**Unit-2 Notes**

**1. Data Types**

Data types define the type of data a variable can hold and the operations that can be performed on it. Java has two primary categories: primitive and non-primitive (reference) data types.

**1.1 Primitive Data Types**

* **byte:** Represents an 8-bit signed integer.
* **short:** Represents a 16-bit signed integer.
* **int:** Represents a 32-bit signed integer.
* **long:** Represents a 64-bit signed integer.
* **float:** Represents a 32-bit single-precision floating-point number.
* **double:** Represents a 64-bit double-precision floating-point number.
* **char:** Represents a single Unicode character.
* **boolean:** Represents a boolean value (true or false).

**Example:**

Java

byte age = 25;

short salary = 30000;

int population = 1000000;

long distance = 1234567890123L;

float pi = 3.14159f;

double gravity = 9.80665;

char gender = 'M';

boolean isAdult = true;

**1.2 Non-Primitive (Reference) Data Types**

These are classes, interfaces, and arrays. They refer to objects in memory, not the actual data itself.

**2. Variables**

Variables are named memory locations used to store data values. They have a data type, name, and value.

**2.1 Declaration**

Java

data\_type variable\_name;

**Example:**

Java

int age;

String name;

**2.2 Initialization**

Assigning a value to a variable.

Java

data\_type variable\_name = value;

**Example:**

Java

int age = 25;

String name = "Alice";

**2.3 Dynamic Initialization**

Assigning a value to a variable at runtime.

Java

data\_type variable\_name;

// ...

variable\_name = value;

**Example:**

Java

int num;

Scanner scanner = new Scanner(System.in);

num = scanner.nextInt();

**2.4 Scope and Lifetime**

* **Scope:** The region of code where a variable is accessible.
  + **Local variables:** Declared within a method or block.
  + **Instance variables:** Declared within a class but outside methods.
  + **Class variables:** Declared with the static keyword.
* **Lifetime:** The duration for which a variable exists.
  + Local variables exist within the method or block.
  + Instance variables exist as long as the object exists.
  + Class variables exist throughout the program's execution.

**3. Type Conversion and Casting**

* **Widening conversion:** Automatic conversion from smaller to larger data types (e.g., int to double).
* **Narrowing conversion:** Explicit conversion from larger to smaller data types (e.g., double to int).

**Example:**

Java

int num = 10;

double d = num; // Widening

int i = (int) d; // Narrowing (potential loss of precision)

**4. Operators**

Operators are symbols used to perform operations on variables and values.

**4.1 Arithmetic Operators:** +, -, \*, /, %, ++, --

**4.2 Relational Operators:** ==, !=, >, <, >=, <=

**4.3 Logical Operators:** &&, ||, !

**4.4 Bitwise Operators:** &, |, ^, ~, <<, >>, >>>

**4.5 Assignment Operators:** =, +=, -=, \*=, /=, %=

**4.6 Ternary Operator:** ? :

**Example:**

Java

int a = 10, b = 5;

int sum = a + b;

boolean isGreater = a > b;

int max = (a > b) ? a : b;