Mathematical Library Methods

Java is known for its interoperability, portability and the ease with which a programmer can carry out all the tasks. Java also provides reusability for its code. The main reason why programmers prefer java is due to its object-oriented approach which makes it very easy to write programs.

The java.lang package is one amongst the many predefined packages that java has to offer. Packages help in reducing the lines of code and making the programmers job easy. The java.lang package consists all the classes and interfaces that are the building blocks of any java program. It has classes like Object, Threads, String, Math and so on, so as to make your job easier. Also, without importing this packages 90% of your java program is incomplete because, it might not understand many aspects of the program during compilation.

Introducing java.lang package

The java.lang package consists of all the essential elements of any Java program. The following are the most important classes which are present in the java.lang package :

1. Object

2. String

3. Math

4. StringBuffer

NOTE: The java.lang package is the only package in Java which is implicitly used in the program. It means, that if you do not mention the syntax for importing java.lang package the program will automatically import it during compilation or running the program.

The syntax for importing java.lang package is

import java.lang.*;

Important classes in java.lang package

It is very important to know the basic classes that are the building blocks of Java.Wehave mentioned some of them for your reference and clear understanding. Just remembering the classes will help you when you are writing the code.

Also, this will give you a detailed understanding of the important classes and will help you remember the basic concepts of this classes.

We will understand their application, but let us stick to the basics right now

Here is the table of basic java.lang classes in the following table:

Classes Description

class Math Math class helps in performing the numeric operations. It provides only static methods. It consists of values for mathematical constant e. It also helps in finding out values for trigonometric functions which is a huge task when it comes to hard code each and every function for solving a trigonometric question through programming.

class Object As we know, Java follows object-oriented paradigm, it has always been communicating using classes and objects. So, whenever we make a class, object is always a base class for all classes.

class String Strings are present in all the major applications and considered as the important part of any java application. It is not only string but also some methods like indexOf, length, split, and substring that it offers.

class StringBuffer This class provides an alternative for String class. String class offers length, split, and substring methods but when it comes to building u strings from two different strings, StringBuffer comes in to picture with his own methods append, toString and insert to name the major ones.

Please note that the method names used in the table above are case sensitive and need to be followed as it is.

In this chapter, we will understand some Mathematical function libraries from the class Math of java.lang package.

The math class consists of all the numerical, static values and fields. It helps us for performing difficult and cumbersome mathematical operations with ease and less lines of code.

There are different math class methods for all types of Mathematical operations from arithmetic to trigonometric, logarithmic to exponential and so on.

The Math class

The math class of java.lang package will be responsible for providing the static methods to implement math functions, which will help us to perform arithmetic operation smoothly. Not only, arithmetic, also can the trigonometric functions be used from the math class. We will understand the working of each class using a programming example.

Let us learn the Math classes one by one along with the example. We have different types of Math classes divided in the following major classification:

1. Trigonometric

2. Exponentiation/Root

3. Utility

4. Rounding

Utility classes

There are various classes in utility classes, they are described as follows:

1. abs(): The abs() function stands for absolute. Absolute number in our common mathematics can be understood using the modulus operator. So if |a|, of real number a used to be the numerical value of a. For example, if we had to calculate the |-10| we used to get 10, |-3.22| used to be 3.22. So, abs() function will also do the same. It will display only the numeric value of the given value. Let us understand by using the following code:

```
public class Absolute {

public static void main(String[] args) {

double z = -21.65;

int q = 123;

double c = 13.44;

System.out.println("absolute value of z is "+Math.abs(z));

System.out.println("absolute value of q is "+Math.abs(q));

System.out.println("absolute value of c is "+Math.abs(c));

}

}
```


The output looks like the following screenshot:

Fig 1.1 Output of abs() function

2. max(): The Math.max() method helps us in finding the maximum number out of the two numbers. We need to remember it can take only two arguments at a time. Let us understand this by the following code:

```
public class Max {

public static void main(String[] args) {

System.out.println("Max value of 525, 655 is: "+Math.max(525, 655));

System.out.println("Max value of 132.23, 132.245 is: "+Math.max(132.23, 132.245));

System.out.println("Max value of 47.23, 245.321 is: "+Math.max(41.89, 23.45));

}
```

```
}
```

The output of the preceding code looks like the following screenshot:

Fig 1.2 Output of max() function

3. min(): The Math.min() method helpsin finding the minimum of two numbers.We need to remember that it can only take upto two arguments at the time. Let us understand this by the following code:

```
public class Min {

public static void main(String[] args) {

System.out.println("Min value of 525, 655 is: "+Math.min(525, 655));

System.out.println("Min value of 132.23, 132.245 is: "+Math.min(132.23, 132.245));

System.out.println("Min value of 47.23, 245.321 is: "+Math.min(41.89, 23.45));

}

}
```

The output of min function is as follows:

Fig 1.3 Output of min function

4. hypot(): In geometry, the hypotenuse is the longest side of a triangle. We determine the hypotenuse as the side opposite to right angle. Java does this using the hypot() function.

