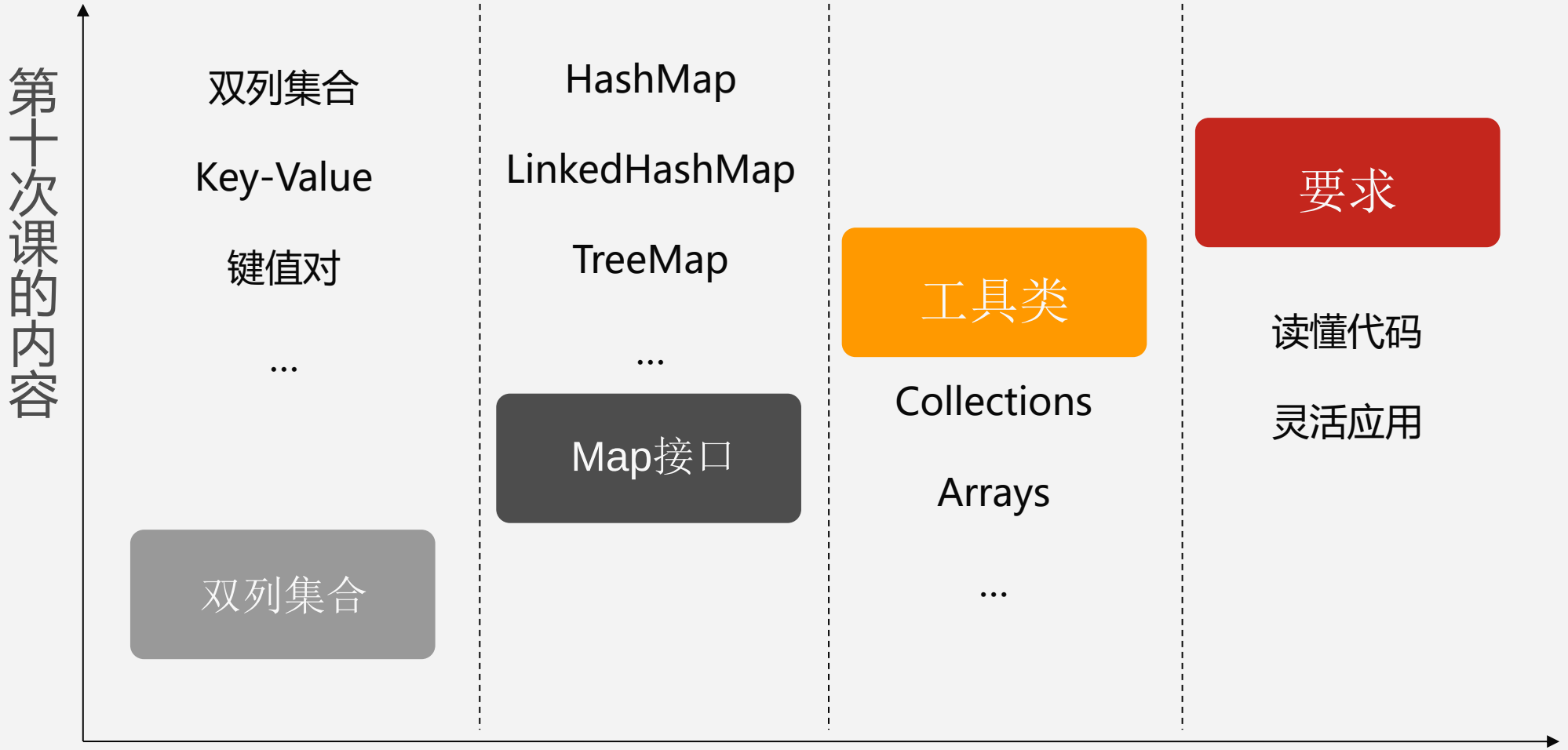
Java

面向对象程序设计

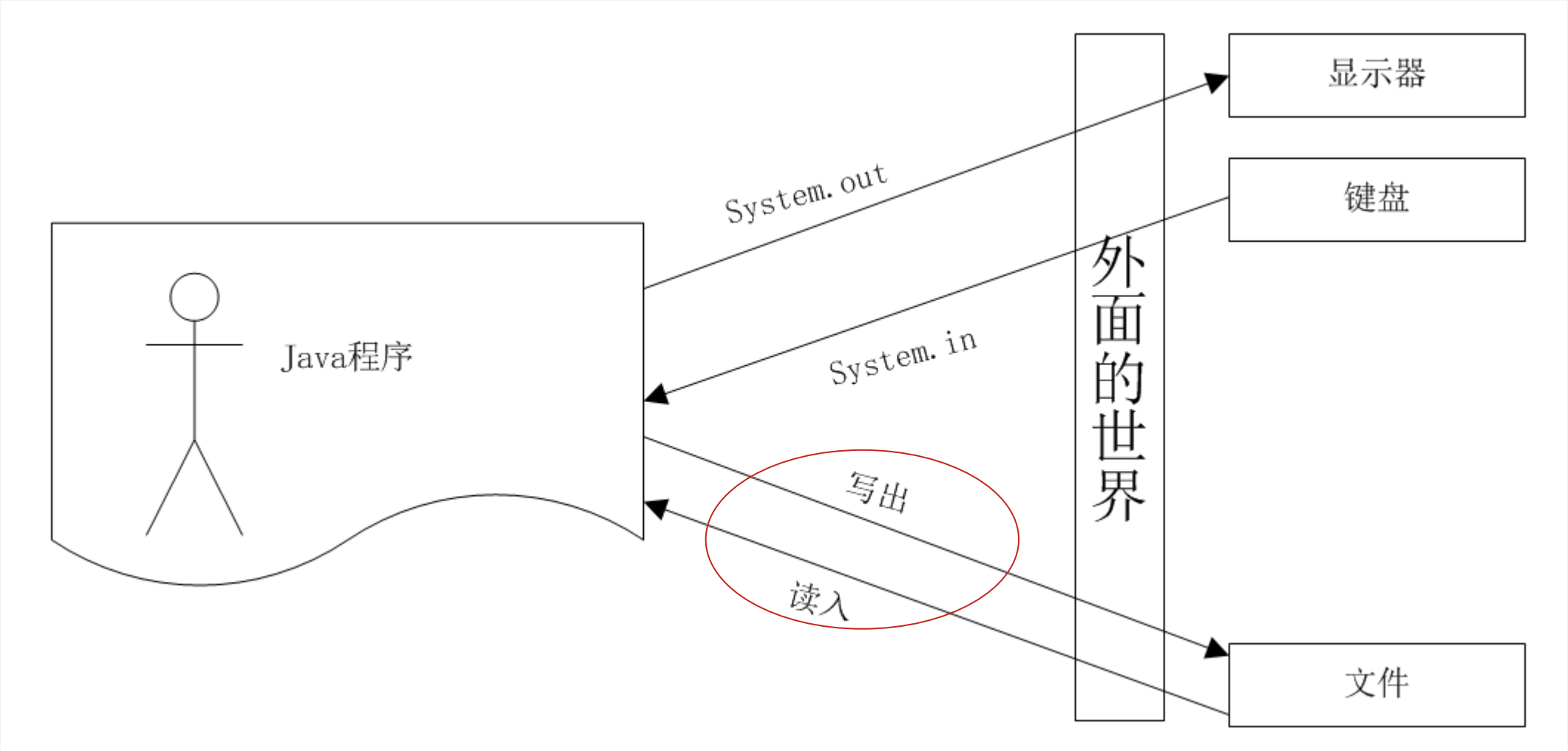
软件学院 贾伟峰

复习

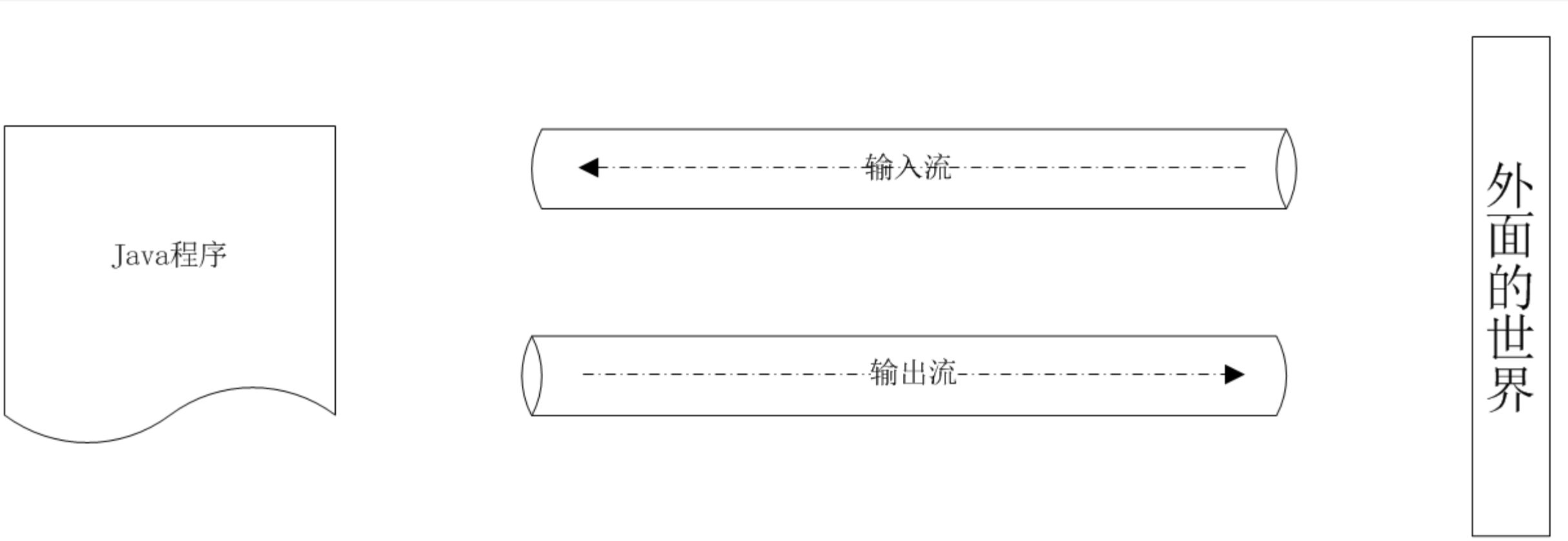

第十次课的内容
双列集合
Key-Value
键值对
...
双列集合
HashMap
LinkedHashMap
TreeMap
...
Map接口
工具类
Collections
Arrays
...
要求
读懂代码
灵活应用

