PSOOP(Java) – Lecture 02
Classes and Objects in Java

-Compiled by Nikahat Mulla

Agenda

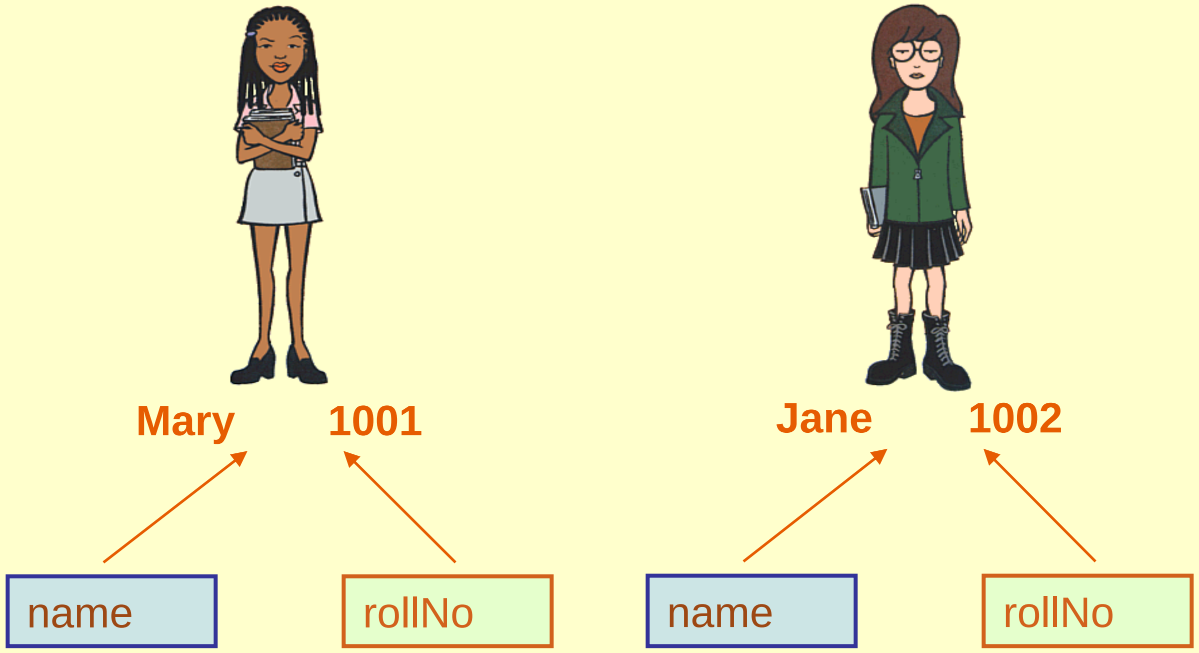
- Classes
- Objects
- Constructors
- Garbage Collection in Java
- Scope of Variables
- Command Line Arguments

Concept of Class

- A class is a description/blueprint/template of a group of objects with common properties (attributes) & behavior (operations)
  - An object is a real world entity which is an instance of a class

    e.g. Mary is an object of Student class

    Jane is an object of Student class



Constituents of a Class

```
public class Student {
private int rollNo;
private String name;


Student(){
        //initialize data members
        }
        Student(String nameParam){
                name = nameParam;
        }
        public int getrollNo (){
                return rollNo;
        }
}
```

Data Members (State)

Constructor

Method
(Behavior)

The main method may or may not be present depending on whether the class is a starter class

note

Access Modifiers – Private & Public

- Four Access Modifiers:
  - private
  - protected
  - public
  - default

Default is NOT a keyword in Java

- Data members are always kept private
  - Accessible only within the class
- The methods which expose the behavior of the object are kept public
  - However, we can have helper methods which are private
- Key features of Object Oriented Programs
  - Encapsulation (code & data bound together)
  - State (data) is hidden & Behavior (methods) is exposed to external world

CREATING OBJECTS

- The *new* operator creates an object & returns a reference to it
- Memory allocation of objects happens in the heap area
- Reference returned can be stored in reference variables

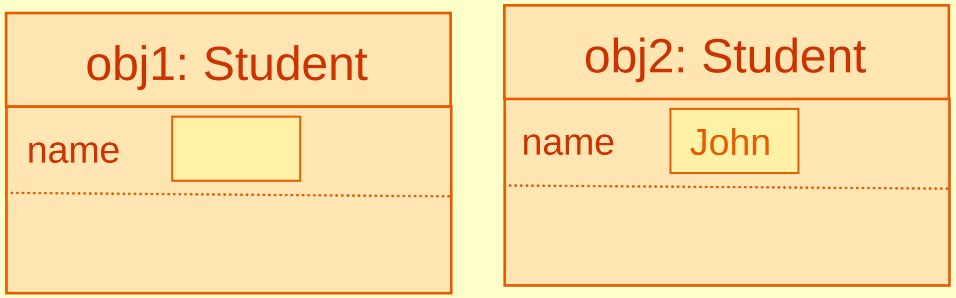
```
Student obj1;
obj1 = new Student();
```

