return keyword in Java

Difficulty Level : Easy   Last Updated : 18 Nov, 2021

In Java, **return** is a reserved keyword i.e, we can’t use it as an identifier. It is used to **exit** from a method, with or without a value. Usage of **return keyword** as there exist two ways as listed below as follows:

- **Case 1:** Methods returning a value
- **Case 2:** Methods not returning a value

Let us illustrate by directly implementing them as follows:

Case 1: Methods returning a value

> *For methods that **define** a return type, return statement **must be** immediately followed by return value.*

Example:

```java
// Java Program to Illustrate Usage of return Keyword

// Main method
class GFG {

    // Method 1
    // Since return type of RR method is double
    // so this method should return double value
    double RR(double a, double b) {
        double sum = 0;
        sum = (a + b) / 2.0;

        // Return statement as we already above have declared
        // return type to be double
        return sum;
    }

    // Method 2
    // Main driver method
    public static void main(String[] args)
    {
        // Print statement
        System.out.println(new A().RR(5.5, 6.5));
    }
}
```

Output

```
6.0
```

Output explanation: When we are calling a class GFG method that has **return sum** which returns the value of sum and that's value gets displayed on the console.

Case 2: Methods not returning a value

For methods that do not return a value, return statement in Java can be skipped. here there arise two cases when there is no value been returned by the user as listed below as follows:

- **#1:** Method not using return statement in void function
- **#2:** Methods with return type void

#1: Method not using return statement in void function

Example

```
// Java program to illustrate no return
// keyword needed inside void method

// Main class
class GFG {

    // Since return type of RR method is
    // void so this method shouldn't return any value
    void demoSum(int a, int b)
    {
        int sum = 0;
        sum = (a + b) / 10;
        System.out.println(sum);

        // No return statement in this method
    }
    // Method 2
    // Main driver method
    public static void main(String[] args)
    {
        // Calling the method
        // Over custom inputs
        new GFG().demosum(5, 5);

        // Display message on the console for successful
        // execution of the program
        System.out.print(
            "No return keyword is used and program executed successfully");
    }

    // Note here we are not returning anything
    // as the return type is void
}
```

Output

```
1
No return keyword is used and program executed successfully
```

> ***Note:** Return statement not required (**but can be used**) for methods with return type void. We can use "return;" which means **not return anything**.*

#2: Methods with ***void return type***

Example 1-A:

```
// Java program to illustrate usage of
// return keyword in void method

// Class 1
// Main class
class GFG {

    // Method 1
    // Since return type of RR method is
    // void so this method should not return any value
    void demofunction(double j)
    {
        if (j < 9)

            // return statement below(only using
            // return statement and not returning
            // anything):
            // control exits the method if this
            // condition(i.e, j<9) is true.
            return;
        ++j;
    }

    // Method 2
    // Main driver method
    public static void main(String[] args)
    {
        // Calling above method declared in above class
        new GFG().demofunction(5.5);

        // Display message on console to illustrate
        // successful execution of program
        System.out.println("Program executed successfully");
    }
}
```

Output

```
Program executed successfully
```

Output explanation: If the statement **if(j<9)** is true then control **exits** from the method and does **not** execute the rest of the statement of the RR method and hence comes back again to *main() method*.

Now moving ahead geek you must be wondering what if we do use return statement at the end of the program?

return statement can be used at various places in the method but we need to ensure that it must be the last statement to get executed in a method.

> ***Note**: return statement **need not to be last statement in a method, but it must be last statement to execute** in a method.*

Example 1-B:

```
// Java program to illustrate return must not be always
// last statement, but must be last statement
// in a method to execute

// Main class
class GFG {

    // Method 1
    // Helper method
    // Since return type of RR method is void
    // so this method should not return any value
    void demofunction(double i)
    {
        // Demo condition check
        if (i < 9)

            // See here return need not be last
            // statement but must be last statement
            // in a method to execute
            return;

        else
            ++i;
    }

    // Method 2
    // main driver method
    public static void main(String[] args)
    {
        // Calling the method
        new GFG().demofunction(7);

        // Display message to illustrate
        // successful execution of program
        System.out.println("Program executed successfully");
    }
}
```