Java中的IO(Input/Output，输入/输出)

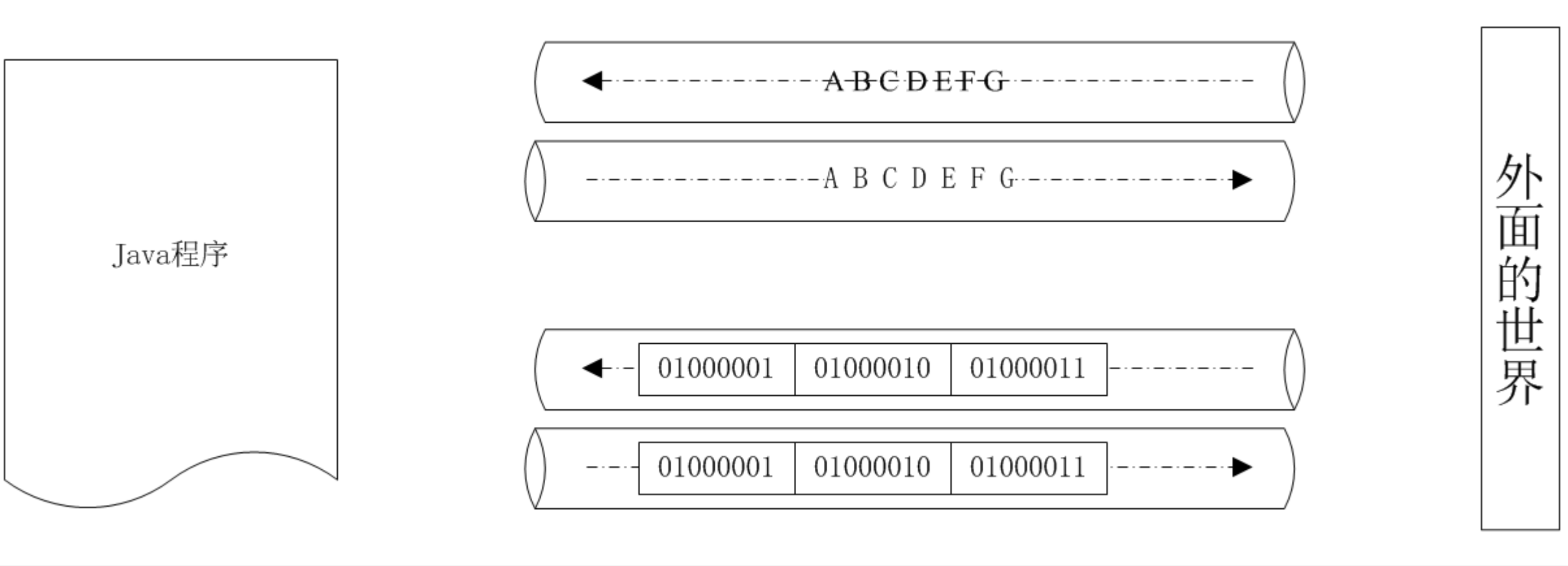
站在程序的世界向外看……



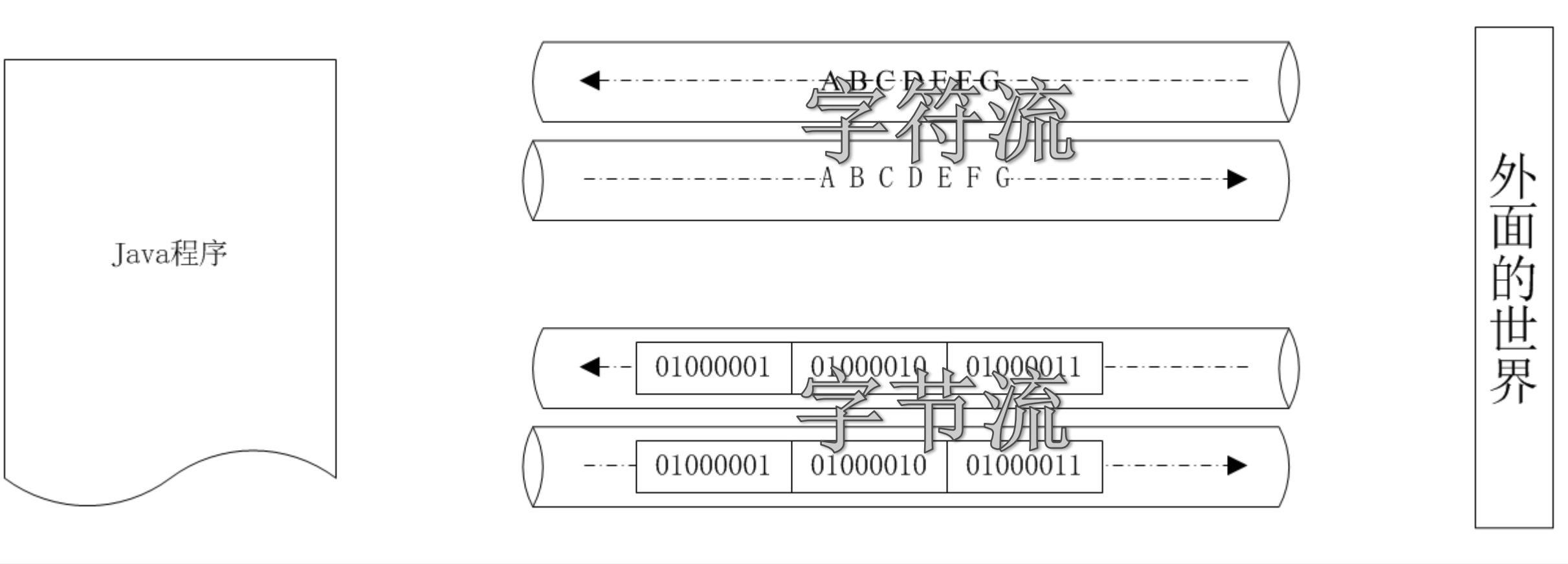
