**Java Documentation:**

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

-> Comments within the program or above to the program or below to the program are called as "Documentation".

-> There are two ways to write comments in java program:

 1) Single line commenting

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

 -> Single line comments can start and end within the same line

 -> Single line comments can always start with '//'

 2) Multi line commenting

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

 -> When we want to write comments into more than one lines called as "Multi-line comments".

 -> Multi-line comments are always in between /\* and \*/.

-> Comments in any programming language never be execute.

-> Comments can always increase the readability of the program.

// WRITE A JAVA PROGRAM TO FIND THE SUM OF TWO NUMBERS.

/\*

 HERE:

 THE TWO NUMBERS STORE INTO TWO VARIABLES

 PERFORM THE SUM ON THOSE TWO NUMBERS

 STORE THAT RESULT INTO NEW VARIABLE

 PRINT THAT RESULT.

\*/

class SumOfTwoNumbers{

 public static void main(String[] args)

 {

 int number1 = 1023;

 int number2 = 3021;

 int sum = number1 + number2;

 System.out.println("The sum of the given two numbers = "+sum);

 }

}

**Dynamic Program:**

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

-> When the program provide the output based on the input which we can process at the time of execution is called as "Dynamic program".

-> To write the Dynamic program, we need:

 1) Memory Statement

 2) Input Statement

 3) Output Statement

**1) Memory Statement**

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

-> Memory statement describe two things:

 i) How much memory need to create

 ii) What is the range of value can store into crated memory.

Ex:

Ex: int number1 = 1023;

here:

 int ==> datatype

 number1 ==> name of the data ==> Identifier

 1023 ==> value ==> literal

-> To define the memory statement, we need three types of tokens:

 1) Datatypes

 2) Identifier

 3) Literals

**Datatypes:**

**=======**

-> Datatypes can be used to describe two things:

 1) Memory of the data

 2) Range of the data

-> Two types of datatypes:

 1) Primitive Datatypes

 2) Non-primitive Datatypes/Reference Datatypes

**1) Primitive Datatypes**

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

-> Any basic or fundamental datatype is called as "Primitive datatype".

-> In Java, there are total 8 primitive datatypes:

 1) Byte datatype

 2) Short datatype

 3) Integer datatype

 4) Long datatype

 5) Float datatype

 6) Double datatype

 7) Character datatype

 8) Boolean datatype

-> To represent these 8-primitive datatypes we have 8 keywords.

**Keywords:**

**=======**



-> Keywords are "pre-defined words" in java.

-> Also called as "Reserved words".

-> Every keyword has the definite meaning can be used to perform the specific task/functionality.

-> According to the Java-8 standards, we have total of 54-keywords.

-> The keywords for Primitive datatype representation are:

 Byte datatype ==> byte

 Short datatype ==> short

 Integer datatype ==> int

 Long datatype ==> long

 Float datatype ==> float

 Double datatype ==> double

 Character datatype ==> char

 Boolean datatype ==> boolean

-> Memory and range of the datatype:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Datatype** | **Keyword** | **Default value** | **Memory** | **Range** |
| Byte type | Byte | 0 | 1 byte | -128 to 127 |
| Short type | Short | 0 | 2 bytes | -32768 to 32767 |
| Integer type | Int | 0 | 4 bytes | -2^31 to 2^31 – 1 |
| Long type | Long | 0 | 8 bytes | -2^63 to 2^63 – 1 |
| Float type | Float | 0.0 f | 4 bytes | 1.8X10^-38 to 3.4X10^38  |
| Double type | Double | 0.0 | 8 bytes | 2.3X10^-308 to 1.8X10^308 |
| Character type | Char | \u0000 | 2 bytes | \u0000 to \uFFFF |
| Boolean type | Boolean | False | 1 bit | true, false |

Note:

=====

Range of data for any type:

 -2^(n-1) to 2^(n-1) - 1

Here:

 n ==> total number of bits

-> Syntax for any data definition using primitive datatypes is:

 datatype name-of-the-data = value;

Ex: byte b = -128;

byte a = 127;

byte c = 0;

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

-> Float data can take maximum of 6 decimal places before and/or after the decimal point.

Ex: 123456.123456f

-> Float value must be suffixed with 'f' or 'F'

-> Double data can take maximum of 12 decimal places before and/or after the decimal point.

Ex: 123456789012.12345;

-> Character can be defined with single quotes.

Within the single quotes, we can write only the single character.

Ex: 'a', 'Q' valid etc.

'abcd' ==> invalid

-> In java, every character can be defined with a unique value called as "Unicode".

-> ASCII can have only 256 values to represent ( 0 to 255)

Whereas Uni codes have total 65536 (== 2^15) values. (0 to 65535)

Here:

 0 ==> \u0000

 65535 ==> \uffff

-> Characters can allowed to define with:

 alphabets:

 a to z ==> 97 to 122

 A to z ==> 65 to 90

 Digits: 0 to 9 ==> 48 to 56