
Java Programming
Basics

JAVA BASICS

Part II

Jyoti Lakhani

VARIABLES

Variable

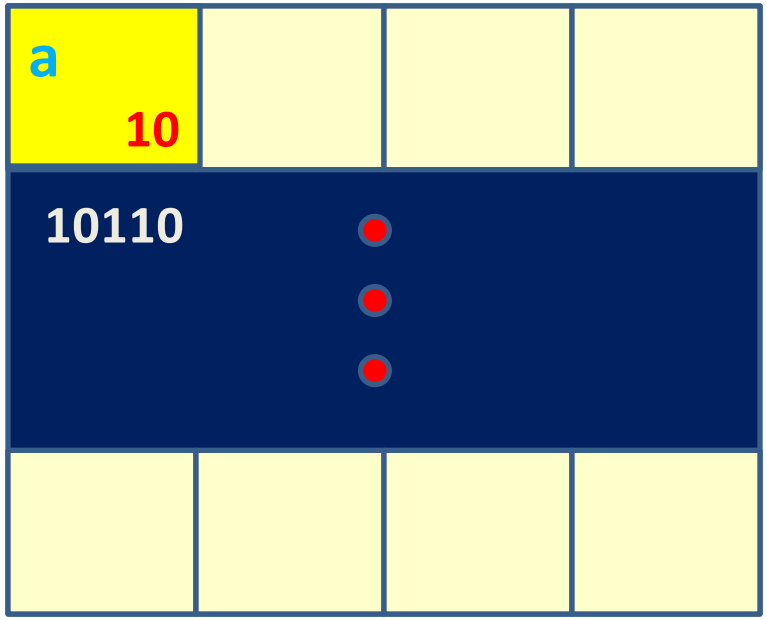
Variables are nothing but reserved memory locations to store values

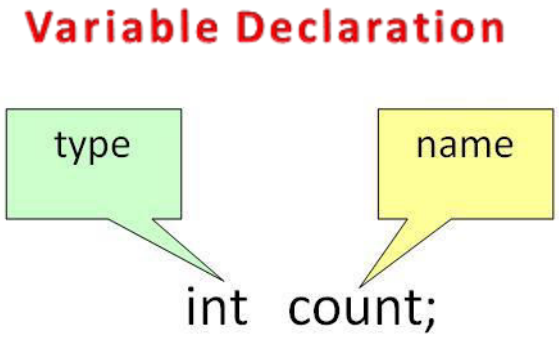
A Variable has –

Memory Address

Name

Value



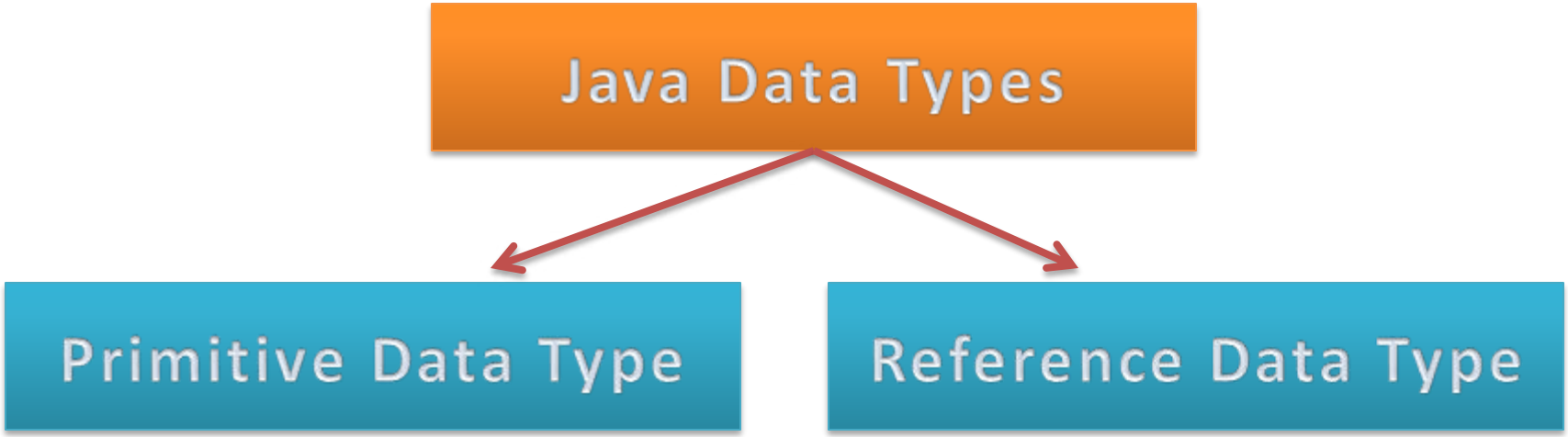


int a=10;

