Variables, Data Types, Comments, Escape Sequences

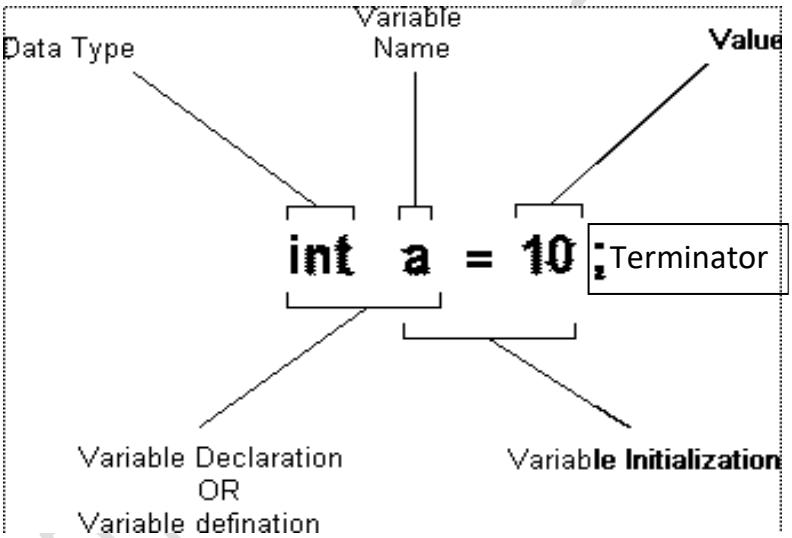
Variables

Variables are containers for storing data values.

Variables are declared by first writing

data types followed by a variable name,

e.g. int a=10;



Here

int is data type,

a is variable name

and after the equals to sign (=) is the value in it 10 the value is always followed by a terminator.

Variable Names

Variable names will always start with an alphabet.

Variable names can contain numbers (1,2,45,66) and underscores (_) but no other special characters (!@#$%^&*). A variable name cannot be used for multiple declarations.

In Java, there are different types of variables, for example:

String - stores text, such as "Hello". String values are surrounded by double quotes

int - stores integers (whole numbers), without decimals, such as 123 or -123

float - stores floating point numbers, with decimals, such as 19.99 or -19.99

char - stores single characters, such as 'a' or 'B'. Char values are surrounded by single quotes

boolean - stores values with two states: true or false

Example of variables with data types

```
String myText = "Hello";        // String
int myNum = 5;                  // Integer (whole number)
float myFloatNum = 5.99f;       // Floating point number
char myLetter = 'D';            // Character
boolean myBool = true;          // Boolean
```

Data types are divided into two groups:

Primitive data types - includes byte, short, int, long, float, double, boolean and char

Non-primitive data types - such as String, Arrays and Classes (you will learn more about these in a later chapter)

Primitive Data Types

A primitive data type specifies the size and type of variable values, and it has no additional methods.

There are eight primitive data types in Java:

S.No	Data Type	Size	Description
1	**byte**	**1 byte**	Stores whole numbers from -128 to 127
2	**short**	**2 bytes**	Stores whole numbers from -32,768 to 32,767
3	**int**	**4 bytes**	Stores whole numbers from -2,147,483,648 to 2,147,483,647
4	**long**	**8 bytes**	Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
5	**float**	**4 bytes**	Stores fractional numbers. Sufficient for storing 6 to 7 decimal digits
6	**double**	**8 bytes**	Stores fractional numbers. Sufficient for storing 15 decimal digits
7	**boolean**	**1 bit**	Stores true or false values
8	**char**	**2 bytes**	Stores a single character/letter or ASCII values

Final Variables

If you don't want others (or yourself) to overwrite existing values, use the final keyword (this will declare the variable as "final" or "constant", which means unchangeable and read-only):

Example

final int myNum = 15;

myNum = 20; **// will generate an error: cannot assign a value to a final variable**

Java Comments

Comments can be used to explain Java code, and to make it more readable. It can also be used to prevent execution when testing alternative code.

Single-line Comments

Single-line comments start with two forward slashes (//).

Any text between // and the end of the line is ignored by Java (will not be executed).

This example uses a single-line comment before a line of code:

```
// This is a comment
System.out.println("Hello World");
```

Another example uses a single-line comment at the end of a line of code:

```
System.out.println("Hello World"); // This is a comment
```

Java Multi-line Comments

Multi-line comments start with /* and ends with */.

Any text between /* and */ will be ignored by Java.

This example uses a multi-line comment (a comment block) to explain the code:

```
/* The code below will print the words Hello World
to the screen, and it is amazing */
System.out.println("Hello World");
```

Escape Sequences

Escape Sequences are used to adjust spacing between lines or characters or the characters themselves.

No.	Syntax	Application	Example
1	**\n**	New Line	System.out.println(“Text1\nText1”);
2	**\t**	Tab eight spaces to right	System.out.println(“Text2\tText2”);
3	**\b**	Back space One space back	System.out.println(“Text3\bText3”);
4	**\r**	Carriage return Start of same line	System.out.println(“Text4\rText4”);
5	**\’**	Printing single quote	System.out.println(“Text5\’ Text5”);
6	**\”**	Printing double quotes	System.out.println(“Text6\”Text6”);
7	****	Printing back space	System.out.println(“Text7\\Text7”);

Practice Tasks:

1. Add the correct Data Types for the following variables:

   _____ myNum = 9;

   _____ myFloatNum = 8.99f;

   _____ myLetter = 'A';

   _____ myBool = false;

   _____ myText = "Hello World";

2. Identify and fill the appropriate Symbols for following Comments
   _____ This is a single-line comment
   _____ This is a multi-line comment _____

3. Create a program that performs the following tasks:
   - Declare a variable of type int and assign it a value.
   - Declare a variable of type double and assign it the value of the int variable.
   - Print both variables.

4. Create a program that uses escape sequences to format the output:
   - Print a message on two separate lines using the newline character (\n).
   - Print a sentence with a tab space (\t) between words.
   - Print a statement with a backspace character (\b) to erase a character.
   - Print a sentence with a carriage return (\r) to overwrite part of the line.
   - Print a string containing single (') and double (") quotes.
   - Print a backslash character (\\).