

## Experiment No.1

Student's Name: Rajdeep Jaiswal

Uid: 20BCS2761

Semester: 3<sup>rd</sup>

Branch: CSE

### Aim/Overview of the practical:

*To study different types of constructors in java.*

## What is a Constructor?

A constructor in Java is similar to a method that is invoked when an object of the class is created.

Unlikes Java Method, a constructor has the same name as that of the class and does not have any return type. For example,

```
class Test {  
    Test() {  
        // constructor body  
    }  
}
```

Here, Test() is a constructor. It has the same name as that of the class and doesn't have a return type.

## Types of Constructor:-

In Java, constructors can be divided into 3 types:

1. No-Arg Constructor
2. Parameterized Constructor
3. Default Constructor

### Topic:1.

**Write a Program to understand the concept of No-Arg Constructor**

### Program Code:

Similar to methods, a Java constructor may or may not have any parameters (arguments).

If a constructor does not accept any parameters, it is known as a no-argument constructor.

For example,

```
private Constructor() {  
    // body of the constructor  
}
```

```
class Main {  
  
    int i;  
  
    // constructor with no parameter  
    private Main() {  
        i = 5;  
        System.out.println("Constructor is called");  
    }  
  
    public static void main(String[] args) {  
  
        // calling the constructor without any parameter  
        Main obj = new Main();  
        System.out.println("Value of i: " + obj.i);  
    }  
}
```

## Output:

```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" -agentlib:jdwp=transport=dt_socket,address=127.0.0.1:55775,suspend=y,server=n -javaagent:C:  
Connected to the target VM, address: '127.0.0.1:55775', transport: 'socket'  
Java HotSpot(TM) 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been appended  
Constructor is called  
Value of i: 5  
Disconnected from the target VM, address: '127.0.0.1:55775', transport: 'socket'  
  
Process finished with exit code 0
```

## Explanation:

*In the above example, we have created a constructor **main()**. Here, the constructor does not accept any parameters. Hence, it is known as a no-arg constructor.*

## Topic:2

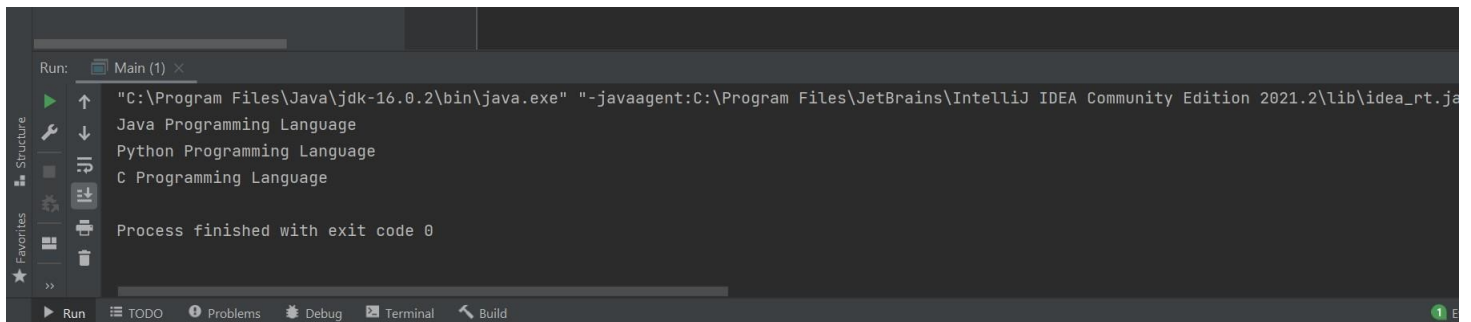
### Write a Program to understand the concept of Parameterized Constructor

A Java constructor can also accept one or more parameters. Such constructors are known as parameterized constructors (constructor with parameters).