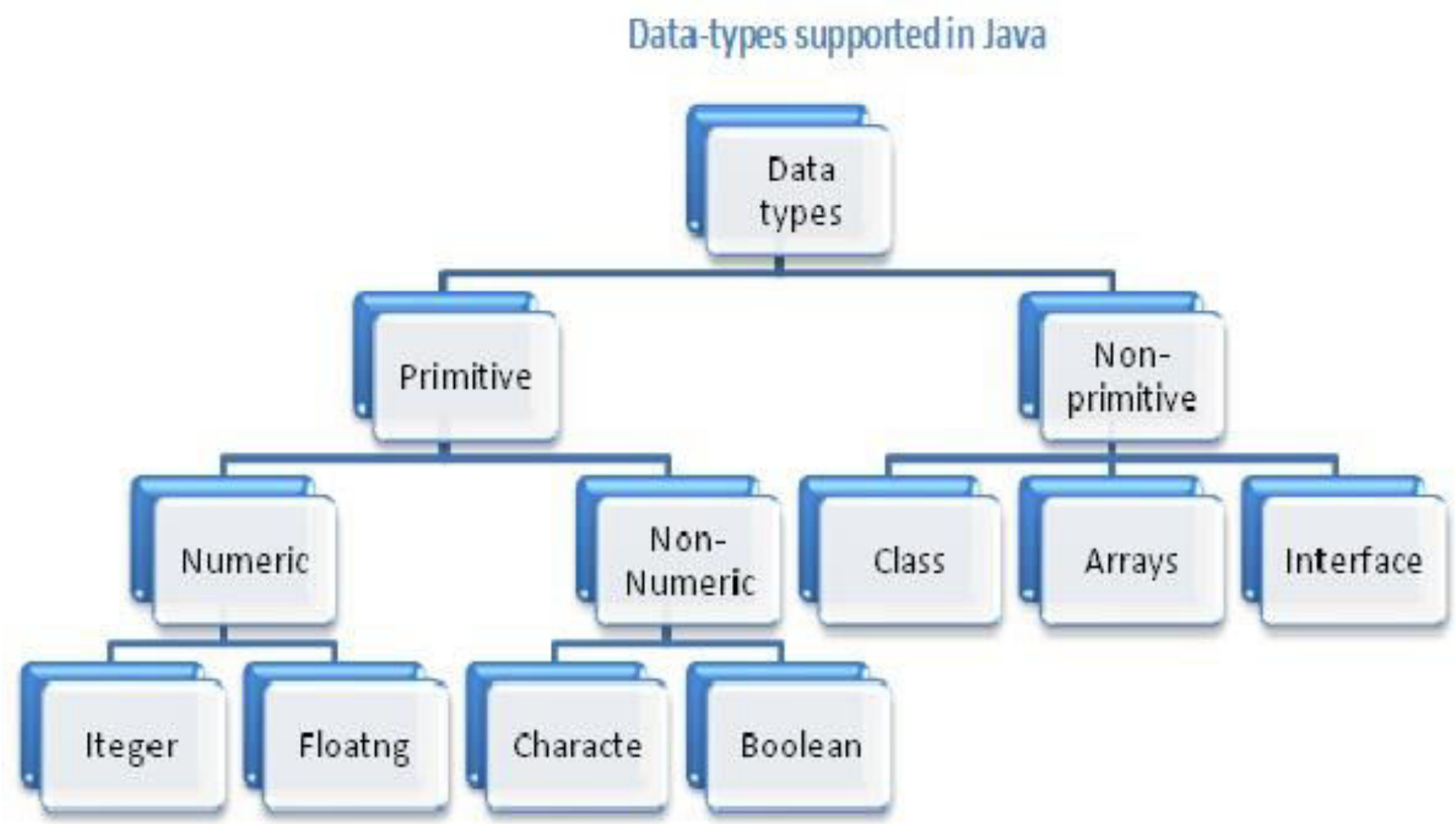
Java Data Types- Classification

There are two data types available in Java:

- **Primitive Data Types**
- **Reference/Non Primitive/ Derived Data Types**



Java Data Types



Primitive DataTypes

There are eight primitive data types supported by Java.