Or

```
Student obj2 = new Student("John");
```

obj1 is a reference variable

new keyword creates an object and returns a reference to it



note

Constructors

- Special methods used to initialize a newly created object
- Called just after memory is allocated for an object
- Initialize objects to required or default values at the time of object creation
- Not mandatory to write a constructor for each class
- A constructor
  - Has the same name as that of the class
  - Doesn’t return any value, not even `void`
  - May or may not have parameters (arguments)
- If a class does not have any constructor, the default constructor is automatically added

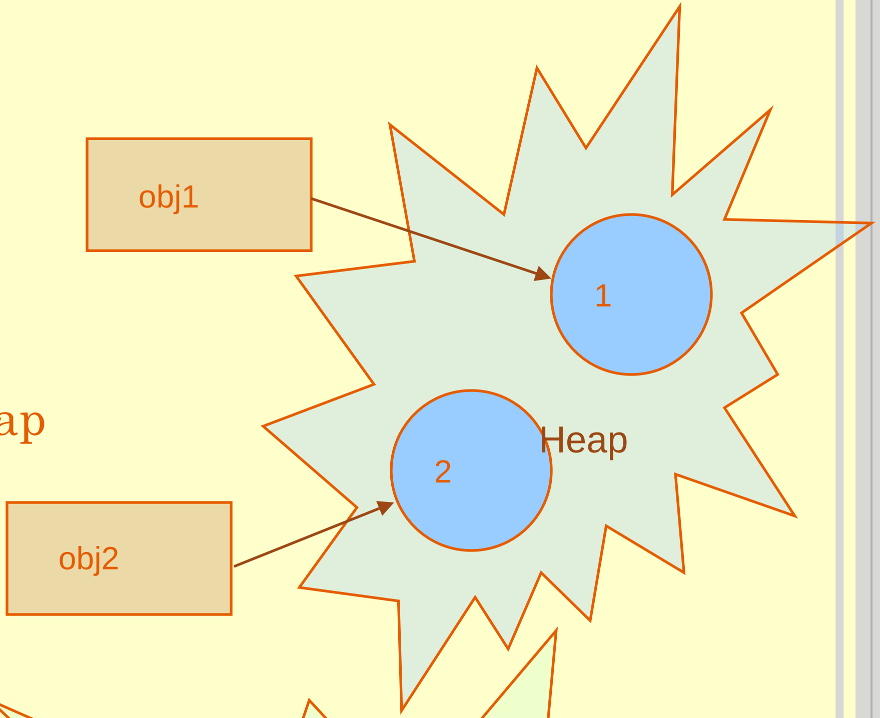
Constructors (Contd…)

- In the absence of a user defined constructor, the compiler initializes member variables to its default values
  - Numeric data types are set to 0
  - Char data types are set to null character ('\0')
  - Reference variables are set to *null*

Lifetime of Objects

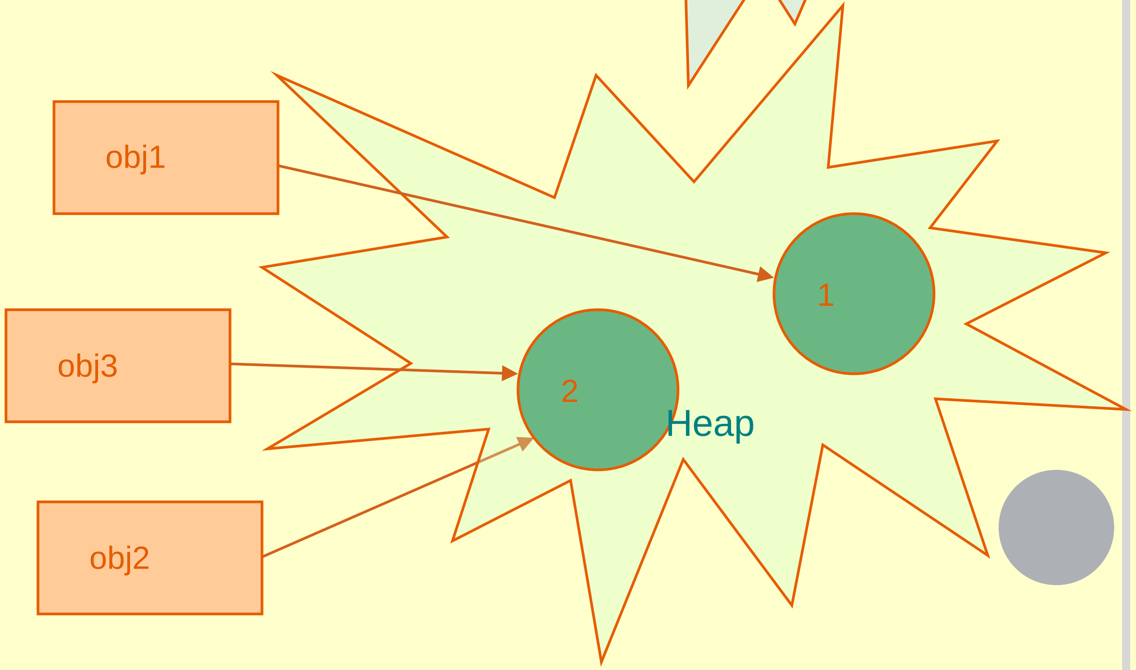
```
Student obj1 = new Student();
  Student obj2 = new Student();
```

