Java Identifiers and Data Types

1. Identifiers in Java

An identifier is the name used to identify a variable, class, method, or any user-defined element in Java.

Why Identifiers are Needed:

Identifiers are used to give unique names to variables, methods, classes, etc., to refer to them in the program.

Rules for Valid Identifiers:

- Must not be a Java keyword.
- Must not contain spaces.
- Must begin with a letter (A-Z or a-z), underscore (_) or a dollar sign ($).
- Cannot start with a digit (0-9).
- Should not exceed 32 characters (recommended; Java allows more, but long names are discouraged).
- Can contain letters, digits, underscores, and dollar signs.

Examples of Valid Identifiers:

- myVariable
- _name
- $amount
- student1

Examples of Invalid Identifiers:

- 1student (starts with digit)

- class (keyword)

- my variable (contains space)

- @name (invalid symbol)

2. Java Data Types

Java has two categories of data types: Primitive and Non-Primitive.

Primitive Data Types:

- int: Stores whole numbers from -2,147,483,648 to 2,147,483,647 (4 bytes)

  Example: int a = sc.nextInt();

- long: Stores larger whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 (8 bytes)

  Example: long b = sc.nextLong();

- float: Stores decimal numbers with ~7 decimal digits of precision (4 bytes)

  Example: float c = sc.nextFloat(); // 12.34567f

- double: Stores decimal numbers with ~15 decimal digits of precision (8 bytes)

  Example: double d = sc.nextDouble(); // 12.3456789012345

- char: Stores a single 16-bit Unicode character

  Example: char ch = 'A';

- boolean: Stores true or false

  Example: boolean flag = true;

Non-Primitive Data Type:

- String: Used to store a sequence of characters

  Example: String name = sc.nextLine(); // Can include spaces

3. Class and Object in Java

A class is a blueprint for objects. An object is an instance of a class.

Example:

```
class Student {
    int id;
    String name;

    void display() {
        System.out.println(id + " " + name);
    }

    public static void main(String[] args) {
        Student s1 = new Student(); // creating object
        s1.id = 101;
        s1.name = "John";
        s1.display();
    }
}
```

In the above code:

- 'Student' is the class.
- 's1' is the object of class Student.
- 'display' is a method in the class.