**Pluralsite\_Course for java trainer is Jim Wilson**

1) Types of comments:  
 1) Line comment – This will hide the line by line we using the symbol is //  
 2) Block Comment – This will comment the block using the symbol is /\* -- \*/ This will help us text ignored within block  
 3) Javadoc comment – This also same as block comment using the symbol is /\*\*--- \*/ It can be used generate a documentation.

2) Packages:  
 \* Packages provides organization  
 \* Follow standard naming convention  
 \* Affect source code structure  
 Package naming conventions  
 \* All lowercase  
 \* Use reverse domain name notation to assure global uniqueness.  
 \* Add further qualifiers to assure uniqueness within in a company or group  
Ex: package com. pluralsight.search;  
 package com.pluralsight.sales.accountmanagement;  
 package com. pluralsight.humanresources.accountmanagement;  
src -> com -> pluralsight -> example -> Main.java

3) Variables:  
 \* Variables naming use only for letters and numbers.  
 \* First character cannot be a number  
 \* Start each world after the first with uppercase  
 \* Variables can be declared final keyword; we cannot be changed once we set

4) Data types:  
 There are four types of data types are:  
 1) Integer type: Byte, Short, int, long   
 2) character type: char  
 \* Store a single Unicode character  
 \* Literal value placed between single quotes  
 \* For Unicode code points, use \u followed by 4-digit hex value   
 3) Boolean type: True or false   
 2) Floating point type: float and double

5) Prefix, Postfix and compound operation:  
 Prefix: ++val It will increment  
 postfix: val++ It will not increment once we print another s.opln(variableName) then it will increment

Compound assignment operator:  
 Combine an operation and assignment  
 \* Apply right side value to left side  
 \* Store result in variable on left side  
 Available for 5 basic math operation are:  
 1) += 2) -= 3) \*= 4) /= 5) %=

1) int myValue = 50; 2) int myValue = 50;   
 myValue += 5; myValue -= 5;  
 System.*out*.println(myValue);*//55* System.*out*.println(myValue);*//45*

3) int myValue = 50; 4) int myValue = 50;  
 myValue \*= 5; myValue /= 5;  
 System.*out*.println(myValue);*//250* System.*out*.println(myValue);*//10*

5) int myValue = 101;  
 myValue %= 3;  
 System.*out*.println(myValue);*//2*

int valA = 21;  
int valB = 6;  
int valC = 3;  
int valD = 1;  
  
int result1 = valA - valB / valC;  
int result2 = (valA - valB) / valC;  
  
