java.lang.Math

java.lang.Math is a class in the Java API that contains utilities for performing higher mathematical operations. You cannot build Math objects and, since all of its fields and methods are static, you don’t have to. Also, as with all java.lang classes, Math doesn’t need to be imported into your application.

<table>
<thead>
<tr>
<th>Class Constants Provided by java.lang.Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>static final double E</td>
</tr>
<tr>
<td>// The double value that is closer than any other to e, the base of the natural logarithms.</td>
</tr>
<tr>
<td>static final double PI</td>
</tr>
<tr>
<td>// The double value that is closer than any other to pi, the ratio of the circumference of a circle to its diameter.</td>
</tr>
</tbody>
</table>

**Example**
Calculate the circumference of a circle of radius $r$ using the formula $circumference = 2\pi r$.

```java
double circumference, r;
...
circumference = 2.0 * Math.PI * r;
```

<table>
<thead>
<tr>
<th>A Short List of Class Methods Available in java.lang.Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>static double abs(double a)</td>
</tr>
<tr>
<td>// Returns the absolute value of a double value.</td>
</tr>
<tr>
<td>static int abs(int a)</td>
</tr>
<tr>
<td>// Returns the absolute value of an int value.</td>
</tr>
<tr>
<td>static double ceil(double a)</td>
</tr>
<tr>
<td>// Returns the smallest (closest to negative infinity) double value that is greater than or equal to the argument and is equal to a mathematical integer.</td>
</tr>
<tr>
<td>static double exp(double a)</td>
</tr>
<tr>
<td>// Returns Euler’s number $e$ raised to the power of the argument.</td>
</tr>
</tbody>
</table>


static double floor( double a )
// Returns the largest (closest to positive infinity) double
// value that is less than or equal to the argument and is equal
// to a mathematical integer.

static double log( double a )
// Returns the natural logarithm (base e) of the argument.

static double max( double a, double b )
// Return the largest of the arguments.

static double min( double a, double b )
// Return the smallest of the arguments.

static double pow( double a, double b )
// Returns the value of the first argument raised to the power
// of the second argument.

static double random( )
// Returns a double value with a positive sign, greater than or
// equal to 0.0 and less than 1.0.

static double rint( double a )
// Returns the double value that is closest in value to the
// argument and is equal to a mathematical integer.

static long round( double a )
// Returns the closest long to the argument.

static double sqrt( double a )
// Returns the correctly rounded positive square root of a
// double value.

### Examples

double x, y;
  . . .
y = Math.abs( x );

int n, m;
  . . .
m = Math.abs( n );

<table>
<thead>
<tr>
<th>x</th>
<th>-1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>n</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>1</td>
</tr>
</tbody>
</table>
### Examples

<table>
<thead>
<tr>
<th>Expression</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>double x, y; y = Math.ceil( x );</td>
<td>x = 2.4, y = 3.0</td>
</tr>
<tr>
<td>double x, y; y = Math.floor( x );</td>
<td>x = 2.9, y = 2.0</td>
</tr>
<tr>
<td>double x, y, p; p = Math.pow( x, y );</td>
<td>x = 2.0, y = 5.0, p = 32.0</td>
</tr>
<tr>
<td>double x, y, p; p = Math.pow( y, x );</td>
<td>x = 2.0, y = 5.0, p = 25.0</td>
</tr>
<tr>
<td>double x; // get randomly chosen fraction</td>
<td></td>
</tr>
<tr>
<td>x = Math.random();</td>
<td>x = 0.08397117700688295</td>
</tr>
<tr>
<td>double w, x, y, z; y = Math.rint( w ); z = Math.rint( x );</td>
<td>w = 2.4, x = 2.6, y = 2.0, z = 3.0</td>
</tr>
<tr>
<td>double x, y; long j, k; j = Math.round( x ); k = Math.round( y );</td>
<td>x = 2.4, y = 2.6, j = 2, k = 3</td>
</tr>
<tr>
<td>double x, y; y = Math.sqrt( x );</td>
<td>x = 36.0, y = 6.0</td>
</tr>
</tbody>
</table>
Math Examples

**Example**
Calculate the area of a circle of radius \( r \) using the formula \( \text{area} = \pi r^2 \).

```java
double area, r;
. . .
area = Math.PI * Math.pow( r, 2 );
```

**Example**
The Theorem of Pythagoras says: *in a right triangle the square of the hypotenuse is equal to the sum of the squares of the other two sides*. For example, the triangle shown to the right is a right triangle because it has a 90\( ^\circ \) angle. The hypotenuse is \( c \), the longest side.

