



## **Tutorial No. 2**

**Title: Implement the concept of constructor.**



Batch: SY-IT (B3)

Roll No.: 16010423076

Tutorial No.:2

**Aim:** Implement the concept of a constructor

---

**Resources needed:** JDK 8 or later, Text Editor/IDE

---

**Theory:**

---

### Constructors in Java

In Java, a constructor is a specific type of method that gets called automatically upon the creation of an instance of a class, guaranteeing proper initialization of the object. Constructors, sharing the same name as the class, are distinguished from regular methods by not having a return type. Parameterized constructors allow for initializing objects with specific values, while no-argument constructors provide default values. Constructors play a key role in setting up the initial state of an object, encapsulating the configuration logic, and allowing for versatility in creating objects. This starting process is essential for preserving the integrity and desired function of items in a Java program.

#### Properties / Features of Constructors:

1. **Same Name as the Class:** Constructors must have the same name as the class they are defined in.
2. **No Return Type:** Constructors do not have a return type, not even void.
3. **Called Automatically:** Constructors are called automatically when an object is instantiated using the new keyword.
4. **Overloading:** Multiple constructors can be defined in a class with different parameter lists.
5. **Default Constructor:** If no constructors are defined, Java provides a default no-argument constructor.
6. **Cannot be Inherited:** Constructors are not inherited by subclasses but can be called using the super keyword.
7. **Constructor Chaining:** One constructor can call another constructor in the same class using this keyword.
8. **Initialization of Final Fields:** Constructors are used to initialize final fields, which cannot be changed later.
9. **Special Member Methods:** Constructors are special methods without return types and are used to initialize object state.

#### Types of Constructors:

In Java, there are primarily two types of constructors: default constructors and parameterized constructors.

A *default constructor* is a constructor that does not take any arguments and is used to initialize objects with default values. The syntax for a default constructor is as follows:

```
public ClassName() {  
    // initialization code  
}
```

A *parameterized constructor*, on the other hand, takes one or more arguments, allowing the programmer to initialize objects with specific values at the time of creation. The syntax for a parameterized constructor is:

```
public ClassName(DataType1 parameter1, DataType2 parameter2, ...) {
    // initialization code
}
```

In addition to these, Java also allows the creation of a *copy constructor*, though it is not a built-in feature and must be manually implemented. A copy constructor initializes a new object as a copy of an existing object. The syntax for a copy constructor is:

```
public ClassName(ClassName existingObject) {
    // initialization code to copy existingObject's values
}
```

**Task: Prepare a document containing the answers of the following question**

1. Write a Java program to model a simple Library system with two classes: **Book** and **Library**. The **Library** class should have a default constructor to display the welcome message. The **Book** class should include a default constructor to initialise the attributes like title, author, isbn, and a parameterized constructor to initialize its attributes like Publication year, price, along with a method to get book details display book details.

**Code :**

```
class Book {
    String title;
    String author;
    String isbn;
    int publicationYear;
    int price;
```

```
    Book() {
        title = "No Title";
        author = "Unknown Author";
        isbn = "000-0000000000";
        publicationYear = 0;
        price = 0;
    }
```

```
    Book(String t, String a, String i, int year, int p) {
        title = t;
        author = a;
        isbn = i;
        publicationYear = year;
        price = p;
    }
```

```
    void displayDetails() {
```

```

        System.out.println("Title : " + title);
        System.out.println("Author : " + author);
        System.out.println("ISBN code : " + isbn);
        System.out.println("Year Published : " + publicationYear);
        System.out.println("Price : Rs " + price);
    }
}

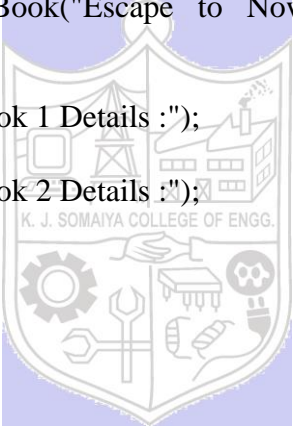
class Library {
    Library() {
        System.out.println("Welcome to my Library, the books are listed Below ");
    }
}

public class Main {
    public static void main(String[] args) {
        Library library = new Library();

        Book book1 = new Book();
        Book book2 = new Book("Escape to Nowhere", "Amar Bhushan", "978-9322008208", 2018, 356);

        System.out.println("\nBook 1 Details :");
        book1.displayDetails();
        System.out.println("\nBook 2 Details :");
        book2.displayDetails();
    }
}

```

**Output :**


```

Output
java -cp /tmp/7K3hs08RtC/Main
Welcome to my Library, the books are listed Below

Book 1 Details :
Title : No Title
Author : Unknown Author
ISBN code : 000-0000000000
Year Published : 0
Price : Rs 0

Book 2 Details :
Title : Escape to Nowhere
Author : Amar Bhushan
ISBN code : 978-9322008208
Year Published : 2018
Price : Rs 356

=== Code Execution Successful ===

```

2. Create a Java program to model a simple Course Registration system with one class: Student. The Student class should include a default constructor to initialize its attributes (name, studentId), a parameterized constructor to initialize its attributes (name, studentId, courseName, grade), and a copy constructor to create a copy of an existing student object. Implement methods in the Student class to display student details. In the main class, demonstrate the creation and usage of these constructors by creating instances of the Student class and displaying their details.