Java对程序和外部世界进行信息传输的抽象：流（Stream）



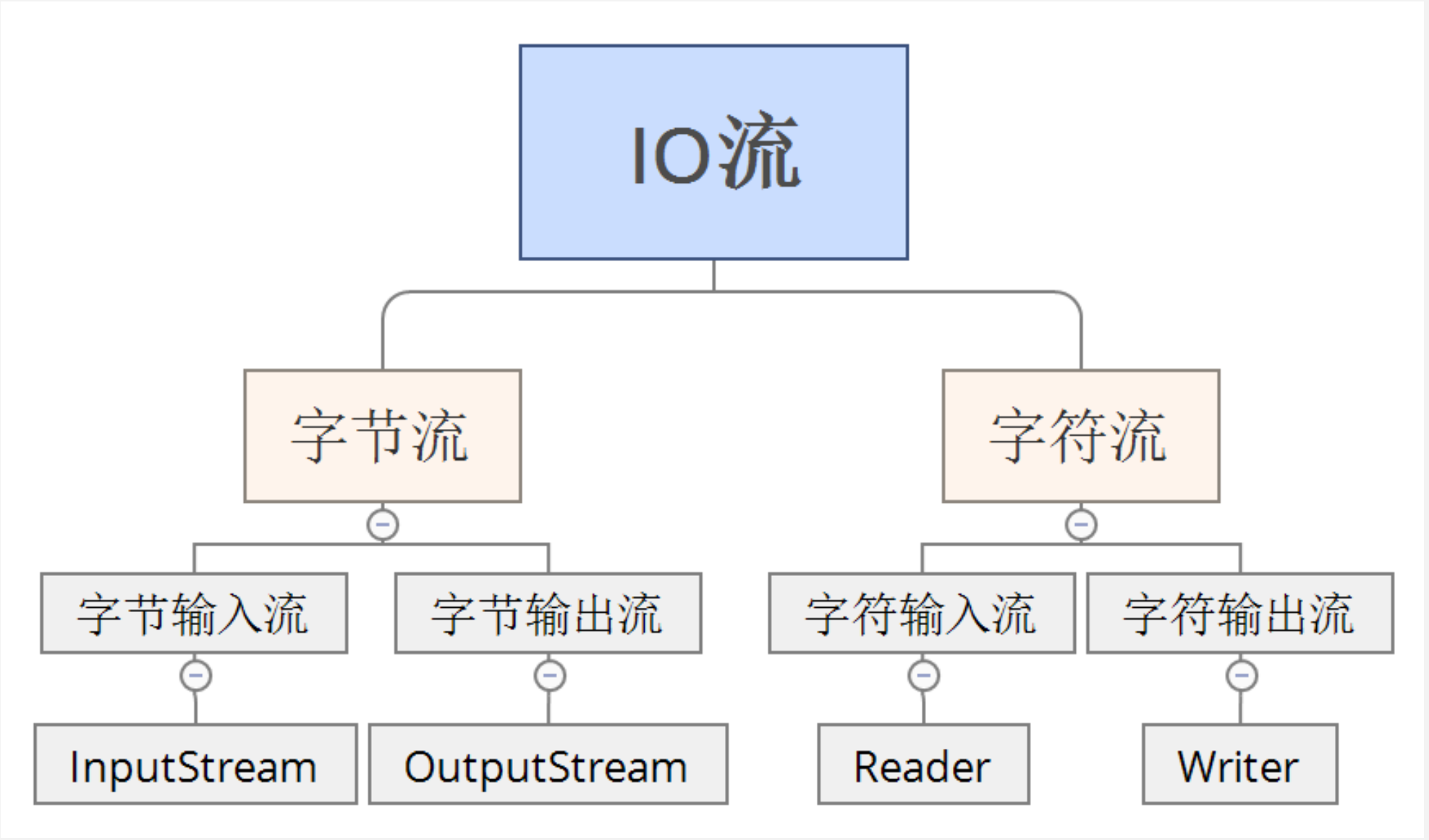
流（Stream）里面，跑的是什么数据呢？



