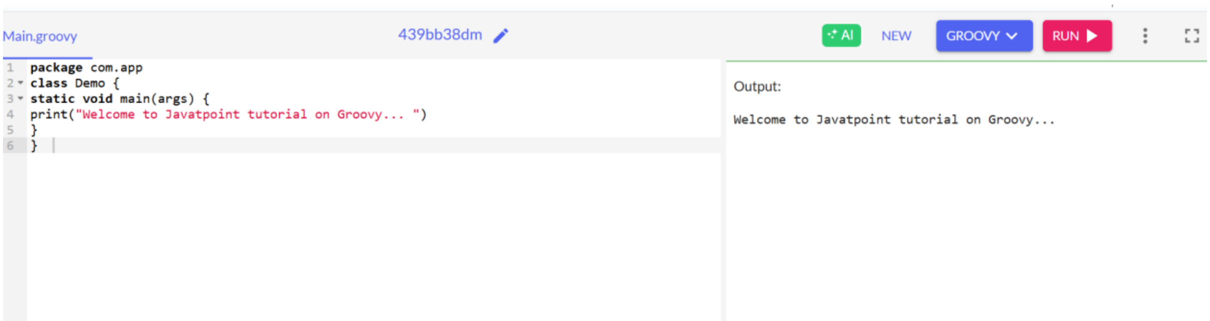
GROOVY PROGAMS ASSIGNMENT

First Groovy Program



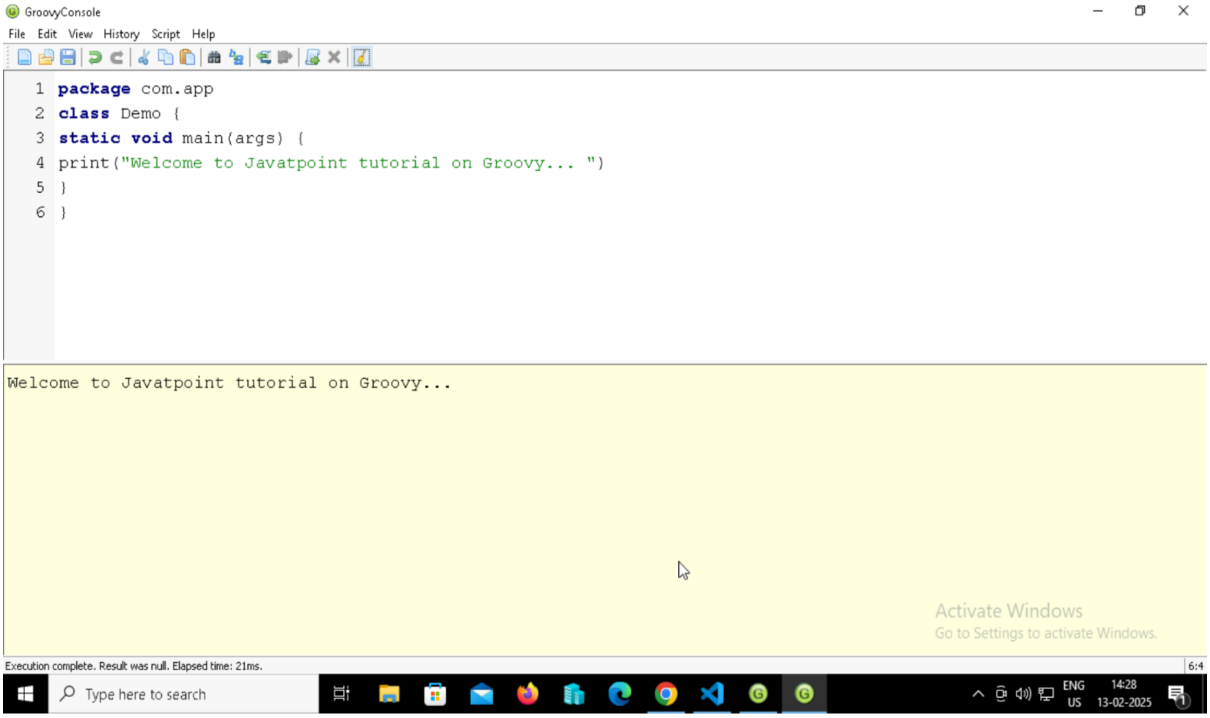
```
package com.app
class Demo {
static void main(args) {
print("Welcome to Javatpoint tutorial on Groovy... ")
}
}
```

