Figure 5.1 Program input via *System.in*.

Using **System.out** was easy. **System.in** is a bit more complicated:

- first we must declare an object of class **BufferedReader**
  - this object will "pre-process" the standard input for us
  - it can be declared inside **main()**, as with

    BufferedReader keyb;
  - or it can be declared outside **main()**, as an instance variable

    private static BufferedReader keyb; // *we'll explain* **static** *soon*
- next we instantiate the actual **BufferedReader** object; this we do inside **main()** with

  keyb = new BufferedReader(new InputStreamReader(System.in));
  - **System.in** belongs to class **InputStream**
  - this instantiation creates *two* new objects—we use an **InputStreamReader** object to create a **BufferedReader** object (see sections 3.5 and 3.6 of the text!)
- input can fail, so we need a way to warn the compiler

  public static void main (String[] args) <u>throws IOException</u> { ... }
- not every program does input, so we need to ask for the input class library

  import java.io.*; // *goes before class definition*

Here's a complete program using standard I/O:

```
/**
 * EchoInput: reads one line from the keyboard and prints it to the screen.
 */

import java.io.*;

public class EchoInput {
    public static void main (String[] args) throws IOException {
        BufferedReader keyb = new BufferedReader(new InputStreamReader(System.in));
        String inputLine = keyb.readLine();
        System.out.println(inputLine);
    }
}
```

readLine of class **BufferedReader** returns (a reference to) a **String**