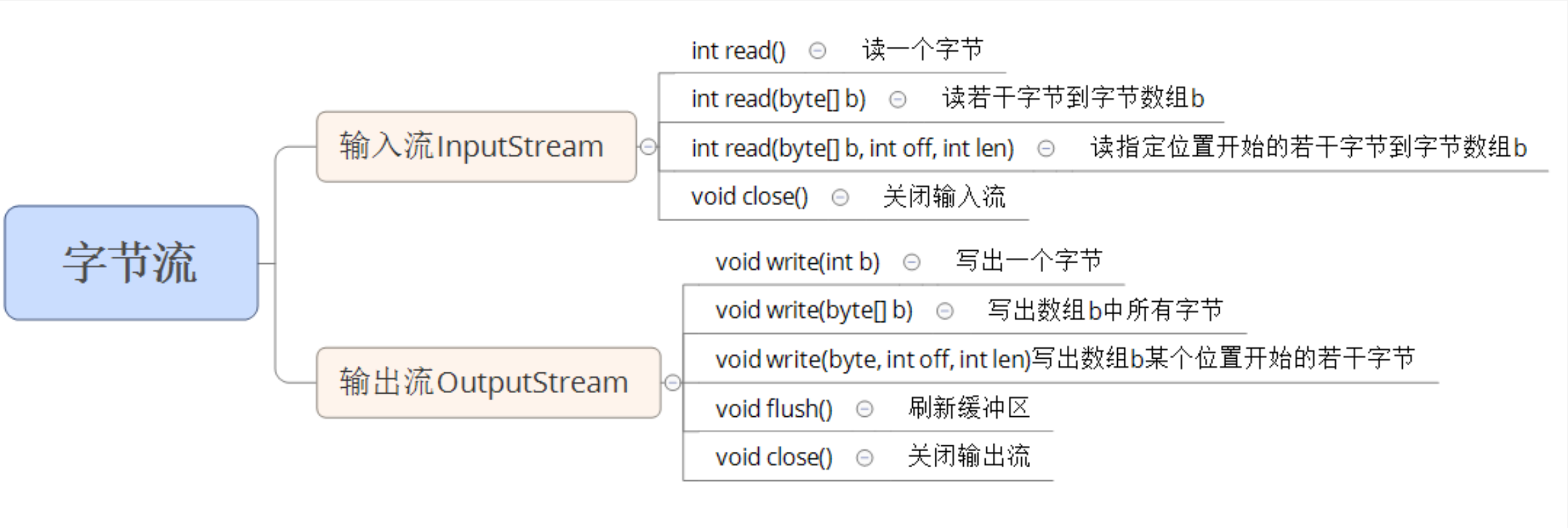
流（Stream）里面，跑的是什么数据呢？



Java 中的IO流分类



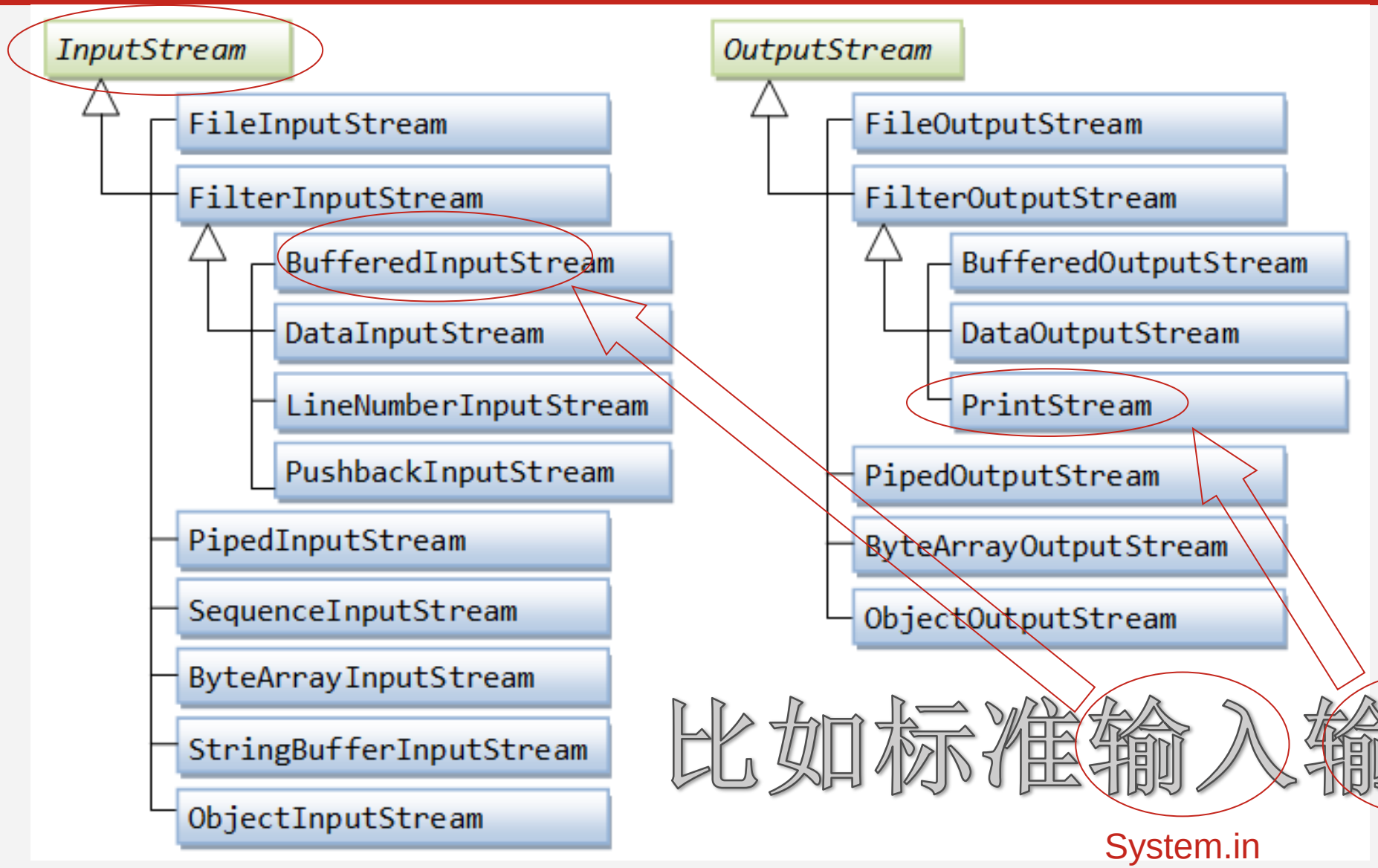
字节流InputStream和OutputStream