Output:

```
Welcome to Javatpoint tutorial on Groovy...
```

Groovy basic syntax

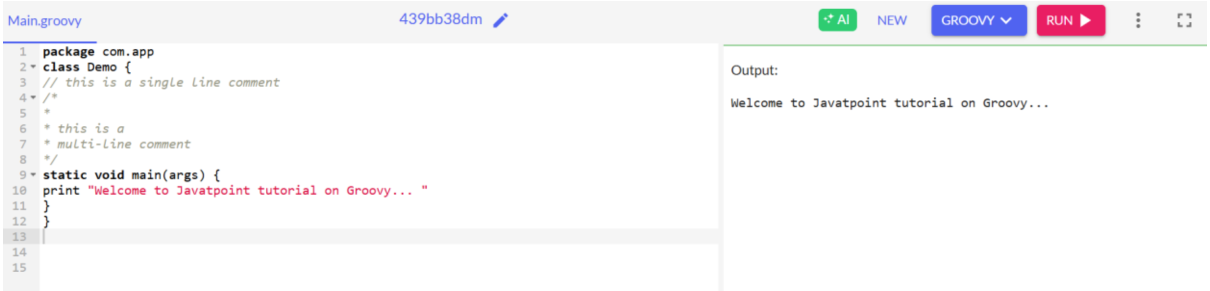
Example programs



```
package com.app
class Demo {
static void main(args) {
print("Welcome to Javatpoint tutorial on Groovy... ")
}
}
```

```
Welcome to Javatpoint tutorial on Groovy...
```

Execution complete. Result was null. Elapsed time: 21ms.



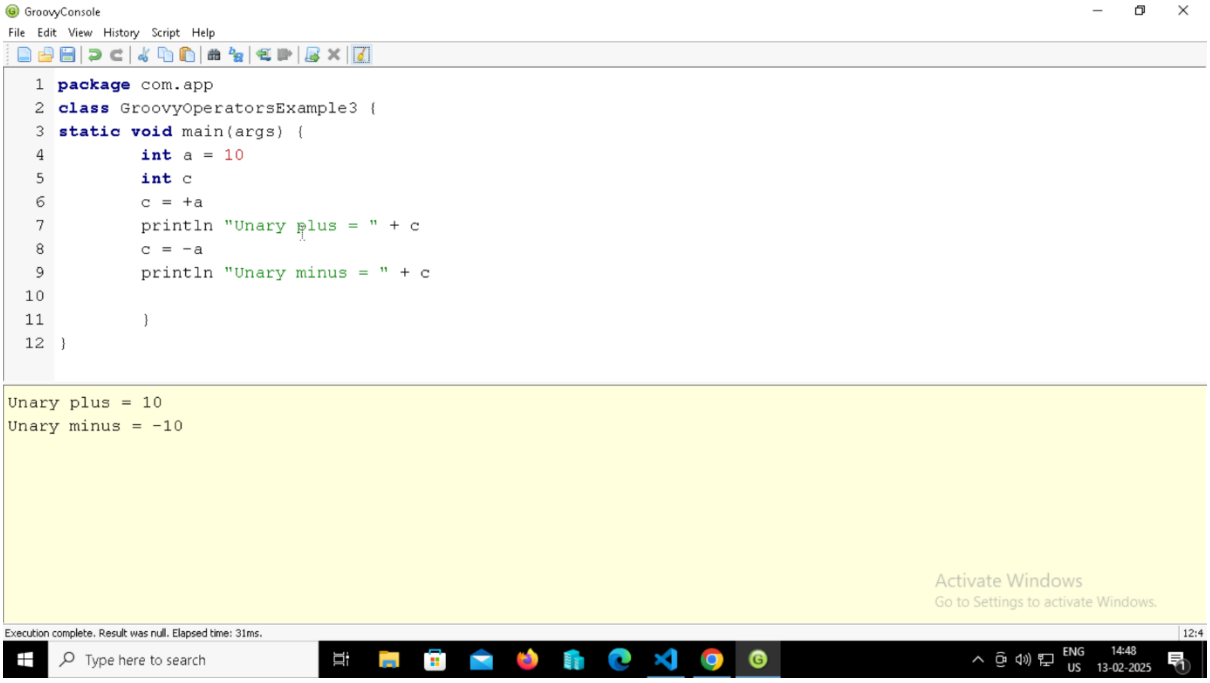
```
package com.app
class Demo {
// this is a single line comment
/*
*
* this is a
* multi-line comment
*/
static void main(args) {
print "Welcome to Javatpoint tutorial on Groovy... "
}
}
```