System.*out*.println(result1);*//19*System.*out*.println(result2);*//5*

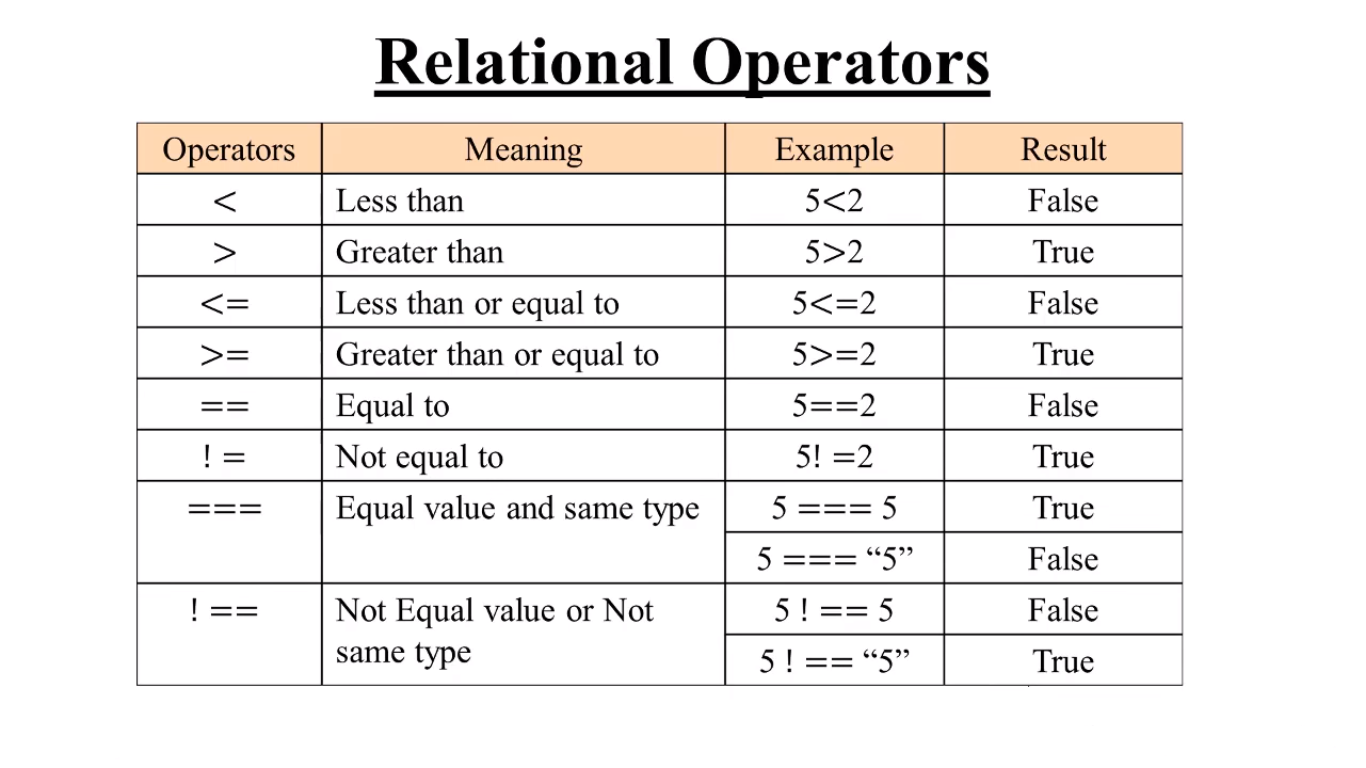
6) Type conversion

It means we are converting the one data type to another data types  
 1) Implicit type conversion:  
 \* In this conversion is automatically performed by the compiler.

\* Widening conversions are performed by automatically  
 Ex: int intValueOne = 50;  
 long longValueOne = intValueOne;  
 System.*out*.println(longValueOne);*//50*

2) Explicit type conversion:  
 \* In this conversion performed explicitly in code with cast operator  
 \* Can perform widening or narrowing conversions be aware of potential side-effects  
 Ex: long longValueTwo = 50;  
 int intValueTwo = (int) longValueTwo;  
 System.*out*.println(intValueTwo);*//50*

float floatVal = 1.0f;  
double doubleVal = 4.0d;  
byte byteVal = 7;  
short shortVal = 7;  
long longVal = 5;  
  
short result1 = byteVal;  
short result2 = (short) longVal;  
short result3 = (short) (byteVal - longVal);  
  
System.*out*.println("Success");

11) Conditional logic and block statements:  
 

int value1 = 7;  
int value2 = 5;  
  
int maxValue = value1 > value2 ? value1 : value2;  
System.*out*.println(maxValue);*//7*

**If else statement example:**  
  
 int value1 = 5;  
 int value2 = 7;  
  
if(value1 > value2){  
 System.*out*.println("value1 is bigger");  
 *//true-statement*}else {  
 System.*out*.println("value1 is not bigger");// value1 is not bigger  
 *//false-ststement*}

**Chaining if-else statements:**

* Evaluated in order top-to-bottom.
* First to test true is executed.

Ex: if (condition 1)

true – statement – 1;  
 else if (condition 2)

true – statement – 2;  
 ----  
 else if (condition-N)  
 true – statement – N  
 else  
 false-statement;

Important example for chaining if-else statement:  
 public class Main {  
 public static void main(String[] args) {  
  
 double value1 = 100.0d;  
 double value2 = 50.0d;  
 double result = 0.0d;  
 char opCode = 'f';  
  
 if(opCode == 'a')  
 result = value1 + value2;*//150* else if (opCode == 's')  
 result = value1 - value2;*//50* else if (opCode == 'm')  
 result = value1 \* value2;*//5000* else if (opCode == 'd')  
 result = value1 / value2;*//0.0d* else  
 result = 0.0d;*//0.0* System.*out*.println(result);  
  
 }  
}

Logical operators:

* Produce a single true or false result from two true or false values.
* May combine two relational tests.
* May combine two Boolean variables.

1. And – operator & - true &true
2. Or - operator | - false|true true|fasle true|true
3. Exclusive or - operator ^ - false^true true^false
4. Negation - operator ! - false

Conditional Logical Operators:

* It similar to standard logical operators
* Right side executes only when needed
* && executes Right only when left is true
* || executes right only when left is false

EX: And - operator && true && true  
 Or - operator || false||true true||----

public class Main {  
 public static void main(String[] args) {  
 int students = 150;  
 int rooms = 0;  
  
 if(rooms !=0 && students/rooms >30){  
 System.*out*.println("crowded");  
 }  
 System.*out*.println("\*\*end of the program\*\*");// end of the program  
 }  
}

Block statement:

* Groups statement together.
* Creates a compound statements.
* Enclose statements in opening and closing brackets.