We will understand more about this when we learn the code. Let us understand this by using the following code:

```
public class Hypotenuse {

public static void main(String[] args) {

int x = 2;

int y = 20;

System.out.println("The length of hypotenuse will be: "+Math.hypot(x, y));

}

}
```

The output of hypot() function is as follows:

Fig 1.4 Output of hypot() function

Exponential/root classes

The exponential or root classes helps us find out the square root values and exponential values. The classes are:

1. sqrt(): This function returns a rounded positive square root of any number. Let us understand this using the following example:

```
public class Squareroot{

public static void main(String a[]){

System.out.println("Square root of 16 is: "+Math.sqrt(196));

System.out.println("Square root of 4.6 is: "+Math.sqrt(153.02));

System.out.println("Square root of 21 is: "+Math.sqrt(1000000));

}

}
```

The output of sqrt() function is as follows:

Fig 1.5 Output of sqrt() function

2. cbrt(): This function returns a cube root of any number. Let us understand this using the following example:

```
public class Cuberoot{

public static void main(String a[]){

System.out.println("Cube root of 196 is: "+Math.cbrt(196));

System.out.println("Cube root of 153.02 is: "+Math.cbrt(153.02));

System.out.println("Cube root of 1000000 is: "+Math.cbrt(1000000));

}

}
```

The output of the preceding code is as follows:

Fig 1.6 Output of cbrt() function

3. pow(): This function returns the value of raising the first argument to second argument. Let us understand this using the following example:

```
public class Powerfun {

public static void main(String a[]){

System.out.println("10 to the power of 3 is: "+Math.pow(10, 3));

System.out.println("12 to the power of 5 is: "+Math.pow(12, 5));

System.out.println("69 to the power of 3 is: "+Math.pow(69, 3));

}

}
```

The output of the preceding code is as follows:

Fig 1.7 Ouput of pow() function

4. exp(): This function returns the value of e raised to the power of x (ex). Let us understand this using the following example:

```
public class Exponential {

public static void main(String a[]){

System.out.println("Exponent value of 10.1 is: "+Math.getExponent(10.1));

System.out.println("Exponent value of 20 is: "+Math.getExponent(20));

System.out.println("Exponent value of 6 is: "+Math.getExponent(6));

}

}
```

The output of exp() functions is as follows:

Fig 1.10 Output of exp() function

Trigonometric classes

Different Trigonometric classes in java.lang.math are as follows:

1. sin(): The sin() is used to find the trigonometric sine value of an angle. Let us see the following example code:

```
public class TrigoSin {

public static void main(String a[]){

System.out.println("Value of sin(60) is: "+Math.sin(30));

System.out.println("Value of sin(30) is: "+Math.sin(60));

System.out.println("Value of sin(90) is: "+Math.sin(90));

}

}
```

The output of the preceding code is as follows:

Fig 1.11 Output of sin() function

2. cos(): The cos() is used to find the trigonometric cosine of an angle. Let us understand using the following example.

```
public class TrigoCos {

public static void main(String a[]){

System.out.println("Value of cos(60) is: "+Math.cos(30));

System.out.println("Value of cos(30) is: "+Math.cos(60));

System.out.println("Value of cos(90) is: "+Math.cos(90));

}

}
```

The output of the preceding code is as follows:

Fig 1.12 Output of cosine() function

3. tan(): The tan() is used to find the trigonometric tangent value of any angle. Let us understand this using the following example:

```
public class TrigoTan {

public static void main(String a[]){

System.out.println("Value of tan(60) is: "+Math.tan(30));

System.out.println("Value of tan(30) is: "+Math.tan(60));
```

```
System.out.println("Value of tan(90) is: "+Math.tan(90));

}

}
```

The output of the preceding code is as follows:

Fig 1.13 Output of tan() function

Rounding classes

There are different classes for rounding off the decimal numbers described as follows:

1. rint(): The rint() class returns the double value of the provided argument and which is equal to the nearest integer. Let us understand this by using a small piece of code:

```
import java.lang.*;

public class Rint {

public static void main(String[] args) {

System.out.println("Rint value of 21.559 is: "+Math.rint(21.159));

System.out.println("Rint value of -45.665 is: "+Math.rint(-45.665));

System.out.println("Rint value of 22.56 is: "+Math.rint(22.56));

System.out.println("Rint value of 4.59 is: "+Math.rint(4.59));

}

}
```

The output of the preceding code is as follows:

Fig 1.14 Output of rint() function

2. floor(): The floor() class returns the largest integer value which is not greater than the argument value

```
import java.lang.*;

public class Floor {

public static void main(String[] args) {

System.out.println("Floor value of 21 is: "+Math.floor(21));
```

```
System.out.println("Floor value of -45.6 is: "+Math.floor(-45.6));

System.out.println("Floor value of 2.1 is: "+Math.floor(2.1));

System.out.println("Floor value of 4.09 is: "+Math.floor(4.09));

}

}
```

The output of the preceding code is as follows:

Fig 1.15 Output of floor() function

NOTE: Before using any math function, it is mandatory to follow the function name with Math.

Java expressions

Expressions in Java can be understood as the fundamental concept of writing any complex Java program. Expressions are used to create a new value, but not limited to it. It can also be used to assign a value to a new variable for using it as per the needs. Expressions are built by using the variables, values and function calls.

Expressions are meant to produce a result for most of the time, but that is not true for all the cases. We can classify expressions on the basis of their behavior:

1. Expressions which produce a value

2. Expressions which assign a value

3. Expressions which have no result, because it might be a part of the function call sometime

Expressions which produce a value

There are a lot of operators in java from addition, subtraction to division and modulus. The expressions which produce a value are those which use one or all the above operators for producing value.

For example, some arithmetic operators like +, -, *, / , >,

Examples of expressions that could produce a value are:

3/4

5%6

Expressions which assign a value

Assignment is used to assign a value to the variable on the left. Values to be assigned are

always written on the right.

int a = 100;

int seconds = 100;

The preceding are the examples of expressions that are used to assign a value.

Expressions with no result

There are times when the expression is not assigning any value nor does it produce any result. But there might be some result when the operands change.

For example, if we have an expression always produce a side effect:

int product = x *z;

For example, this value does not produce any value for us when we run the program containing the preceding expression. It is executed only for the effect of it. This mostly happens in when you are trying to call the function from some other functions.