Primitive data types are predefined by the language and named by a keyword

Primitive DataType- Byte

- Byte data type is an 8-bit signed integer
- Minimum value is -128 (-2^7)
- Maximum value is 127 (inclusive)(2^7 -1)
- Default value is 0
- Byte data type is used to save space in large arrays, mainly in place of integers, since a byte is four times smaller than an int
- Example: byte a = 100 , byte b = -50

Primitive Data Type- Short

- Short data type is a 16-bit signed integer
- Minimum value is -32,768 (-2^15)
- Maximum value is 32,767 (inclusive) (2^15 -1)
- Short data type can also be used to save memory as byte data type
- A short is 2 times smaller than an int
- Default value is 0
- Example: short s = 10000, short r = -20000

Primitive Data Type- int

- Int data type is a 32-bit signed integer
- Minimum value is - 2,147,483,648.(-2^31)
- Maximum value is 2,147,483,647(inclusive).(2^31 -1)
- Int is generally used as the default data type for integral values unless there is a concern about memory
- The default value is 0
- Example: int a = 100000, int b = -200000

Primitive Data Type- long

- Long data type is a 64-bit signed integer
- Minimum value is -9,223,372,036,854,775,808.(-2^63)
- Maximum value is 9,223,372,036,854,775,807 (inclusive) (2^63 -1)
- This type is used when a wider range than int is needed
- Default value is 0L.
- Example: long a = 100000L, int b = -200000L

Primitive Data Type- float

- Float data type is a single-precision 32-bit IEEE 754 floating point
- Float is mainly used to save memory in large arrays of floating point numbers
- Default value is 0.0f
- Float data type is never used for precise values such as currency
- Example: float f1 = 234.5f

Primitive Data Type- double

- double data type is a double-precision 64-bit IEEE 754 floating point
- This data type is generally used as the default data type for decimal values, generally the default choice
- Double data type should never be used for precise values such as currency
- Default value is 0.0d
- Example: double d1 = 123.4

Primitive Data Type- boolean

- boolean data type represents one bit of information
- There are only two possible values: true and false
- This data type is used for simple flags that track true/false conditions
- Default value is false
- Example: boolean one = true

Primitive Data Type- char

- char data type is a single 16-bit Unicode character
- Minimum value is '\u0000' (or 0)
- Maximum value is '\uffff' (or 65,535 inclusive)
- Char data type is used to store any character
- Example: char letterA ='A'

Reference Data Types

- Reference variables are created using defined constructors of the classes
- They are used to access objects. These variables are declared to be of a specific type that cannot be changed. For example, Employee, Puppy etc.
- Class objects, and various type of array variables come under reference data type
- Default value of any reference variable is null
- A reference variable can be used to refer to any object of the declared type or any compatible type
- Example: Animal animal = new Animal("giraffe");

Java Variables

Java – Variable Types

In Java, all variables must be declared before they can be used. The basic form of a variable declaration is shown here:

```
type identifier [ = value][, identifier [= value] ...] ;
```

The *type* is one of Java's datatypes. The *identifier* is the name of the variable. To declare more than one variable of the specified type, use a comma-separated list.