Ex: {  
 statement – 1;  
 statement – 2;  
 ----  
 statement – N;

}

public class Main {  
 public static void main(String[] args) {  
 int v1 = 10;  
 int v2 = 4;  
  
 final int diff;  
  
 if(v1 > v2)  
 diff = v1 - v2;  
 System.*out*.println("v1 is bigger than v2, diff = " + diff);  
 else// here we got an error  
 diff = v2 - v1;  
 System.*out*.println("v1 is not bigger than v2, diff = " + diff");  
 }  
}

We solving this error using block statement  
 public class Main {  
 public static void main(String[] args) {  
 int v1 = 10;  
 int v2 = 4;  
  
 final int diff;  
  
 if(v1 > v2) {  
 diff = v1 - v2;  
 System.*out*.println("v1 is bigger than v2, diff = " + diff);*//v1 is bigger than v2, diff = 6* }  
 else {  
 diff = v2 - v1;  
 System.*out*.println("v1 is not bigger than v2, diff = " + diff);  
 }  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 double students = 30.0d;  
 double rooms = 4.0d;  
  
 if(rooms > 0.0d) {  
 System.*out*.println(students);  
 System.*out*.println(rooms);  
  
 double avg = students/rooms;  
 System.*out*.println(avg);  
 }  
 }  
}

Switch statement:

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

**Looping and arrays:**

1. While loop 2) Do-while loop 3) For loop 4) Arrays 5) For-each loop
2. Example for the while loop  
   public class Main {  
    public static void main (String[] args) {  
    int someVal = 4;  
    int factorial = 1;  
     
    while (someVal > 1) {  
    factorial \*= someVal;  
    someVal--;  
    }  
    System.*out*.println(factorial);  
    }  
   }
3. Example for the Do-while loop:  
   \* Condition checked at the loop end   
   \* Loop body always runs at least once.

Ex: do

Statement;  
 while(condition)

public class Main {  
 public static void main (String[] args) {  
 int val = 5;  
 do {  
 System.*out*.println(val);  
 System.*out*.println(" \* 2 = ");  
  
 val \*= 2;  
 System.*out*.println(val);  
 }while (val < 25);  
 }  
}

1. For Loop

* Condition checked at the loop end.
* Loop body may never run
* Simplified notation for loop control values
* It will same as the while loop, and we achieve less code compare to while loop.

Ex: for (initialize; condition; update)

Statement.

public class Main {  
 public static void main(String[] args) {  
 int i = 1;  
  
 while (i < 100) {  
 System.*out*.println(i);  
  
 i \*=2;  
 }  
  
 for (int j = 1; j <100 ; j \*= 2) {  
 System.*out*.println(j);  
 }  
 }  
}

1. Arrays:

* Provides an ordered collection of elements.
* Each element accessed via an index.
* Index range from 0 to number-of-elements minus 1.
* Number of elements can be via array’s length value.

public class Main {  
 public static void main(String[] args) {  
 float[] theVal = new float[3];  
 theVal[0] = 10.0f;  
 theVal[1] = 20.0f;  
 theVal[2] = 15.0f;  
  
 float sum = 0.0f;  
  
 for (int index = 0; index < theVal.length; index++) {  
 sum += theVal[index];  
 System.*out*.println(sum);  
 }  
 }  
}

or we can simply writ this code like this   
  
public class Main {  
 public static void main(String[] args) {  
 float[] theVal = {10.0f, 20.0f, 15.0f};  
  
 float sum = 0.0f;  
  
 for (int index = 0; index < theVal.length; index++) {  
 sum += theVal[index];  
 System.*out*.println(sum);  
 }  
 }  
}//45

1. For-each loop:  
   \* Executes a statement once for each array member.  
   \* Handle getting collection length  
   \* Handles accessing each value.

For (loop-variables: array)

Statement;

public class Main {  
 public static void main(String[] args) {  
 float[] theVal = {10.0f, 20.0f, 15.0f};  
 float sum = 0.0f;  
  
 for (float currentVal : theVal) {  
 sum += currentVal;  
 }  
 System.*out*.println(sum);  
 }  
 }*//45.0*