OPERATORS

- In java, Operators are special symbols that perform operations on variables and values. Java provides several types of operators.
- Unary Operators, Binary Operators and Ternary Operators

UNARY OPERATOR

- Unary operator is used to operate on single variable.
  - + (Unary plus) → +a
  - - (Unary minus) →-a
  - ++ (Increment) → a++ (Post - Increment) or ++a (Pre – Increment)
  - -- (Decrement) → a-- (Post – decrement) or --a (pre – decrement)

OPERAND

- The values or variables on which the operator acts.
- Example : a + b ;
- Here a and b are operands and + is and operator.

BINARY OPERATORS

- A Binary operator is an operator that operates on two operands.

Types of Binary operators in Java

1. Arithmetic Operators
2. Relational Operators
3. Logical Operators
4. Assignment Operators

ARITHMETIC OPERATORS

- Arithmetic Operators are used for mathematical operations.
  - +(Addition) → a + b
  - - (Subtraction) → a - b
  - * (Multiplication) → a * b
  - / (Division) → a / b (Gives Quotient)
  - % (Modulo) → a % b (Gives the reminder)

RELATIONAL OPERATORS

- Relational Operators are used for comparing values.
  - == (Equal to) → a == b
  - != (Not Equal to) → a != b

- > (Greater than) → a > b
- < (Less than) → a < b
- >= (Greater than or equal to) → a >= b
- <= (Less than or equal to) → a <= b

LOGICAL OPERATORS

- Logical Operators are used for logical operations
  - && (Logical AND) → a && b
  - || (Logical OR) → a || b
  - ! (Logical NOT) → !a

ASSIGNMENT OPERATORS

- Assignment Operators are used to assign values
  - += (add and assign) → a += 5 (same as a = a + 5)
  - -= (Subtract and assign) → a -= 5
  - *= (Multiply and assign) → a *= 5
  - /= (Divide and assign) → a /= 5
  - %= (Modulus and assign) → a %= 5

TERNARY OPERATOR

- (Condition) ? Value_if_true : Value_if_false ;
- Example : int result = (a > b) ? a : b ;

(if a > b , result = a , otherwise result = b)

CASTING

- Casting in java is the process of converting one data type into another.
- Casting will be done on two data types.
    - Premitive data types
    - Non – Premitive data types
- For primitive data types there are two types of casting
    - Implicit Casting (Widening) – Automatic conversion
    - Explicit Casting (Narrowing) – Manual conversion

Implicit Casting (Widening)

- Happens automatically when a smaller data type is converted into a larger data type.
- No data loss occur.
- Example – byte → short →int → long → float →double
- Example int num = 10;

  double doublenum = num;// implicit casting from int to double.

  S.o.p (“Integer : “ + num);

S.o.p (“Double : “ + doublenum);

EXPLICIT CASTING (Narrowing)

- Happens when a larger data type is converted into a smaller data type.
- Must be done manually using type casting.
- Data loss may occur.
- Example : double →float→long→int→short→byte
- Example : double doubleNum = 10.99;

  int intNum = (int) doubleNum; // Explicit casting from double to int

  S.o.p("Double: " + doubleNum); // Output: 10.99

  S.o.p("Integer: " + intNum); // Output: 10 (decimal part lost)