Output

```
Program executed successfully
```

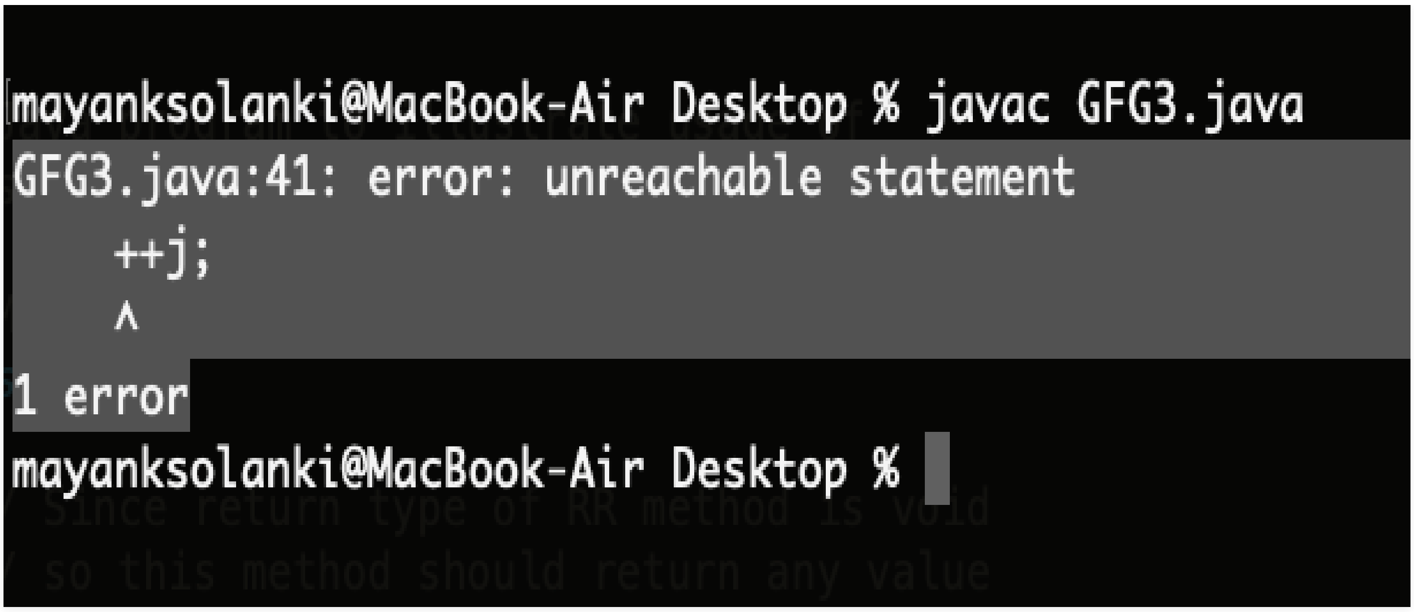
Output explanation:

As the condition **(i<9)** becomes true, it executes **return** statement, and hence flow comes **out** of 'demofunction' method and comes back again to main. Following this, the ***return statement*** must be the last statement to execute in a method, which means **there is no** point in defining any code after return which is clarified below as follows:

Example 2A

```java
// Java program to illustrate usage of
// statement after return statement

// Main class
class GFG {

    // Since return type of RR method is void
    // so this method should return any value
    // Method 1
    void demofunction(double j)
    {
        return;

        // Here get compile error since can't
        // write any statement after return keyword

        ++j;
    }

    // Method 2
    // Main driver method
    public static void main(String[] args)
    {

        // Calling the above defined function
        new GFG().demofunction(5);
    }
}
```

Output:



```
mayanksolanki@MacBook-Air Desktop % javac GFG3.java
GFG3.java:41: error: unreachable statement
    ++j;
    ^
1 error
mayanksolanki@MacBook-Air Desktop %
```

Example 2-B

```java
// Java program to illustrate usage
// of return keyword

// Main class
class GFG {

    // Since return type of RR method is
    // void so this method should not return any value
    // Method 1
    void demofunction(double val)
    {

        // Condition check
        if (val < 0) {

            System.out.println(val);
            return;

            // System.out.println("oshea");
        }
        else
            ++val;
    }

    // Method 2
    // Main drive method
    public static void main(String[] args)
    {

        // CAlling the above method
        new GFG().demofunction(-1);

        // Display message to illustrate
        // successful execution of program
        System.out.println("Program Executed Successfully");
    }
}
```

Output

```
-1.0
Program Executed Successfully
```

Note: *In the above program we do uncomment statements it will throw an error.*

This article is contributed by **Rajat Rawat**. If you like GeeksforGeeks and would like to contribute, you can also write an article using write.geeksforgeeks.org or mail your article to review-team@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.