### Program Code:

```
class Main {  
  
    String languages;  
  
    // constructor accepting single value  
    Main(String lang) {  
        languages = lang;  
        System.out.println(languages + " Programming Language");  
    }  
  
    public static void main(String[] args) {  
  
        // call constructor by passing a single value  
        Main obj1 = new Main("Java");  
        Main obj2 = new Main("Python");  
        Main obj3 = new Main("C");  
  
    }  
}
```

### Output:



The screenshot shows the Run console in IntelliJ IDEA. The output is as follows:

```
Run: Main (1) x  
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2\lib\idea_rt.jar" -Dfile.encoding=UTF-8  
Java Programming Language  
Python Programming Language  
C Programming Language  
Process finished with exit code 0
```

## **Explanation:**

In the above example, we have created a constructor named **main()**. Here, the constructor takes a single parameter. Notice the expression,

```
Main obj1 = new Main("Java");
```

Here, we are passing the single value to the constructor. Based on the argument passed, the language variable is initialized inside the constructor.

## **Topic:3.**

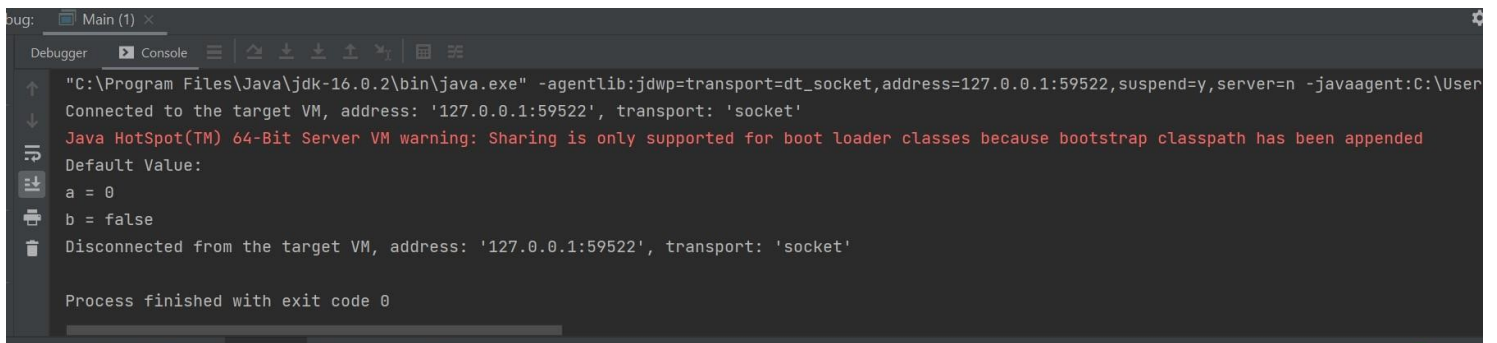
### **Write a Program to understand the concept of Default Constructor**

If we do not create any constructor, the Java compiler automatically create a no-arg constructor during the execution of the program. This constructor is called default constructor.

## Program Code:

```
class Main {  
  
    int a;  
    boolean b;  
  
    public static void main(String[] args) {  
  
        // A default constructor is called  
        Main obj = new Main();  
  
        System.out.println("Default Value:");  
        System.out.println("a = " + obj.a);  
        System.out.println("b = " + obj.b);  
  
    }  
}
```

## Output:



```
Debugger Console  
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" -agentlib:jdwp=transport=dt_socket,address=127.0.0.1:59522,suspend=y,server=n -javaagent:C:\User  
Connected to the target VM, address: '127.0.0.1:59522', transport: 'socket'  
Java HotSpot(TM) 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been appended  
Default Value:  
a = 0  
b = false  
Disconnected from the target VM, address: '127.0.0.1:59522', transport: 'socket'  
  
Process finished with exit code 0
```

## Explanation:

Here, we haven't created any constructors. Hence, the Java compiler automatically creates the default constructor.

The default constructor initializes any uninitialized instance variables with default values.

**Learning outcomes (What I have learnt):**

- Constructors are invoked implicitly when you instantiate objects.
- The two rules for creating a constructor are:  
The name of the constructor should be the same as the class.  
A Java constructor must not have a return type.
- If a class doesn't have a constructor, the Java compiler automatically creates a **default constructor** during run-time. The default constructor initializes instance variables with default values. For example, the `int` variable will be initialized to `0`
- Constructor types:  
**No-Arg Constructor** - a constructor that does not accept any arguments  
**Parameterized constructor** - a constructor that accepts arguments  
**Default Constructor** - a constructor that is automatically created by the Java compiler if it is not explicitly defined.
- A constructor cannot be `abstract` or `static` or `final`.
- A constructor can be overloaded but can not be overridden

**Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):**

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			