a)

```
1  public class ParameterExample {
2      public static void main(String[] args) {
3          int spaces1 = 3;
4          int spaces2 = 5;
5
6          System.out.print("*");
7          writeSpaces(spaces1);
8          System.out.println("*");
9
10         System.out.print("!");
11         writeSpaces(spaces2);
12         System.out.println("!");
13
14         System.out.print("'");
15         writeSpaces(8);
16         System.out.println("'");
17
18         System.out.print("<");
19         writeSpaces(spaces1 * spaces2 - 5);
20         System.out.println(">");
21     }
22
23     // writes "number" spaces on the current output line
24     public static void writeSpaces(int number) {
25         for (int i = 1; i <= number; i++) {
26             System.out.print(" ");
27         }
28     }
29 }
```

b)

```
1  public class ParameterExample2 {
2      public static void main(String[] args) {
3          int x = 17;
4          doubleNumber(x);
5          System.out.println("x = " + x);
6          System.out.println();
7
8          int number = 42;
9          doubleNumber(number);
10         System.out.println("number = " + number);
11     }
12
13     public static void doubleNumber(int number) {
14         System.out.println("Initial value = " + number);
15         number *= 2;
16         System.out.println("Final value = " + number);
17     }
18 }
```

c)

Math Constants

Constant	Description
`E`	base used in natural logarithms (2.71828. . .)
`PI`	ratio of circumference of a circle to its diameter (3.14159 . . .)

Useful Static Methods in the Math Class

Method	Description	Example
`abs`	absolute value	`Math.abs(-308)` returns `308`
`ceil`	ceiling (rounds upward)	`Math.ceil(2.13)` returns `3.0`
`cos`	cosine (radians)	`Math.cos(Math.PI)` returns `-1.0`
`exp`	exponent base *e*	`Math.exp(1)` returns `2.7182818284590455`
`floor`	floor (rounds downward)	`Math.floor(2.93)` returns `2.0`
`log`	logarithm base *e*	`Math.log(Math.E)` returns `1.0`
`log10`	logarithm base 10	`Math.log10(1000)` returns `3.0`
`max`	maximum of two values	`Math.max(45, 207)` returns `207`
`min`	minimum of two values	`Math.min(3.8, 2.75)` returns `2.75`
`pow`	power (general exponentiation)	`Math.pow(3, 4)` returns `81.0`
`random`	random value	`Math.random()` returns a random `double` value k such that $0.0 \leqslant k < 1.0$
`round`	round real number to nearest integer	`Math.round(2.718)` returns `3`
`sin`	sine (radians)	`Math.sin(0)` returns `0.0`
`sqrt`	square root	`Math.sqrt(2)` returns `1.4142135623730951`
`toDegrees`	converts from radians to degrees	`Math.toDegrees(Math.PI)` returns `180.0`
`toRadians`	converts from degrees to radians	`Math.toRadians(270.0)` returns `4.71238898038469`

D)

```
1  public class BusSong {
2      public static void main(String[] args) {
3          verse("wheels", "go", "round and round");
4          verse("wipers", "go", "swish, swish, swish");
5          verse("horn", "goes", "beep, beep, beep");
6      }
7
8      public static void verse(String item, String verb, String sound) {
9          System.out.println("The " + item + " on the bus " +
10                             verb + " " + sound + ",");

11           System.out.println(sound + ",");
12           System.out.println(sound + ".");
13           System.out.println("The " + item + " on the bus " +
14                              verb + " " + sound + ",");
15           System.out.println("All through the town.");
16           System.out.println();
17       }
18   }
```

It produces the following output:

```
The wheels on the bus go round and round,
round and round,
round and round.
The wheels on the bus go round and round,
All through the town.

The wipers on the bus go swish, swish, swish,
swish, swish, swish,
swish, swish, swish.
The wipers on the bus go swish, swish, swish,
All through the town.

The horn on the bus goes beep, beep, beep,
beep, beep, beep,
beep, beep, beep.
The horn on the bus goes beep, beep, beep,
All through the town.
```

E)

```
Exception in thread "main"
        java.lang.StringIndexOutOfBoundsException:
        String index out of range: 13
        at java.lang.String.substring(Unknown Source)
        at ExampleProgram.main(ExampleProgram.java:3)
```

F)

Table 3.3 Useful Methods of `String` Objects

Method	Description	Example (assuming `s` is `"hello"`)
`charAt(index)`	character at a specific index	`s.charAt(1)` returns `'e'`
`endsWith(text)`	whether or not the string ends with some text	`s.endsWith("llo")` returns `true`
`indexOf(text)`	index of a particular character or `String` (–1 if not present)	`s.indexOf("o")` returns `4`
`length()`	number of characters in the string	`s.length()` returns `5`
`startsWith(text)`	whether or not the string starts with some text	`s.startsWith("hi")` returns `false`
`substring(start, stop)`	characters from start index to just before stop index	`s.substring(1, 3)` returns `"el"`
`toLowerCase()`	a new string with all lowercase letters	`s.toLowerCase()` returns `"hello"`
`toUpperCase()`	a new string with all uppercase letters	`s.toUpperCase()` returns `"HELLO"`

G)

```
Exception in thread "main" java.util.InputMismatchException
        at java.util.Scanner.throwFor(Unknown Source)
        at java.util.Scanner.next(Unknown Source)
        at java.util.Scanner.nextInt(Unknown Source)
        at Example.main(Example.java:13)
```

H)

```
1   // This program prompts for information about a loan and
2   // computes the monthly mortgage payment.
3
4   import java.util.*;   // for Scanner
5
6   public class Mortgage {
7       public static void main(String[] args) {
8           Scanner console = new Scanner(System.in);
9
10          // obtain values
11          System.out.println("This program computes monthly " +
12                             "mortgage payments.");
13          System.out.print("loan amount     : ");
14          double loan = console.nextDouble();
15          System.out.print("number of years : ");
16          int years = console.nextInt();
17          System.out.print("interest rate   : ");
18          double rate = console.nextDouble();
19          System.out.println();
20
21          // compute result and report
22          int n = 12 * years;
23          double c = rate / 12.0 / 100.0;

24          double payment = loan * c * Math.pow(1 + c, n) /
25                                   (Math.pow(1 + c, n) - 1);
26          System.out.println("payment = $" + (int) payment);
27      }
28  }
```

The following is a sample execution of the program (user input is in bold):

```
This program computes monthly mortgage payments.
loan amount     : 275000
number of years : 30
interest rate   : 6.75

payment = $1783
```

I)

However, because the payment is stored in a variable of type `double`, such a statement would print all the digits of the number. For example, for the log listed above, it would print the following:

```
payment = $1783.6447655625927
```

That is a rather strange-looking output for someone who is used to dollars and cents. For the purposes of this simple program, it's easy to cast the `double` to an `int` and report just the dollar amount of the payment:

```
System.out.println("payment = $" + (int) payment);
```