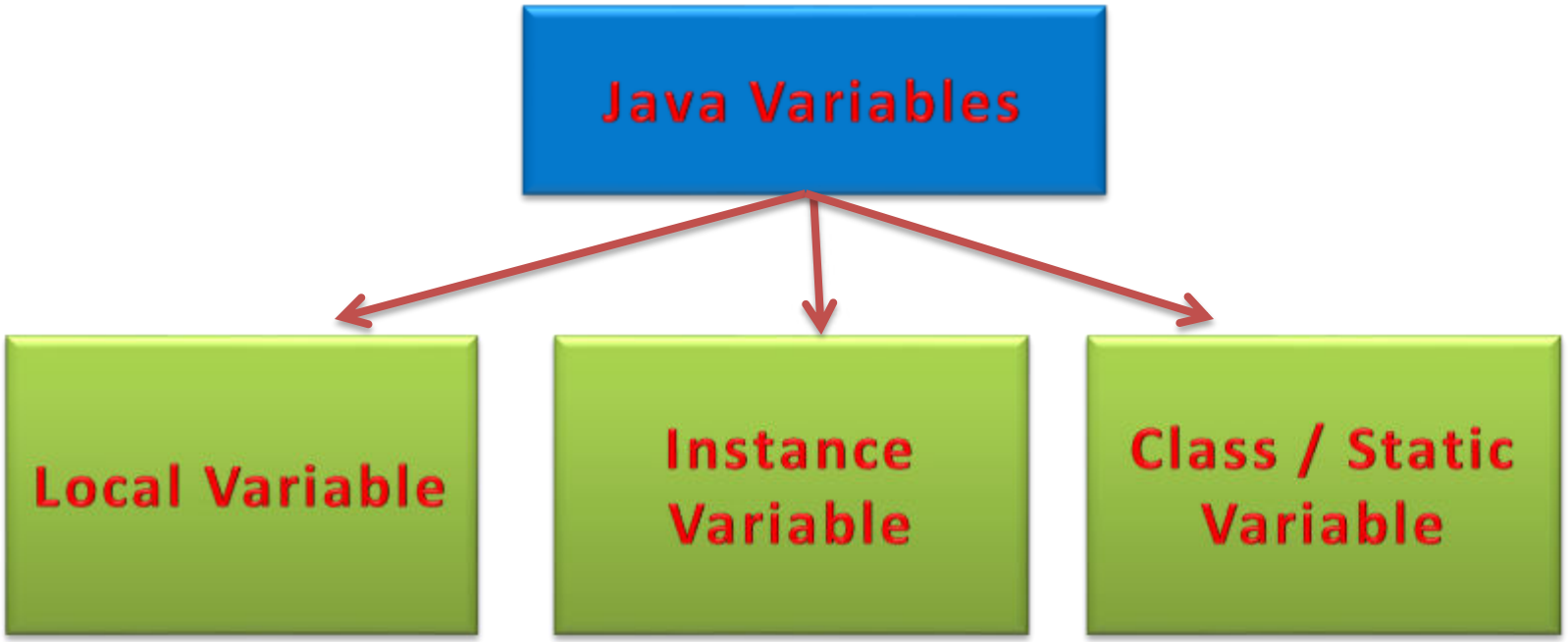
Here are several examples of variable declarations of various types. Note that some include an initialization

```
int a, b, c;         // declares three ints, a, b, and c

int d = 3, e, f = 5; // declares three more ints, initializing d and f

byte z = 22;         // initializes z

double pi = 3.14159; // declares an approximation of pi

char x = 'x';        // the variable x has the value 'x'
```

Java – Variable Types

There are three kinds of variables in Java:

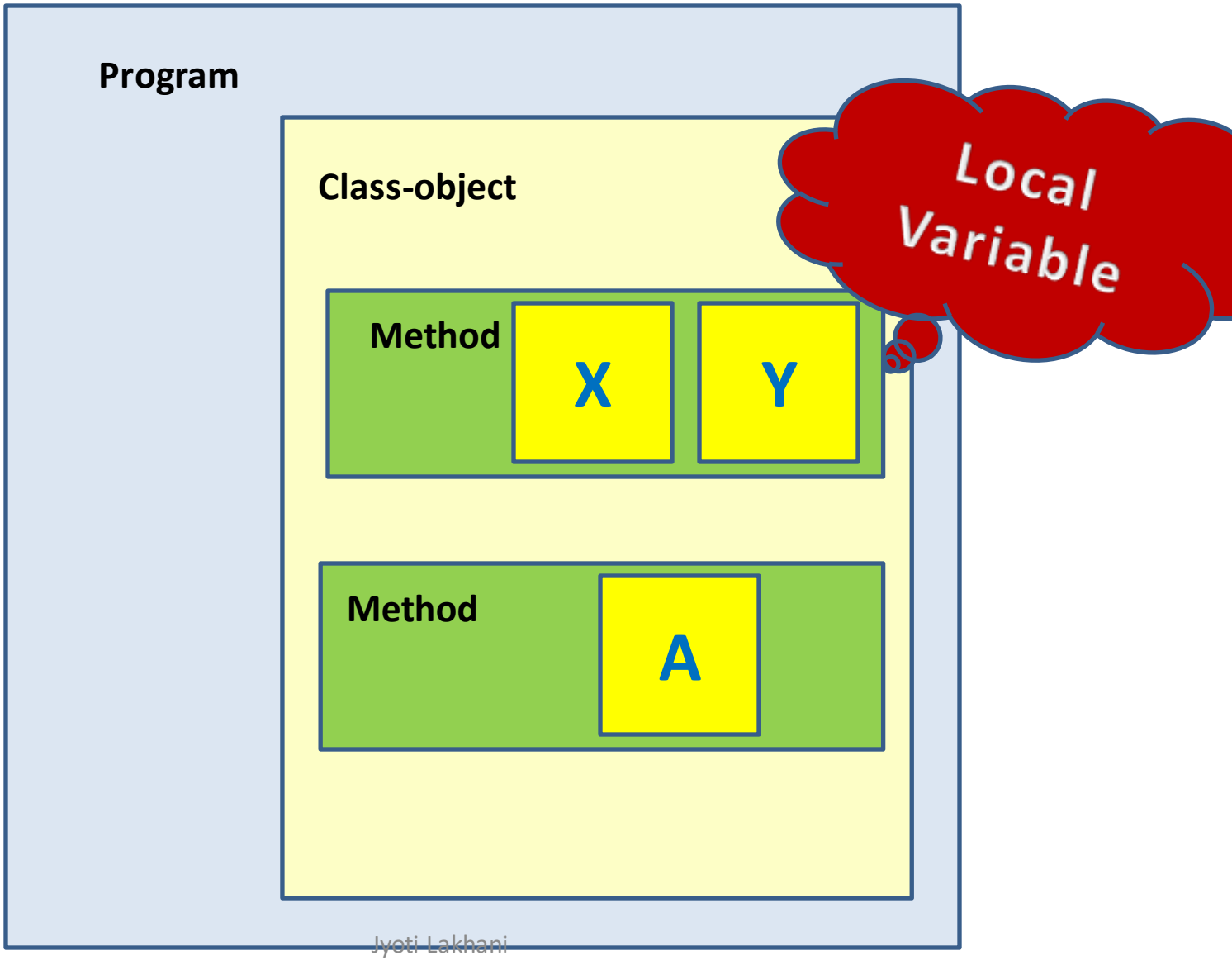
- Local variables
- Instance variables
- Class/static variables



Java – Local Variables

- Local variables are declared in methods, constructors, or blocks
- Local variables are created when the method, constructor or block is entered and the variable will be destroyed once it exits the method, constructor or block *
- Access modifiers cannot be used for local variables
- Local variables are visible only within the declared method, constructor or block
- Local variables are implemented as stack internally *
- There is no default value for local variables so local variables should be declared and an initial value should be assigned before the first use.

Java- Local Variable



Java – Instance Variables

- Instance variables are declared in a class, but outside a method, constructor or any block.

- When a space is allocated for an object in the heap, a slot for each instance variable value is created.

- Instance variables are created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed.

- Instance variables hold values that must be referenced by more than one method, constructor or block, or essential parts of an object's state that must be present throughout the class.

