Arithmetic Operators(+, -, \*, /, %)

If we apply any arithmetic operator between a & b, the result type is always

max(int, type of a, type b)

byte + byte = int

byte + short = int

short + short = int

byte + long = long

long + double = double

float + long = float

char + char = int //sout(**'a'**+); 195

char + double = double //sout(**'a'**+0.89); //97.89

byte -> short -> int -> long -> float -> double

|

char

Infinity

In integral arithmetic(byte, short, int, long) there is no way to represent infinity hence if infinity is the result we will get Arithmetic exception in integral arthimetic

System.***out***.println(10/0); *//Arithmetic exception because there is no way to mention a word*

*//infinity in integral arithmentic*

But in floating point arthimetic(float and double) there is a way to represent infinity.

For float and double classes contains the following two constant

POSITIVE\_INFINITY;

NEGATIVE\_INFINITY;

Hence even tough result is infinity we wont get any arithmetic exception in floating point arithmetic

*System.****out****.println(10/0.0); //infinity because in floating point arithmetic there is a way to*

*//represent infinity*

System.***out***.println(-10/0.0); //-infinity

NaN(Not a number)

In intregral arithmetic(byte, short, int, long) there is no way to represent unfined result hence if the result is undefined we will get run-time exception ArthimeticException

System.***out***.println(0/0); *//ArthimeticException*

But in floating point Arithmetic(float & double) there is a way to represent Undefined result for this float and double classes contains NaN constant. Hence if the result is undefined we wont get arithmetic exception in floating point arthicmetic.

System.***out***.println(0.0/0); *//NaN constant*

System.***out***.println(-0.0/0); *//NaN constant*

Note: For any x value including NaN, the following expressions return **false**

System.***out***.println(10 < Float.***NaN***);

System.***out***.println(10 <= Float.***NaN***);

System.***out***.println(10 > Float.***NaN***);

System.***out***.println(10 >= Float.***NaN***);

System.***out***.println(10 == Float.***NaN***);

System.***out***.println(Float.***NaN*** == Float.***NaN***);

For any x value including NaN, the following expression return **true**

System.***out***.println(10 != Float.***NaN***);

System.***out***.println(Float.***NaN*** != Float.***NaN***);

**ArithmeticException:**

It is a Runtime Exception not compile time error

It is possible only in integral arithmetic but not in floating point arithmetic

The only operator which cause AE are / and %