Ввод и вывод информации - это то, с чем сталкивается программист на каждом шагу. Это относится как к написанию коммерческих продуктов, так и к решению олимпиадных задач. Сложилось так, что новичку тяжело разобраться в том, как эти действия выполняютя средствами языка Java. В данном тексте я попробую облегчить эту проблему, бегло рассмотрев возможности пакета [java.io](http://java.sun.com/javase/6/docs/api/java/io/package-summary.html).

В центре всего, что связано с вводом/выводом информации, лежит понятие потока (*stream*). Поток - это просто источник информации или же место, куда информацию можно направить. Нас интересует случай работы с файлами.

Простейший вид потока - тот, который работает с двоичным представением информации, иначе, с байтами. Все бинарные потоки наследуются от двух абстрактных классов - [InputStream](http://java.sun.com/javase/6/docs/api/java/io/InputStream.html) и [OutputStream](http://java.sun.com/javase/6/docs/api/java/io/OutputStream.html). Нам важны два потомка - [FileInputStream](http://java.sun.com/javase/6/docs/api/java/io/FileInputStream.html) и [FileOutputStream](http://java.sun.com/javase/6/docs/api/java/io/FileOutputStream.html) соответственно. Вот кусок текста программы, которая копирует содержимое файла *a.txt* в файл *b.txt*:

FileInputStream in = null;   
FileOutputStream out = null;   
try {   
 in = new FileInputStream("a.txt");   
 out = new FileOutputStream("b.txt");   
 int c;   
 while ((c = in.read()) != -1) { out.write(c); }   
} finally {   
 if (in != null) { in.close(); }   
 if (out != null) { out.close(); }   
}

Не забывайте закрывать потоки методом **close()**! Ваши данные могут быть утеряны. Оба класса обладают удобными конструкторами от пути к файлу. Если *b.txt* не существует, то будет создан. Иначе - переписан. Не забывайте отлавливать исключения (или же указывать их в заголовках методов).

На практике работать с двоичным представлением данных при написании олимпиадных задач не приходится. Небольшим шажком вперед послужат [FileReader](http://java.sun.com/javase/6/docs/api/java/io/FileReader.html) и [FileWriter](http://java.sun.com/javase/6/docs/api/java/io/FileWriter.html), которые очень похожи на бинарных братьев. Для нас отличие только в том, что они работают с символами. Новую возможность предоставляют [BufferedReader](http://java.sun.com/javase/6/docs/api/java/io/BufferedReader.html) и [PrintWriter](http://java.sun.com/javase/6/docs/api/java/io/PrintWriter.html), которые можно рассматривать, как обертки для уже предыдущей пары классов. Эта возможность - построчное чтение и запись файла. Приведем все тот же пример копирования файла:

BufferedReader in = null;   
PrintWriter out = null;   
try {   
 in = new BufferedReader(new FileReader("a.txt"))