Creating the variable

```
// decleration
datatype varible_name ;
```

Initialization

```
// initialization
varible_name = value;
```

Reinitialization

In reinitialization value will get **modified** with **new value**. If programmer is trying to print a value from a variable it always gives the **modified value**

Direct declaration with initialization

```
float mark1 = 50.0f;
float mark2 = 100.0f;
float mark3 = 80.5f;
```

Reinitialize or modification of initialized value

```
mark1 = 90.0f;
mark2 = 70.0f;
mark3 = 89.5f;
System.out.println(mark1);
System.out.println(mark2);
System.out.println(mark3);
```

> **Note:** duplicate variables are not allowed

Scope of a variable

- **Visibility** of a variable is known as scope of a variable
- Based on the scope of the variable , variable is categorized into **3 types**

1. Local variable

2. Static variable
3. Non-Static variable

Local variable

Any variable which is declared inside of the method block or any other block is known as **local variable**

Characteristics of Local variable

- These variables are not stored with default values
- These variables must be initialized before using it
- The scope of a local variable is only inside this specific block, Hence it cannot be used outside of the block

> **Example :**

```
{
// block 1
int a = 10;
System.out.println(a);
}
{
// block 2
int a = 20;
System.out.println(a);
}
```

- Block 1 will print **10** and Block 2 will print **20** ,
-

because here the 2 a variables are local variable which can only act inside the block.

> **Note :**

- We can create variables with same name into different local blocks .

Datatype	Default Value	Size	
byte	0	1 byte	
short	0	2 byte	
int	0	4	byte
long	0	8 byte	

Datatype	Default Value	Size	
float	0.0f/F	2 byte	
double	0.0d/D	4 byte	
char	space (or) `\u000'	2 byte	
Boolean	false	1 bit	

Operators

Operators are the **predefined symbols** which is used to perform some specific tasks on given **operands**

Operands

Operands are the data given to the operator. Operands are 3 types

- **unary operator** - which works on **1 operand**
- **Binary operator** - which works on **2 operands**
- **Ternary operator** - which works on **3 or more operands**

Based on types of operators, operators are classified as

1. Arithmetic operator
2. Assignment operator
3. Relational operator
4. Logical operator
5. Increment/Decrement operator
6. Conditional operator

1) Arithmetic operator

Arithmetic operator is used to perform **arithmetic operation** on given operand. **Primitive values** should be used for arithmetic operator.

Arithmetic operations

```
int a = 10;
int b = 20;
```

1. Addition (+)

```
System.out.println(a+b) // 30
```

2. Subtraction (-)

```
System.out.println(a-b) // -20
```

3. Multiplication (*)

```
System.out.println(a*b) // 300
```

4. Division (/)

```
System.out.println(a/b) // 0
System.out.println(11/2) // 5
System.out.println(3/2) // 1
System.out.println(56/7) // 8
```

5. Modulus (%)

```
System.out.println(a%b) // 10
System.out.println(11%2) // 1
System.out.println(3%2) // 1
System.out.println(3%7) // 3
```

Assignment operator

- Assignment operator is used to assign a value a value to a variable
- **Assignment - =**

```
int a = 10;
Sytem.out.println(a) //10
```

- **Addition assignment - +=**

```
a += 5;
//a = a + 5;
Sytem.out.println(a) //5
```

- **Subtraction assignment - -=**

```
a -= 5;
// a = a - 5;
```

```
Sytem.out.println(a) //5
```

- **Multiplication assignment** - *=

```
a *= 5;
//a = a * 5;
Sytem.out.println(a) //50
```

- **Division** - /=

```
a /= 5;
// a = a/5;
Sytem.out.println(a) //2
```

- **Modulus** - %=

```
a %= 5;
// a = a % 5
Sytem.out.println(a) //0
```

Relational operator

- It is used to check the **relation** between **2 operands** . The **return type** of the this operator is **Boolean**
- **The operators are**

```
int a = 3;
int b = 2;
```

1. **Equality** - ==: Its is used to compare the operands are same .

```
boolean c = a==b;
System.out.println(c); // false
int x = 10;
int y = 10;
boolean z = x==y;
System.out.println(c); // true
```

2. **Less than** - <: Its checks that the left side value is lesser than the right side value

```
boolean d = a<b;
System.out.println(d); //false
```

```
```

3. **Greater than** - >: Its checks that the left side value is greater than the right side value.

```
boolean e = a>b;
System.out.println(e); //true
```

4. **Less than equals to** - <= Its checks that the left side value is less than or equals the right side value.

```
boolean f <= a>b
System.out.println //false
```

6. **Increment/Decrement** : there are 2 increment type they are 1) Pre-increment 2) Post-increment

```
int m = 5;
int n = 3;
int res = ++m + m++;
System.out.println(res);// 12
System.out.println(m);// 7
System.out.println(n);// 3
```

> Example :

```
package  operators;

public  class  IncDec {
public  static  void  main(String[] args) {
int  x  =  5;
int  y  =  10;
pint  z  =  ++x  +  x  +  --y  +  y++  +  y;
z++;
System.out.println(x);
System.out.println(y);
System.out.println(z);
}
}
```

5) Logical Operator:

- Its is used to perform **logical operation** on given **Boolean operands**.
- Its gives **Boolean result** The Logical operators are,

1. Logical AND (&&)
2. Logical OR (||)
3. Logical Not (!)

> OR table

01	02	res
F	F	F
F	T	T
T	F	T
T	T	T

> AND table

01	02	res
F	F	F
F	T	F
T	F	F
T	T	T