Java – Instance Variables

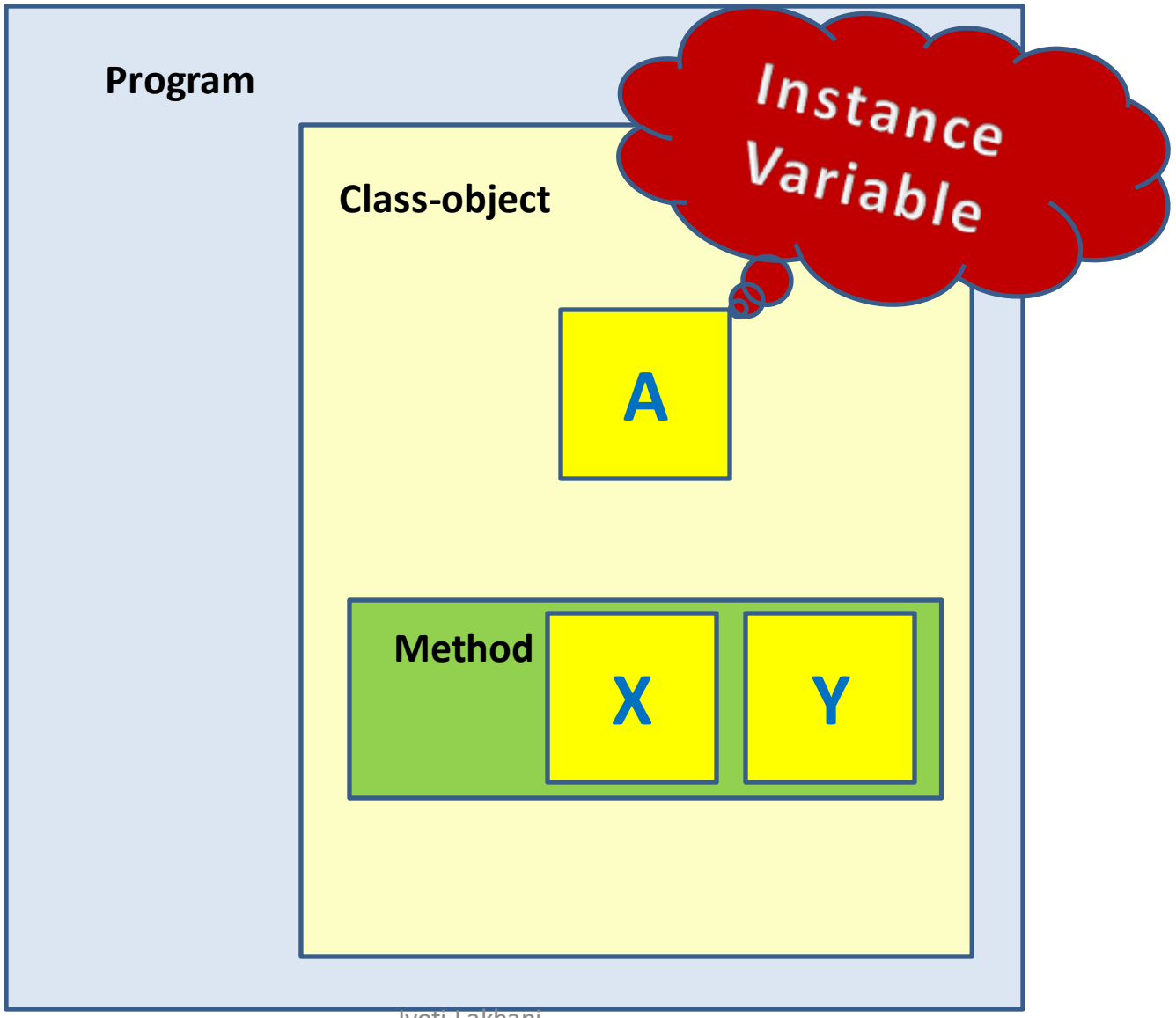
- Instance variables can be declared in class level before or after use.
- Access modifiers can be given for instance variables
- The instance variables are visible for all methods, constructors and block in the class. Normally, it is recommended to make these variables private (access level). However visibility for subclasses can be given for these variables with the use of access modifiers.
- Instance variables have default values. For numbers the default value is 0, for Booleans it is false and for object references it is null. Values can be assigned during the declaration or within the constructor.

Java – Instance Variables

- Instance variables can be accessed directly by calling the variable name inside the class. However within static methods and different class (when instance variables are given accessibility) should be called using the fully qualified name .

ObjectReference.VariableName.

