Basic IO: Scanner and printf

Reading Input from the Cons

1. Create a Scanner object

```
Scanner input = new Scanner( System.in );
```

2. Use the method nextDouble() to obtain to a double value. For example,

```
System.out.print( "Enter a double value: " );
Scanner input = new Scanner( System.in );
double d = input.nextDouble();
```

ComputeAreaWithConsoleInput.java

```java
import java.util.Scanner; // Scanner is in the java.util package

public class ComputeAreaWithConsoleInput {
  public static void main (String[] args) {
    // Create a Scanner object
    Scanner input = new Scanner( System.in );

    // Prompt the user to enter a radius
    System.out.print( "Enter a number for radius: " );
    double radius = input.nextDouble();

    // Compute area
    double area = radius * radius * Math.PI;

    // Display result
    System.out.println( "The area for the circle of radius " +
      radius + " is " + area
    );
  }
}
```

```
Enter a number for radius:  2.5  Enter
The area for the circle of radius 2.5 is 19.6349375
```

```
Enter a number for radius:  23  Enter
The area for the circle of radius 23.0 is 1661.90111
```

ComputeAverage.java

```java
import java.util.Scanner; // Scanner is in the java.util package

public class ComputeAverage {
  public static void main (String[] args) {
    // Create a Scanner object
    Scanner input = new Scanner( System.in );

    // Prompt the user to enter three numbers
    System.out.print( "Enter three numbers: " );
    double number1 = input.nextDouble();
    double number2 = input.nextDouble();
    double number3 = input.nextDouble();

    // Compute average
    double average = (number1 + number2 + number3) / 3;

    // Display result
    System.out.println( "The average of " + number1 + " " + number2
      + " " + number3 + " is " + average
    );
  }
}
```

```
Enter three numbers: 1 2 3 Enter
The average of 1.0 2.0 3.0 is 2.0
```

enter input in one line

```
Enter three numbers: 10.5 Enter
11 Enter
11.5 Enter
The average of 10.5 11.0 11.5 is 11.0
```

enter input in multiple lines

```
Scanner(InputStream source)
Scanner(File source)
Scanner(String source)
```
Constructors: sets up the new scanner to scan values from the specified source.

```
String next()
```
Returns the next input token as a character string.

```
String nextLine()
```
Returns all input remaining on the current line as a character string.

```
boolean nextBoolean()
byte nextByte()
double nextDouble()
float nextFloat()
int nextInt()
long nextLong()
short nextShort()
```
Returns the next input token as the indicated type. Throws `InputMismatchException` if the next token is inconsistent with the type.

```
boolean hasNext()
```
Returns true if the scanner has another token in its input.

```
Scanner useDelimiter(String pattern)
Scanner useDelimiter(Pattern pattern)
```
Sets the scanner's delimiting pattern.

```
Pattern delimiter()
```
Returns the pattern the scanner is currently using to match delimiters.

```
String findInLine(String pattern)
String findInLine(Pattern pattern)
```
Attempts to find the next occurrence of the specified pattern, ignoring delimiters.

Some methods of the Scanner class

Formatting Output

Use the `printf` method (similar like printf in C/C++).

```
System.out.printf( format, items );
```

Where format is a string that may consist of substrings and format specifiers (格式说明符，格式规格符).

A format specifier specifies how an item should be displayed.

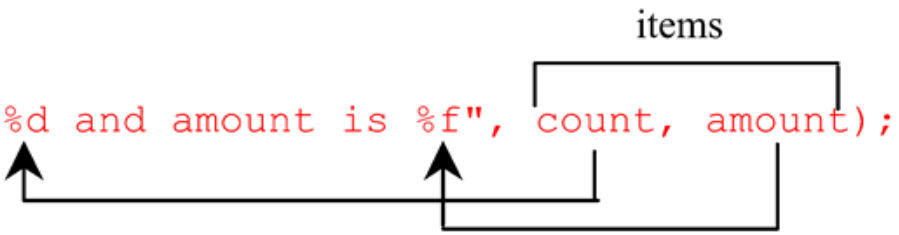
An item may be a numeric value, character, boolean value, or a string.

Each specifier begins with a percent sign.

Frequently-Used Specifiers

Specifier	Output	Example
`%b`	a boolean value	`true or false`
`%c`	a character	`'a'`
`%d`	a decimal integer	`200`
`%f`	a floating-point number	`45.460000`
`%e`	a number in standard scientific notation	`4.556000e+01`
`%s`	a string	`"Java is cool"`

```
int count = 5;
double amount = 45.56;
System.out.printf("count is %d and amount is %f", count, amount);
```




```
display              count is 5 and amount is 45.560000
```

System.out.printf(…)

System.out.printf (…)

Format specifiers: `%.2f, %10.2f`

`%f` is used to output values of type float or double.

`.2` represents the number of decimal places (2) to output to the right of the decimal point—known as the number's precision.

Any floating-point value output with `%.2f` will be rounded to the hundredths position.

`10` in `%10.2f` represents the total width of the real number occupied

To be continued …