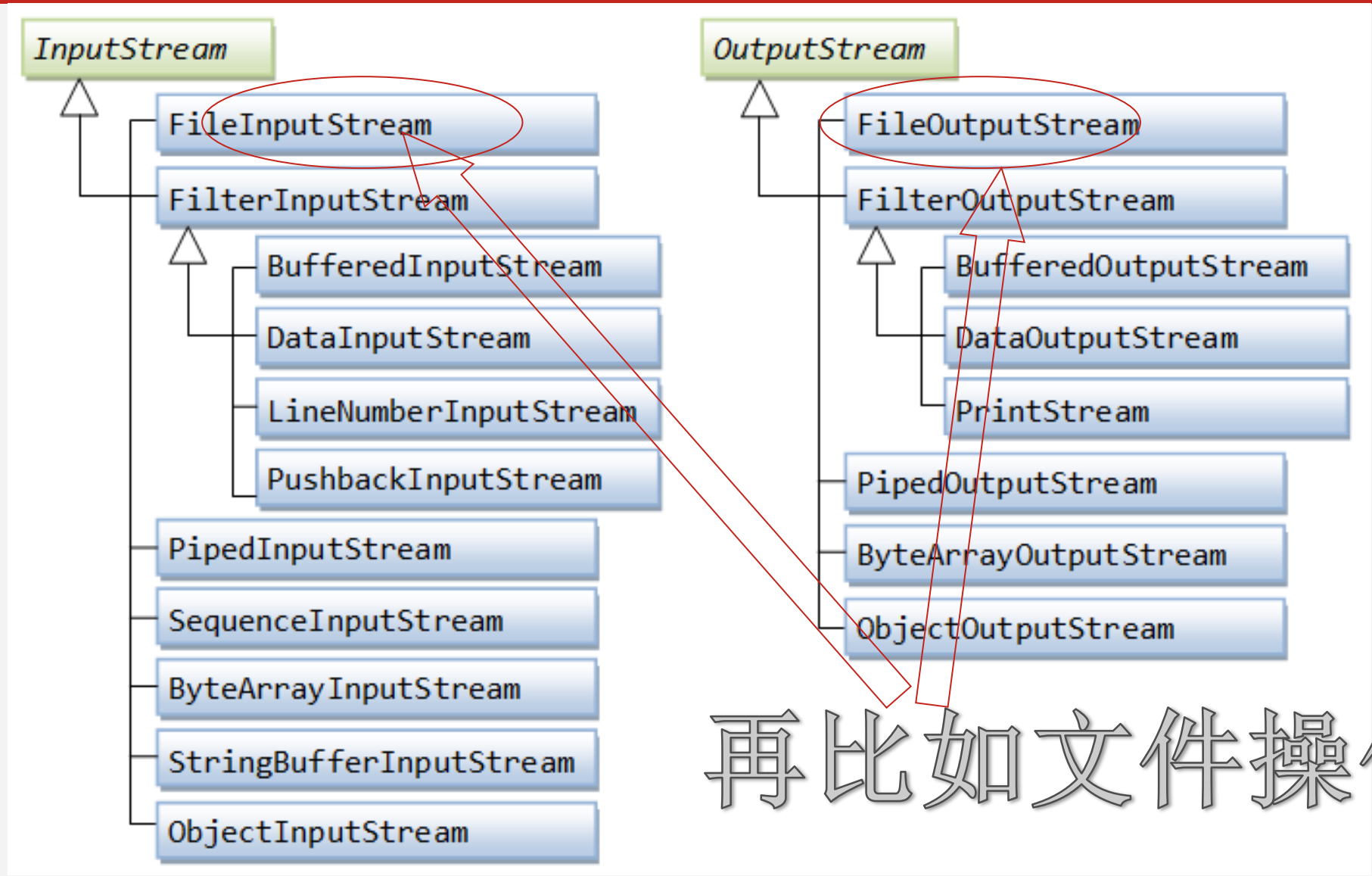
能否new InputStream()/OutputStream()?

抽象类InputStream/OutputStream无法实例化!