Java- Instance Variable



Java – Class/ Static Variables

- Class variables also known as static variables are declared with the *static* keyword in a class, but outside a method, constructor or a block.
- There would only be one copy of each class variable per class, regardless of how many objects are created from it.
- Static variables are rarely used other than being declared as constants. Constants are variables that are declared as public/private, final and static. Constant variables never change from their initial value.
- Static variables are stored in static memory. It is rare to use static variables other than declared final and used as either public or private constants.
- Static variables are created when the program starts and destroyed when the program stops.

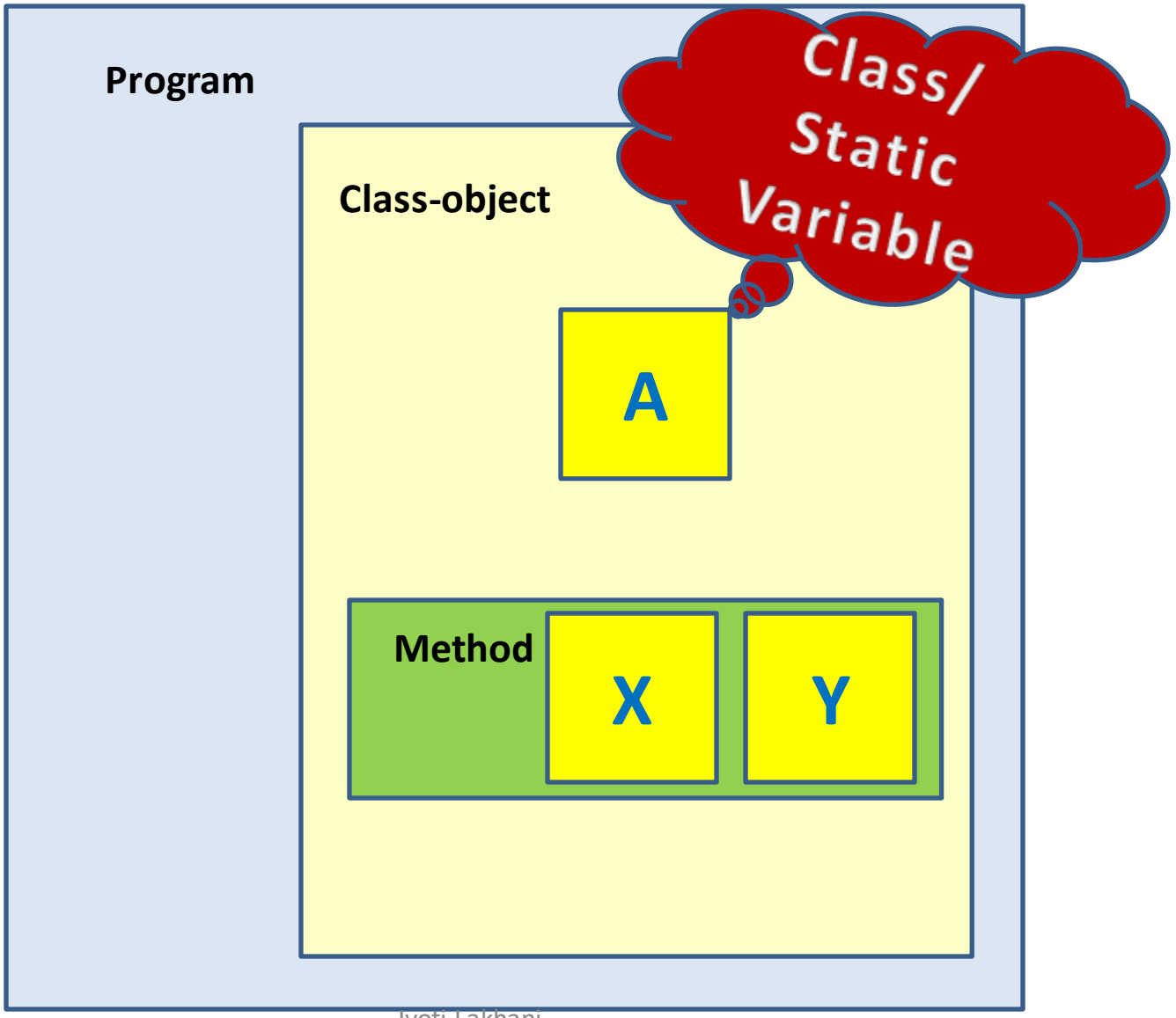
Java – Class/ Static Variables

- Visibility is similar to instance variables. However, most static variables are declared public since they must be available for users of the class.
- Default values are same as instance variables. For numbers, the default value is 0; for Booleans, it is false; and for object references, it is null. Values can be assigned during the declaration or within the constructor. Additionally values can be assigned in special static initializer blocks.
- Static variables can be accessed by calling with the class name .