Output:

```
Welcome to Javatpoint tutorial on Groovy...
```

GROOVY OPERATORS

Groovy unary operator

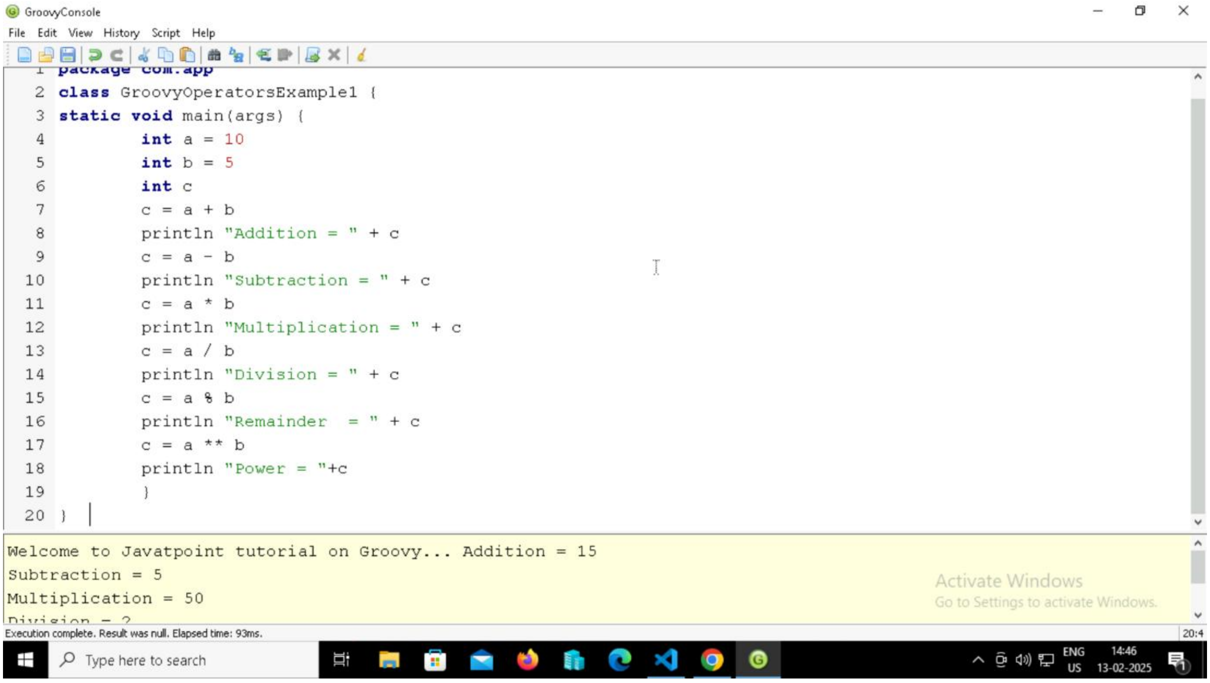


```
package com.app
class GroovyOperatorsExample3 {
static void main(args) {
        int a = 10
        int c
        c = +a
        println "Unary plus = " + c
        c = -a
        println "Unary minus = " + c

        }
}
```

```
Unary plus = 10
Unary minus = -10
```

Execution complete. Result was null. Elapsed time: 31ms.