可以实例化的是他们的子类



可以实例化的是他们的子类（P283）



项目改进：将银行存取款系统中，用户账户的读取，从文件中读出，而非直接手工构造

```
11      private DBUtil() {
12          // -------刘备---------------------
13          User u1 = new User();
14          u1.setCardId("1001");
15          u1.setCardPwd("123456");
16          u1.setUserName("刘备");
17          u1.setCall("13281525712");
18          u1.setAccount(1000);
19          users.put(u1.getCardId(), u1);
20          // ------关羽---------------------
21          User u2 = new User();
22          u2.setCardId("1002");
23          u2.setCardPwd("234567");
24          u2.setUserName("关羽");
25          u2.setCall("13181527253");
26          u2.setAccount(1000);
27          users.put(u2.getCardId(), u2);
```

这些信息目前是手工编制，能否改为从文件读取？

```
1001, 123456, 刘备, 13281525712, 1000
1002, 234567, 关羽, 13181527253, 1000
```

文件格式如上

构思：一个字节一个字节的读取，遇到回车换行（\r\n）开始转为字符串进行处理

```
19              fs = new FileInputStream("data\\user.dat");
20              byte[] content = new byte[1024];
21              int i=0;
22              int conInteger = 0;
23              while(true){
24                  try {
25                      conInteger = fs.read();
26                  } catch (IOException e) {
27                      // TODO Auto-generated catch block
28                      e.printStackTrace();
29                  }
30                  if(-1 == conInteger){
31                      break;
32                  }else if('\r' == (char)conInteger || '\n' == (char)conInteger){
33                      try {
34                          System.out.println(new String(content, "GBK"));
35                          i=0;
36                      } catch (UnsupportedEncodingException e) {
37                          // TODO Auto-generated catch block
38                          e.printStackTrace();
39                      }
40                      continue;
41                  }else{
42                      content[i] = (byte)conInteger;
43                      i++;
44                  }
45              }
```

此处仅是打印读出的行，如果要引入到银行存取款模拟系统中该如何做？1024个字节后面没用到的怎么办？还记得trim()么？

项目改进：增加从文件读取的功能。

```
private DBUtil() {
    getUsersFromInputStream("user.dat");
}
```

user.dat放在根目录下

此处指eclipse中该project的根目录。

```
49      //读取文件，以InputStream的形式读取
50      private void getUsersFromInputStream(String isName) {
51          try {
52              fs = new FileInputStream(isName);
53              byte[] content = new byte[1024];
54              int i=0;
55              int conInteger = 0;
56              while(true){
57                  try {
58                      conInteger = fs.read();
59                  } catch (IOException e) {
60                      // TODO Auto-generated catch block
61                      e.printStackTrace();
62                  }
63                  if(-1 == conInteger){
64                      break;
65                  }else if('\r' == (char)conInteger || '\n' == (char)conInteger){
66                      try {
67                          this.processUserString(new String(content, "GBK").trim());
68                          i=0;
69                      } catch (UnsupportedEncodingException e) {
```

```
39      public void processUserString(String userString) {
40          String [] userFields = userString.split(",");
41          User u = new User();
42          u.setCardId(userFields[0]);
43          u.setCardPwd(userFields[1]);
44          u.setUserName(userFields[2]);
45          u.setCall(userFields[3]);
46          u.setAccount(Integer.parseInt(userFields[4]));
47          users.put(u.getCardId(), u);
48      }
```

继续改进：为系统提供注册账户的功能，且能够进行存盘操作。

注册功能

Bank.java

```
69      private void register() {
70          User u = new User();
71          Scanner scanner = new Scanner(System.in);
72          System.out.println("输入卡号：");
73          u.setCardId(scanner.nextLine());
74          System.out.println("输入用户名：");
75          u.setUserName(scanner.nextLine());
76          System.out.println("输入密码：");
77          u.setCardPwd(scanner.nextLine());
78          System.out.println("输入手机号：");
79          u.setCall(scanner.nextLine());
80          System.out.println("输入余额：");
81          u.setAccount(scanner.nextInt());
82          DBUtil dbUtil = DBUtil.getInstance();
83          dbUtil.addUser(u);
84      }
```

DBUtil.java

```
42      //增加一个用户
43      public void addUser(User u) {
44          users.put(u.getCardId(), u);
45      }
46
```

存盘功能

Bank.java

```
63    private void save() {
64        // TODO Auto-generated method stub
65        DBUtil dbUtil = DBUtil.getInstance();
66        dbUtil.update();
67    }
```

DBUtil.java

```
47    //存盘操作
48    public void update() {
49        Set<String> userSet = users.keySet();
50        StringBuffer uStringBuffer = new StringBuffer();
51        for(String cardId:userSet) {
52            User u = (User)users.get(cardId);
              String uString = u.getCardId() + ","
                      + u.getCardPwd() + ","
                      + u.getUserName() + ","
                      + u.getCall() + ","
                      + u.getAccount() + "\r\n";
58            uStringBuffer.append(uString);
59        }
60        putUsersToFile(uStringBuffer.toString(), "user.dat");
61    }
62    //写入文件的函数
63    private void putUsersToFile(String uString, String osName) {
64        try {
65            fos = new FileOutputStream(osName);
66            try {
67                fos.write(uString.getBytes("GBK"));
68            } catch (UnsupportedEncodingException e) {
69                // TODO Auto-generated catch block
70                e.printStackTrace();
```