ClassName.VariableName

- When declaring class variables as public static final, then variables names (constants) are all in upper case. If the static variables are not public and final the naming syntax is the same as instance and local variables.

Java- Class/Static Variable



Java – Modifires

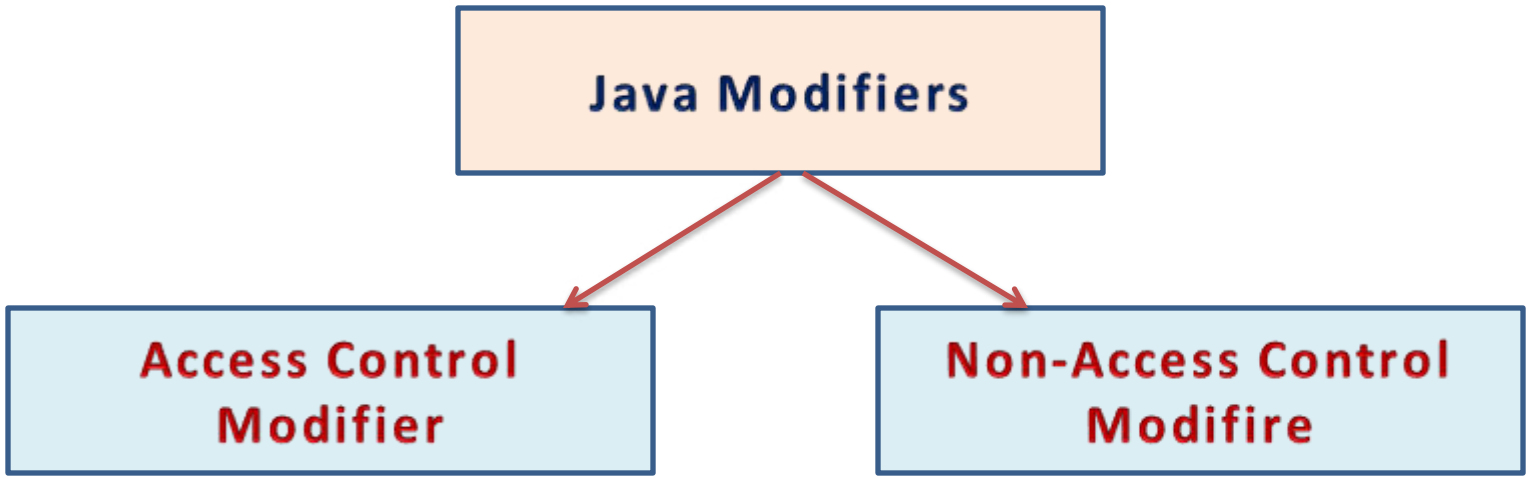
Modifiers are keywords used to change meanings of defnitions

To use a modifier, you include its keyword in the definition of a class, method, or variable

The Java language has a wide variety of modifiers, including the following:

- Java Access Modifiers
- Non Access Modifiers

Java – Modifires



Java – Access Control Modifiers

Java access modifiers are used to set access levels for classes, variables, methods and constructors.

The four access levels are:-

- Visible to the package, the default. No modifiers are needed
- Visible to the class only (private)
- Visible to the world (public)
- Visible to the package and all subclasses (protected)

Java – Non Access Modifiers

Java provides a number of non-access modifiers to achieve Special functionality

- The ***static*** *modifier for creating class methods and variables*
- The ***final*** *modifier for finalizing the implementations of classes, methods, and variables*
- The ***abstract*** *modifier for creating abstract classes and methods*
- The ***synchronized*** *and* ***volatile*** *modifiers, which are used for threads*