Both Student objects now live on the heap
- → References : 2
- → Objects : 2



```
Student obj3 = obj2;
```

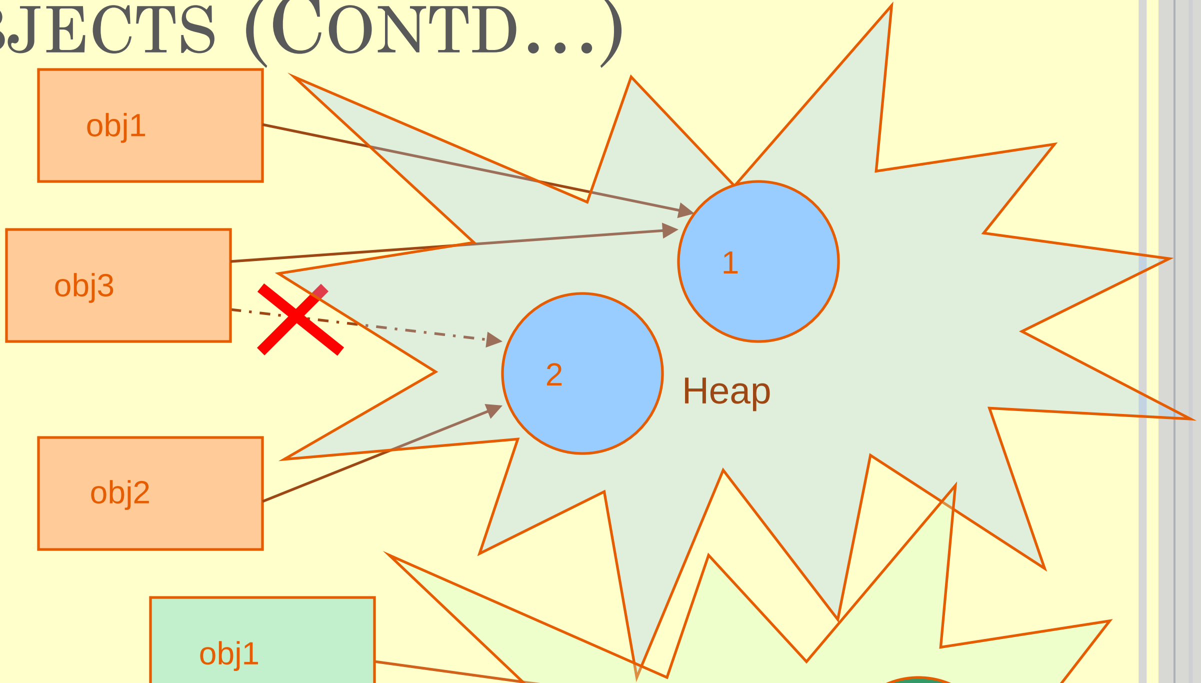
- → References : 3
- → Objects : 2



Lifetime of Objects (Contd…)