复制一个文件

InputStream和OutputStream联合使用. p287

讨论：复制的效率问题。

带缓冲功能的字节流：BufferedInputStream和BufferedOutputStream

字符流的使用：读取文件

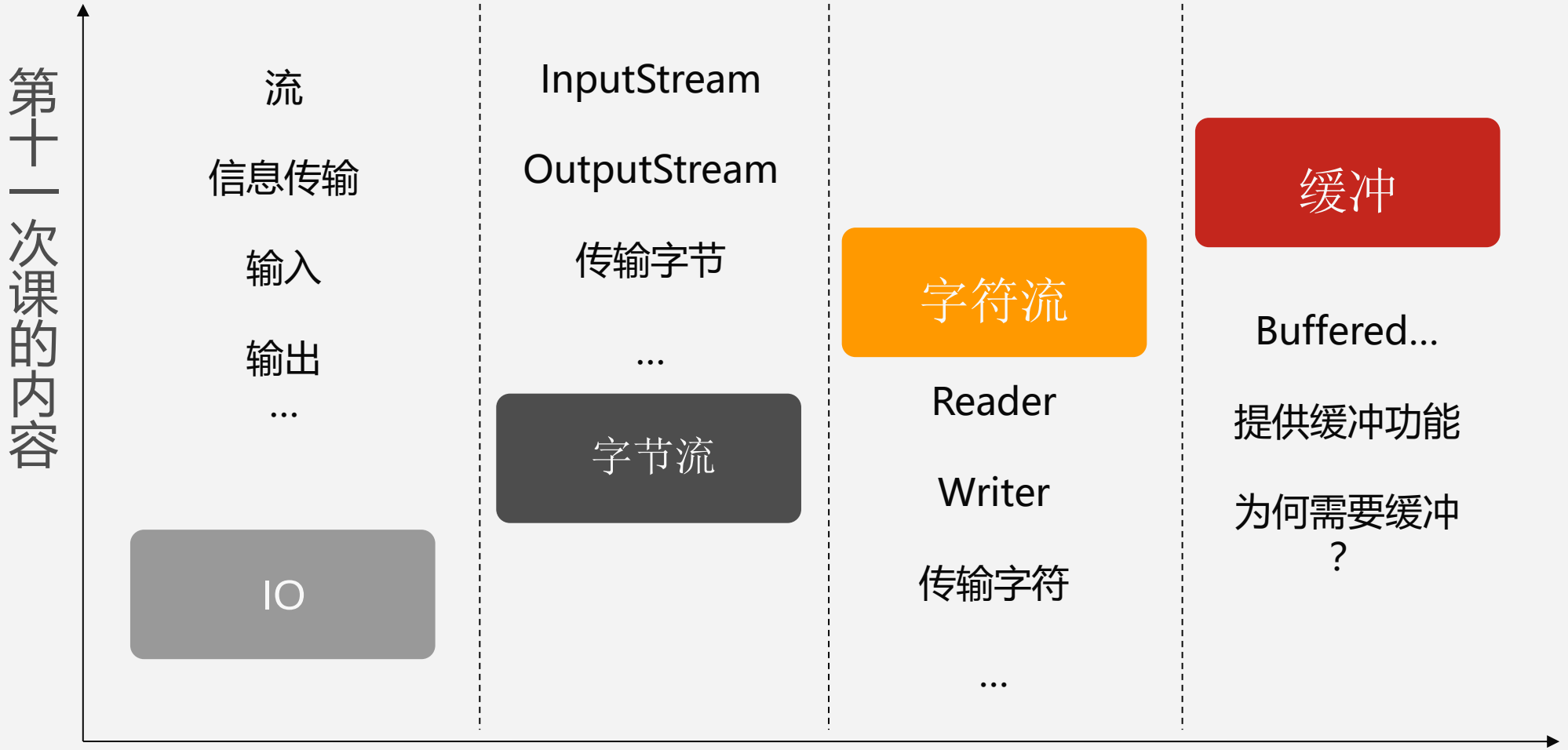
```
26      //字符流方式读取文件
27      private void getUsersFromReader(String fileName) {
28          try {
29              reader = new FileReader(fileName);
30          } catch (FileNotFoundException e) {
31              // TODO Auto-generated catch block
32              e.printStackTrace();
33          }
34          BufferedReader br = new BufferedReader(reader);
35          String userString;
36          try {
37              while((userString=br.readLine()) != null) {
38                  processUserString(userString);
39              }
40          } catch (IOException e) {
41              // TODO Auto-generated catch block
42              e.printStackTrace();
43          }
44      }
```

字符流的使用：写入文件

```
88      //存盘操作——字符流
89      public void update2() {
90          Set<String> userSet = users.keySet();
91          try {
92              writer = new FileWriter("user.dat");
93          } catch (IOException e1) {
94              // TODO Auto-generated catch block
95              e1.printStackTrace();
96          }
97          BufferedWriter bfw = new BufferedWriter(writer);
98          for (String cardId : userSet) {
99              User u = (User) users.get(cardId);
100             String uString = u.getCardId() + "," + u.getCardPwd() + "," + u.getUserName()
101                     + u.getAccount() + "\r\n";
102             try {
103                 bfw.write(uString);
104             } catch (IOException e) {
105                 // TODO Auto-generated catch block
106                 e.printStackTrace();
107             }
108         }
109         if (bfw != null)
110             try {
```

下面是关闭输出流的操作

总结


第十一次课的内容
流
信息传输
输入
输出
...
IO
InputStream
OutputStream
传输字节
...
字节流
字符流
Reader
Writer
传输字符
...
缓冲
Buffered...
提供缓冲功能
为何需要缓冲？