Code :

```
class Student {
    String name;
    String studentId;
    String courseName;
    String grade;

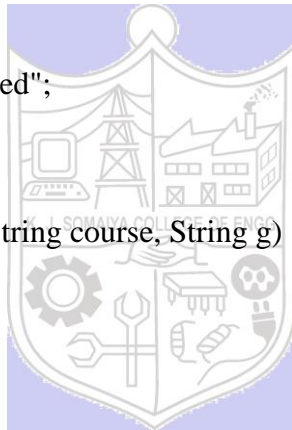
    Student() {
        name = "Unknown";
        studentId = "0000";
        courseName = "Not Registered";
        grade = "Not Assigned";
    }

    Student(String n, String id, String course, String g) {
        name = n;
        studentId = id;
        courseName = course;
        grade = g;
    }

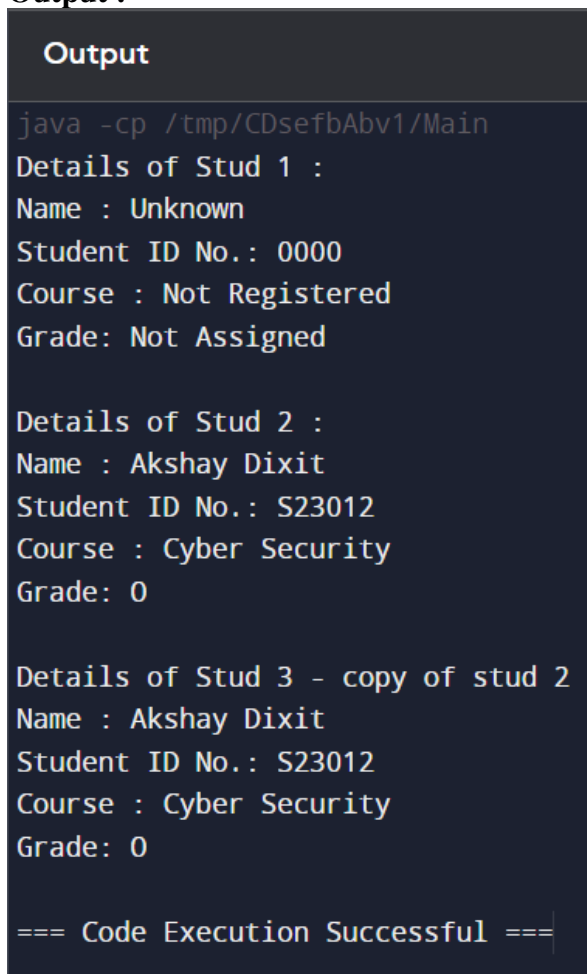
    Student(Student student) {
        name = student.name;
        studentId = student.studentId;
        courseName = student.courseName;
        grade = student.grade;
    }

    void displayDetails() {
        System.out.println("Name : " + name);
        System.out.println("Student ID No.: " + studentId);
        System.out.println("Course : " + courseName);
        System.out.println("Grade: " + grade);
    }
}

//main public class
public class Main {
```



```
public static void main(String[] args) {  
    Student student1 = new Student();  
    Student student2 = new Student("Akshay Dixit", "S23012", "Cyber Security", "O");  
    Student student3 = new Student(student2);  
    System.out.println("Details of Stud 1 : ");  
    student1.displayDetails();  
    System.out.println("\nDetails of Stud 2 : ");  
    student2.displayDetails();  
    System.out.println("\nDetails of Stud 3 - copy of stud 2");  
    student3.displayDetails();  
}
```

**Output :**

**Output**

```
java -cp /tmp/CDsefbAbv1/Main  
Details of Stud 1 :  
Name : Unknown  
Student ID No.: 0000  
Course : Not Registered  
Grade: Not Assigned  
  
Details of Stud 2 :  
Name : Akshay Dixit  
Student ID No.: S23012  
Course : Cyber Security  
Grade: 0  
  
Details of Stud 3 - copy of stud 2  
Name : Akshay Dixit  
Student ID No.: S23012  
Course : Cyber Security  
Grade: 0  
  
=== Code Execution Successful ===
```

---

**Outcomes:**

CO1: Apply fundamental Object Oriented Methodology concepts using java programming

---

**Conclusion: (Conclusion to be based on the outcomes achieved)**

From this article I have learned more about the basics of Object-Oriented Programming, how to design and develop classes, constructors and methods. I have learned through the first program how modeling a simple Library system helped me in practicing creating and using default and parameterized constructors. The second program, where I developed a Course Registration system, further solidified my understanding of OOP.

---

**Grade: AA / AB / BB / BC / CC / CD /DD**

Signature of faculty in-charge with date

---

**References Books**

1. Herbert Schildt; JAVA The Complete Reference; Seventh Edition, Tata McGraw-Hill Publishing Company Limited 2007.
2. Java 7 Programming - Black Book : Kogent Learning Solutions Inc.
3. Sachin Malhotra, Saurabh Chaudhary "Programming in Java", Oxford University Press, 2010
4. Jaime Nino, Frederick A. Hosch, 'An introduction to Programming and Object Oriented Design using Java', Wiley Student Edition.