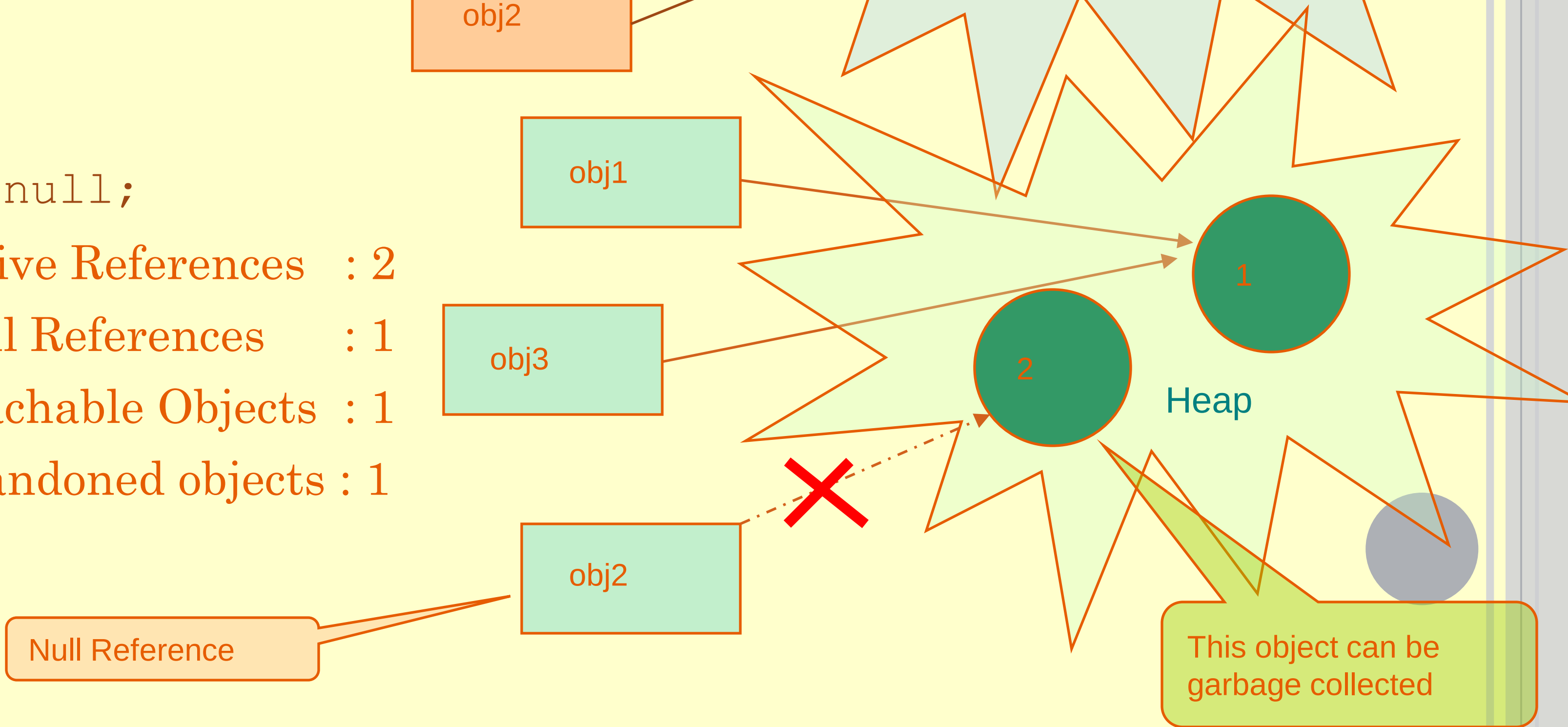
```
obj3 = obj1;
```

- → References : 3
- → Objects : 2



```
obj2 = null;
```

- → Active References : 2
- → Null References : 1
- → Reachable Objects : 1
- → Abandoned objects : 1



Garbage Collection

- In C/C++, it is the programmer's responsibility to de-allocate the dynamically allocated memory using the *free()* function

- JVM automatically de-allocates memory (Garbage Collection)

- An object which is not referred by any reference variable is removed from memory by the Garbage Collector

- Primitive types are not objects & cannot be assigned *null*

note

Scope of Variables

- Instance Variables (also called Member Variables)
  - Declared inside a class
  - Outside any method or constructor
  - Belong to the object
  - Stored in heap area with the object to which they belong to
  - Lifetime depends on the lifetime of object

- Local Variables (also called Stack Variables)
  - Declared inside a method
  - Method parameters are also local variables
  - Stored in the program stack along with method calls and live until the call ends

note

Scope of Variables (Contd…)

- If we don't initialize instance variables explicitly, they are awarded predictable *default initial values*, based only on the type of the variable

Type	Default Value
boolean	false
byte	(byte) 0
short	(short) 0
int	0
long	0L
char	\u0000
float	0.0f
double	0.0d
object reference	null

