**Day-08**

**27-02-2025**

**================**

**Literals:**

**======**

-> Literal is a value

-> Different types of literals are:

1) Integral Literals

2) Floating-point Literals

3) Character Literal -> 'a'

4) Boolean Literal -> true, false

1) Integral Literals

----------------------

-> are 4-types:

1) Decimal Literal

2) Binary Literal

3) Octal Literal

4) Hexadecimal Literal

-> Decimal literal is called as "base-10" literal.

because the decimal literal can allow to define with total of 10 letters.

Those are:

0 to 9

Ex: salary = 350000;

-> Binary literal is called as "Base-2 literal".

because this allow to define with only two letters

those are:

0 and 1

-> The binary literal in java can always prefix with '0b' or '0B'.

Ex: 1010101 ==> decimal by default

0b1010101 ==> binary

-> Octal literal is called as "base-8" literal.

because this can allow to define with the total of 8 letters.

those are:

0 to 7

Ex: 123, 1023 etc. ==> decimal by default

-> The octal literal in java can always prefix with '0'.

Ex: 0123, 01023 ==> octal

-> Hexadecimal literal also called as "Base-16" literal.

because , we can use the total of 16 letters to define the hexadecimal numbers.

those are:

0 to 9,

alphabets: a to f

Here:

a ==> 10, b = 11, c = 12, d = 13, e = 14 and f = 15

->also called as "alpha numeral"

-> Hexadecimal literals can always be prefix with '0x' or '0X'

Ex: 0x123, 0XAF12FA etc.

-> Long variable can accept the value from byte range, short range, int range and also long range.

-> long variable value must be suffixed with 'l' or 'L'.

-> Order for the integer datatypes:

byte << short << int << long

class IntegralLiteral{

    public static void main(String[] args)

    {

        byte a; // -128 to 127

        short b; // -32768 to 32767

        int c; // -2^31 to 2^31 - 1

        long d; // -2^63 to 2^63 - 1

        // a = -130; // compile-time error

        /\*  here: -130 is out of the byte range

        the out of byte range value can understand as integer by default.

        integer ==> 4-bytes

        byte ==> 1-byte

        is possible to store 4-byte value into 1-byte? No.

        \*/

        a = 0B10101;

        System.out.println(a);

        b = 0327;

        System.out.println(b);

        c = 123l;

        System.out.println(c);

        d = 127l;

        System.out.println(d);

    }

}

Note:

=====

-> Any integral literal can able to print by println() or print() or printf() in decimal by default.

Q-1: What happened if the short range value can assign to byte variable?

Ans: Compile time error

Here: short is the 2-byte range value whereas byte is 1-byte range value.

Storing of 2-byte range value into 1-byte range value is not possible. That is why we can get "compile-time error".

Q-2: Is it possible to assign an integer value into long?

Ans: Yes

2) Floating-point Literals

----------------------------

-> there are two ways to define floating-point literals:

1) using decimal point

2) using scientific format/exponential format

-> The float literal can always be suffixed with 'f' or 'F'.

class FloatLiterals{

public static void main(String[] args)

{

float f;

double d;

f = 1203.234f;

System.out.println(f);

f = 1e5F; // 1 X 10^5

System.out.println(f);

d = 123.094;

System.out.println(d);

d = 1.3e-9F; // 1.3 X 10^-9

System.out.println(d);

}

}

Note:

=====

No binary literal, octal and hexadecimal literal as float or double value.

Q-1: What happened that I have assigned an integer value to the float variable?

Ans: Possible to assign an integer range value to the float variable.

class FloatLiterals{

public static void main(String[] args)

{

float f;

f = 1023;

System.out.println(f);

}

}

Q-2: What happened that I have assigned a float value to the integer variable?

Ans: Compile-time Error

byte << short << int << long << float << double

3) Character Literal

----------------------

-> the character can always define with single quote.

That single quote allowed to define with single letter:

letter can be:

either: alphabet or digit or special characters.

Ex: 'a', '9', '#' ==> valid characters

'abcd' ==> invalid

-> To the character variable, we can also allow to assign integer range value also.

-> Java programming language is Uni-code based language.

-> 2-bytes ==> 65536 values

we can allow to assign the values from 0 to 65535 to the character variable.

0 to 9 ==> 48 to 56

A to Z ==> 65 to 90

a to z ==> 97 to 122

class CharacterLiteral{

public static void main(String[] args)

{

char ch;

ch = 'x';

System.out.println(ch);

ch = 1001;

System.out.println(ch);

ch = 65535;

System.out.println(ch);

}

}