**JAVA**

Group of statements – Methods

Group of methods – Class

Group of classes – Package

**Class:**

* A class is a group of methods.
* A class binds variables and methods (which deals with the variables) together into a capsule.
* A class consists of state and behaviour.
* A class is like a blueprint or design to create objects.
* An object is an instance of a class.

**Number System:**

1. Binary number – 0 & 1
2. Octal number – 0 to 7
3. Decimal number – 0 to 9
4. Hexa-decimal number – 0 to 9 and A to F

**Data types in Java:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr no** | **Data types** | **Default size** |  | **Default Value** |
| 1 | byte | 1byte | Numbers | 0 |
| 2 | short | 2bytes | 0 |
| 3 | int | 4bytes | 0 |
| 4 | long | 8bytes | 0 |
| 5 | float | 4bytes | Decimal numbers | 0.0 |
| 6 | double | 8bytes | 0.0 |
| 7 | char | 2bytes | Single character |  |
| 8 | String | depends of size | Sequence of character | Null |
| 9 | boolean | 1bit | True / False | false |

**Types of Operators:**

1. Arithmetic operator – +, -, \*, /, %
2. Relational operator – <, >, <=, >=, ==, !=
3. Logical operator – &&, ||, !
4. Bitwise operator – &, |, ^
5. Shift operator – <<, >>
6. Unary operator – ++, --
7. Assignment operator – +=,-=, /=, \*=, %=
8. Conditional operator –

Arithmetic Operators: These operators are used to arithmetic operations between two variables like addition (+), subtraction (-), multiplication (\*), division (/), modulus (%).

Rotational Operators: These operators are used to show the relation between two variables like Is Equal to (==), Not Equal to (!=), Greater Than (>), Less Than (<), Greater Than or Equal to (>=), Less Than or Equal to (<=).

Logical Operators: These operators are used to check whether an argument is True or False. There are 3 logical operators

1. Logical AND (&&) – True when both arguments are true
2. Logical OR (||) – True when either of one argument is true
3. Logical NOT (!)

Bitwise Operators (&, |, ^): We can apply therse operators for integral types

1. Bitwise AND (&) – True when both arguments are true else false
2. Bitwise OR (|) – True when at least one argument is true
3. Bitwise XOR (^) – True when both arguments are different

Shift Operators: They are of two types

1. Right Shift (>>)
2. Left Shift (<<)

Right Shift (>>): It is a binary operator that takes two numbers, right shifts the bits of the first operand & the second operand decides the number of places to shift.

Example: 5>>1 – 1 0 1 (binary number of 5)

Here 2nd operand 1 number of places to shift from right i.e., 1

Left Shift (<<): it is binary operator that takes two numbers, left shift the bits of the first operand and the second operand decides the number of places to shift.

Example: 5<<1 – 1 0 1 (binary number of 5)

Here we add zero to the left side depending the 2nd operand value,

1 0 1 0 = **10**

Unary Operators: (++, --):

Assignment Operators (+=, -=, \*=, /=): there are three types of assignment operators

1. Simple:

int x = 10

1. Chained:

int a, b, c, d

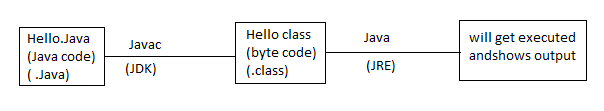
a = b = c = d = 10

1. Compound:

int a = 10

a += 10

* Java is platform **Independent.**



**Difference between JDK, JRE, JVK:**

JDK (Java Developing Kit): It provides the environment to develop and run the program (javac). (JDK=JRE + developing tools)

JRE (Java Runtime Environment): It provides the environment to run the program (java). (JRE=JVM + library tools)

JVM (Java Virtual Machine)

**Arrays:**

1. An array is indexed collection of fixed number of homogeneous data elements (i.e., collection of similar data types.). The main advantage of arrays is we can represent huge number of values by using single variable so that readability of the code will be improved
2. The main disadvantage of arrays is fixed in size i.e., once we fixed an array there is no chance of increasing or decreasing the size based on our requirement hence to use arrays concepts we used know the size in advance, which may not possible always.
3. Array index starts from zero.
4. Array size to be mentioned while declaring the variable. [directly or indirectly by initializing]
5. Arrays are stored in sequential order.
6. The size of array variable=data size \* array size.
7. Array size is fixed i.e. the size cannot be changed programmatically.

**Array creation:** int [] data = new int [5]

* 1. Here 5 is size of the array.
  2. [] represents array.
  3. data represents variable.

Example:

public static void main(String[] args)

{

int[] data = new int[] {10,45,64,99,70,19};

int big=data[0];

for(var d:data)

{

if(d>big)

big=d;

}

System.out.println("Big ="+big);

}