- Local variables are not initialized implicitly

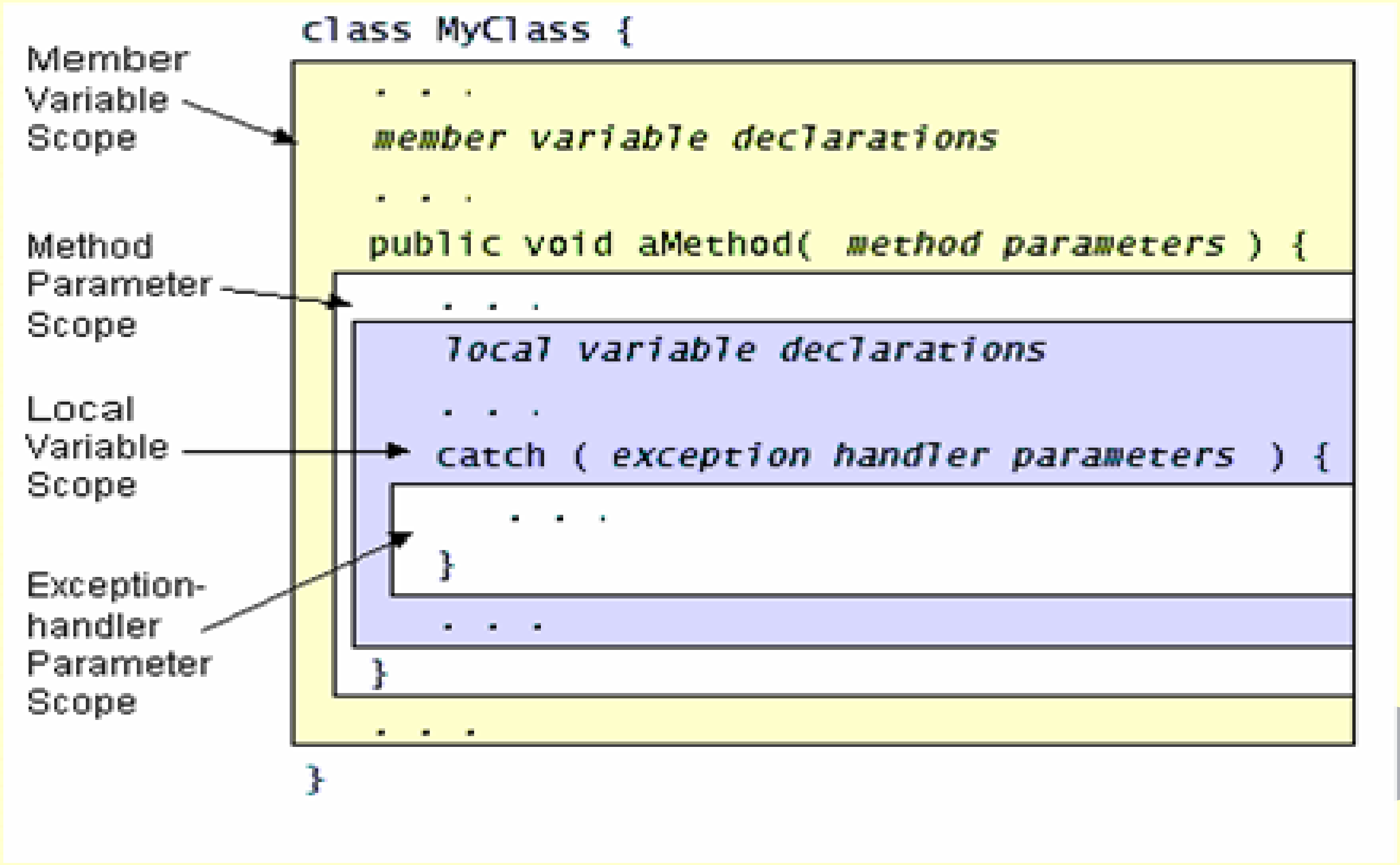
SCOPE OF VARIABLES (CONTD…)

```
class Student{
   int rollNo;
   String name;
   public void display (int z){
      int x=z+10;
     }
  }
```

rollNo and name are instance variables to be stored in the heap

z and x are local variables to be stored in the stack

Scope of Variables (Contd…)



```
class MyClass {
    . . .
    member variable declarations
    . . .
    public void aMethod( method parameters ) {
        . . .
        local variable declarations
        . . .
        catch ( exception handler parameters ) {
            . . .
        }
        . . .
    }
    . . .
}
```

Command Line Arguments

- Information that follows program's name on the command line when it is executed
- This data is passed to the application in the form of String arguments

```
class Echo {
public static void main (String args[]) {
for (int i = 0; i < args.length; i++)
System.out.println(args[i]);
 }
}
 Try this: Invoke the Echo application as
follows
C:\> java Echo Drink Hot Java
Drink
Hot
Java
```

note

Thank You