Lesson 6…..Methods of the *Math* Class

One of the most useful methods of the *Math* class is *sqrt( )* …which means square root. For example, if we want to take the square root of 17 and store the result in *p*, do the following:

```
double p = Math.sqrt(17);
```

Notice that we must store the result in a *double*…. *p* in this case. We must store in a *double* since square roots usually don't come out even.

Signature of a method:
Below we will give the description of some methods of the *Math* class… along with the signatures of the method. First, however, let's explain the meaning of **signature** (also called a **method declaration**). Consider the signature of the *sqrt( )* method:

```
double sqrt( double x )
  |      |        |
type returned  method name  type of parameter we send to the method
```

Method	Signature	Description
abs	int abs(int x)	Returns the absolute value of x
abs	double abs(double x)	Returns the absolute value of x
pow	double pow(double b, double e)	Returns b raised to the e power
sqrt	double sqrt(double x)	Returns the square root of x
ceil	double ceil(double x)	Returns next highest whole number from x
floor	double floor(double x)	Returns next lowest whole number from x
min	double min(double a, double b)	Returns the smaller of a and b
max	double max(double a, double b)	Returns the larger of a and b
min	int min(int a, int b)	Returns the smaller of a and b
max	int max(int a, int b)	Returns the larger of a and b
	(For both *min* and *max* there are also versions that both accept and return types *float*, *short*, and *long*. See Appendix C for more on these three data types.)	
random	double random( )	Returns a random double (range $0 \leq r < 1$)
round	long round(double x)	Returns x rounded to nearest whole number
PI	double PI	Returns 3.14159625…..

Now, we offer examples of each (most of these you can do on a calculator for verification):

1. double d = -379.22;
   System.out.println(Math.abs(d)); //**379.22**

2. double b = 42.01;
   double e = 3.728;
   System.out.println (Math.pow(b, e)); //**1126831.027**

3. double d = 2034.56;
   System.out.println(Math.sqrt(d)); //**45.10609715**

4. double d = 1.4;
   System.out.println(Math.ceil(d)); //**2.0**

5. double d = -1.6;
   System.out.println(Math.ceil(d)); //**-1.0**

6. double d = 1.4;
   System.out.println(Math.floor(d)); //**1.0**

7. double d = -1.6;
   System.out.println(Math.floor(d)); //**-2.0**

   The last four examples illustrating *floor* and *ceil*ing are best understood with the following drawing:

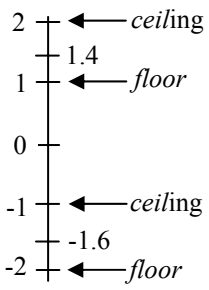

Figure 6-1 Relationship of *ceil*ing and *floor*

Just think of the *ceil*ing as it is in a house… on top. Likewise, think of the *floor* as being on the bottom.

Therefore, *Math.ceil(-1.6)* being *-1* makes perfect sense since *-1* is above. Similarly, *-2* is below *-1.6* so it makes sense to say that *-2* is *Math.floor(-1.6)*.

8. double d = 7.89;
   System.out.println(Math.log(d)); //**2.065596135** …log is base e.

9. double x = 2038.5;
   double y = -8999.0;
   System.out.println(Math.min(x,y)); //**-8999.0**

10. double x = 2038.5;
    double y = -8999.0;
    System.out.println(Math.max(x,y)); //**2038.5**

11. double x = 148.2;
    System.out.println(Math.round(x)); //**148**

    double x = 148.7;
    System.out.println(Math.round(x)); //**149**

    double x = -148.2;
    System.out.println(Math.round(x)); //**-148**

    double x = -148.7;
    System.out.println(Math.round(x)); //**-149**

12. System.out.println(Math.PI); //**3.14159265…**

Advanced *Math* methods:
Below are some additional *Math* methods that advanced math students will find useful:

Method	Signature	Description
log	double log(double x)	Returns log base e of x
sin	double sin(double a)	Returns the sine of angle a… a is in rad
cos	double cos(double a)	Returns the cosine of angle a… a is in rad
tan	double tan(double a)	Returns the tangent of angle a… a is in rad
asin	double asin(double x)	Returns arcsine of x…in range -PI/2 to PI/2
acos	double acos(double x)	Returns arccosine of x…in range 0 to PI
atan	double atan(double x)	Returns arctan of x. in range -PI/2 to PI/2
toDegrees	double toDegrees(double angRad)	Converts radians into degrees
toRadians	double toRadians(double angDeg)	Converts degrees into radians

Exercise on Lesson 6

1. Write code that will take the square root of *x* and store the result in *y*.

2. Write code that will multiply the value of the integer *j* times the absolute value of the integer *m* and then store the result in the integer *k*.

3. Is the following legal? If not, what would you do to make it legal?
   ```
   int k = Math.abs(-127.5);
   ```

4. Write a statement that will print the result of $2^{1.5}$.

5. System.out.println(Math.ceil(-157.2));

6. System.out.println(Math.floor(-157.2));

7. System.out.println(Math.ceil(157.2));

8. System.out.println(Math.floor(157.2));

9. System.out.println(Math.round(-157.2));

10. System.out.println(Math.ceil(-157.7));

11. System.out.println(Math.ceil(157));

12. System.out.println(Math.ceil(157.7));

13. Write a statement that will print the natural log of 18…. same as ln(18) on a calculator.

14. Write a line of code that multiplies *double p* times π and stores the result in *b*.

Project… Compute This

Create a new project called *ComputeThis* having a class called *Tester*. The *main* method of *Tester* should calculate the value of the following formulas and present the answers as shown.

$d1 = 3\pi\sin(187°) + |\cos(122°)|$ …Remember that the arguments of sin and cos must be in radians.

$d2 = (14.72)^{3.801} + \ln 72$ …ln means log base e

The output of your code should appear as follows:

```
d1  =  -0.618672237585067

d2  =  27496.988867001543
```

Verify these answers with a calculator.