Calculate \( c \) given \( a \) and \( b \).

```java
double a, b, c;
. . .
c = Math.sqrt( Math.pow( a, 2 ) + Math.pow( b, 2 ) );
```

**Example**
Round the value in \( x \) to the nearest integer and put it into \( m \) without changing \( x \).

```java
double x;
long m;
. . .
m = Math.round( x );
```

**Example**
Round the value in \( x \) to the nearest one-hundredth. For example, \( \frac{1}{3} \) becomes 0.33 whereas \( \frac{2}{3} \) becomes 0.67.

```java
double x;
. . .
x = Math.rint( x*100 ) / 100;
```
**Example**
Round the value in `x` to the nearest one-hundred. For example, 2,149 becomes 2,100 whereas 2,150 becomes 2,200.

```java
double x;
. . .
x = Math.round( x/100 ) * 100;
```

**Example**
Students taking courses at State College pay $239 per credit hour for tuition up to a maximum of $3,000. For example a student taking 12 credit hours pays \(239 \times 12 = 2,868\) whereas a student taking 13 credit hours pays $3,000 since \(239 \times 13 = 3,107\).

Calculate a student’s total tuition given his or her number of credit hours.

```java
final double FEE_PER_CREDIT = 239;
double creditHours, totalTuition;
. . .
totalTuition = Math.min( 3_000, creditHours*FEE_PER_CREDIT );
```

**Example**
Most hand-held calculators compute \(a^b\) using the formula \(e^{b \ln a}\), where \(e\) is Euler’s number and \(\ln a\) is the natural logarithm of \(a\). The formula works because \(a^b = e^{\ln a^b} = e^{b \ln a}\).

Use the formula to calculate \(a^b\), given \(a\) and \(b\).

```java
double a, b, power;
. . .
power = Math.pow( Math.E, b*Math.log( a ) );
```

**Example**
Another solution to \(a^b = e^{b \ln a}\).

```java
double a, b, power;
. . .
power = Math.exp( b*Math.log( a ) );
```
# Exercises

For each of the following exercises, write the Java statements to perform the specified calculation using appropriate methods from `java.lang.Math`. Be sure to declare whatever variables you need.

1. Pretend you drop a ball from the top of a tall tower. You can calculate the tower’s height knowing how long it takes the ball to reach the ground, and vice-versa. Ignoring the slowing effect of air resistance, the relationship between time and height is:

\[ h = \frac{1}{2} gt^2 \]

\( h \) is the height in meters, \( t \) is the time in seconds and \( g \) is the force of gravity (9.81 m/sec²). Solving for \( t \):

\[ t = \sqrt{\frac{2h}{g}} \]

Calculate the height \( t \) given the height \( h \) in feet (not meters).

2. Calculate the volume of a cube using the formula \( \text{volume} = \text{edge}^3 \), where \( \text{edge} \) is the length of the edge of the cube.

3. Calculate the volume of a sphere using the formula \( \text{volume} = \frac{4}{3} \pi r^3 \).

4. The distance of the point \((x, y)\) from the origin \((0, 0)\) of a Cartesian plane can be calculated using the Theorem of Pythagoras as shown in the picture.

\[ d = \sqrt{x^2 + y^2} \]

Calculate \( d \) given \( x \) and \( y \).
5. The distance between two points on a Cartesian plane can be calculated by applying the Theorem of Pythagoras to the distance between the x’s and the y’s (x and y in the picture to the right).

Given the coordinates of two points (x₁, y₁) and (x₂, y₂), calculate the distance d between them.

6. Given two points on a Cartesian plane (x₁, y₁) and (x₂, y₂), the midpoint between them is:

\[
\left( \frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2} \right)
\]

Given (x₁, y₁) and (x₂, y₂), calculate the midpoint, rounding its coordinates to the nearest integer.
7. Suppose \((x_1, y_1)\) and \((x_2, y_2)\) are opposing corners of a rectangle. For example, in the picture at right:

\[(3, 7)\] opposes \((10, 2)\)
\[(3, 2)\] opposes \((10, 7)\)

Given the coordinates \((x_1, y_1)\) and \((x_2, y_2)\) of opposing corners, calculate the rectangle’s height, width and the coordinates of its upper-left corner. The code must work no matter which two opposing corners are given and no matter which order they are given in.

For example, given \((x_1, y_1) = (10, 7)\) and \((x_2, y_2) = (3, 2)\) the height is 5, the width 7 and the upper-left corner is \((3, 7)\).