Math class

The Java Math class has many methods that allows you to perform mathematical tasks on numbers.

Math.max(*x*,*y*)

This method can be used to find the highest value of x and y:

Example,
Math.max(6, 10);

Math.min(*x*,*y*)

This method can be used to find the lowest value of x and y:

Example,
Math.min(6, 10);

Math.sqrt(*x*)

This method returns the square root of x:

Example,
Math.sqrt(81);

Math.abs(x)

The Math.abs(x) method returns the absolute (positive) value of x:

Example,
Math.abs(-5.6);

Random Numbers

Math.random() returns a random number between 0.0 (inclusive), and 1.0 (exclusive):

```
Math.random();
int randomNum = (int)(Math.random() * 101);  // 0 to 100
```

```
// To print random number between 0 to 50
public class Main
{
  public static void main(String[] args)
 {
    int randomNum = (int)(Math.random() * 50);  // 0 to 50
    System.out.println(randomNum);
  }
}
```

```
// To print bigger number
public class Mtah
{
  public static void main(String[] args)
    {
        System.out.println(Math.max(6, 10));
    }
}
```

Conditional statements in Java:

- If to specify a block of code to be executed, if a specified condition is true
- If else to specify a block of code to be executed, if the same condition is false
- If…else if to specify a new condition to test, if the first condition is false

Operators used with conditional statements to make different decisions

- Less than: a < b
- Less than or equal to: a <= b
- Greater than: a > b
- Greater than or equal to: a >= b
- Equal to a == b
- Not Equal to: a != b

Syntax

```
if (condition)
 {
  // block of code to be executed if the condition is true
}
```

Example_1:

```
public class Main
 {
  public static void main(String[] args)
 {
    if (50 > 25)
{
     System.out.println(“50 is greater than 25");
    }
  }
}
```

Example_2:

```
public class Main {
  public static void main(String[] args) {
    int x = 2;
    int y = 1;
    if (x > y) {
      System.out.println("x is greater than y");
    }
  }
}
```

Example_3:

```
class IfStatement {
  public static void main(String[] args) {
    int number = 10;
    if (number < 0)
   {
      System.out.println("The number is negative.");
    }
    System.out.println("Statement outside if block");
  }
}
```

Example_4:

```
class Main
 {
  public static void main(String[] args)
{
    // create a string variable
    String language = "Java";

    // if statement
    if (language == "Java")
 {
      System.out.println("Best Programming Language");
    }
  }
}
```

Syntax:

```
if (condition)
 {
  // block of code to be executed if the condition is true
}
 else
 {
  // block of code to be executed if the condition is false
}
```

```
class Main {
  public static void main(String[] args)
{
    int number = 10;

if (number > 0)
 {
     System.out.println("The number is positive.");
   }

else
 {
     System.out.println("The number is not positive.");
   }

   System.out.println("Statement outside if...else block");
  }
}
```

Java if...else...if Statement

In Java, we have an **if...else...if** ladder, that can be used to execute one block of code among multiple other blocks.

Syntax:

```
if (condition1)
{
  // code
}
else if(condition2)
 {
  // code
}
else if (condition3)
 {
  // code
}
.
.
Else
 {
  // code
}
```

When the test condition of if is true, codes inside the body of that if block is executed. And, program control jumps outside the if...else...if ladder.

If all test expressions in if are false, codes inside the body of else are executed.

```
class Main {
  public static void main(String[] args) {

    int number = 0;

    // checks if number is greater than 0
    if (number > 0) {
      System.out.println("The number is positive.");
    }

    // checks if number is less than 0
    else if (number < 0) {
      System.out.println("The number is negative.");
    }

    // if both condition is false
    else {
      System.out.println("The number is 0.");
    }
  }
}
```

Switch Statement

- Instead of writing many if..else statements, you can use the switch statement.

- The switch statement selects one of many code blocks to be executed

```
switch(expression)
{
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

```
public class Main {
  public static void main(String[] args) {
    int day = 4;
    switch (day) {
      case 1:
        System.out.println("Monday");
        break;
      case 2:
        System.out.println("Tuesday");
        break;
      case 3:
        System.out.println("Wednesday");
        break;
      case 4:
        System.out.println("Thursday");
        break;
      case 5:
        System.out.println("Friday");
        break;
      case 6:
        System.out.println("Saturday");
        break;
      case 7:
        System.out.println("Sunday");
        break;
    }
  }
}
```

default Keyword

The default keyword specifies some code to run if there is no case match

```
public class Main {
  public static void main(String[] args) {
    int day = 4;
    switch (day) {
      case 6:
        System.out.println("Today is Saturday");
        break;
      case 7:
        System.out.println("Today is Sunday");
        break;
      default:
        System.out.println("Looking forward to the
Weekend");
    }
  }
}
```

Java for Loop

In computer programming, loops are used to repeat a block of code. For example, if you want to show a message **50** times, then rather than typing the same code **50** times, you can use a loop.

In Java, there are three types of loops.

- for loop
- while loop
- do…while loop