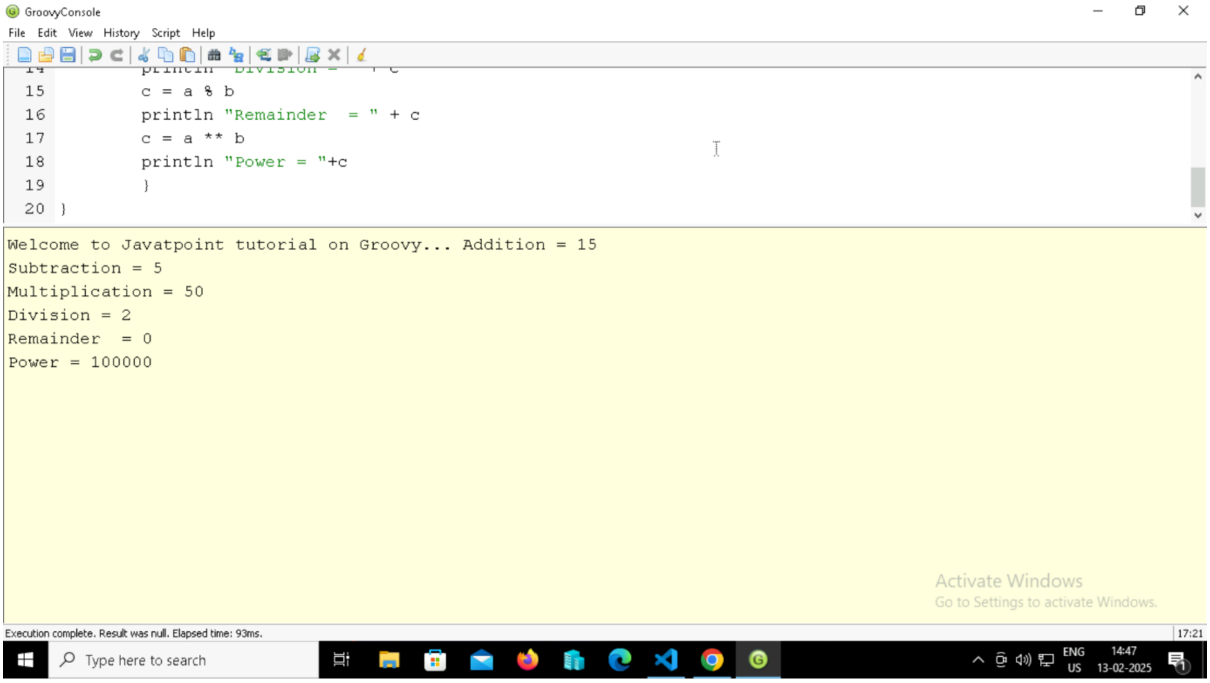
Groovy arthimetic operators



```
package com.app
class GroovyOperatorsExample1 {
static void main(args) {
        int a = 10
        int b = 5
        int c
        c = a + b
        println "Addition = " + c
        c = a - b
        println "Subtraction = " + c
        c = a * b
        println "Multiplication = " + c
        c = a / b
        println "Division = " + c
        c = a % b
        println "Remainder  = " + c
        c = a ** b
        println "Power = "+c
        }
}
```

```
Welcome to Javatpoint tutorial on Groovy... Addition = 15
Subtraction = 5
Multiplication = 50
Division = 2
```

Execution complete. Result was null. Elapsed time: 93ms.

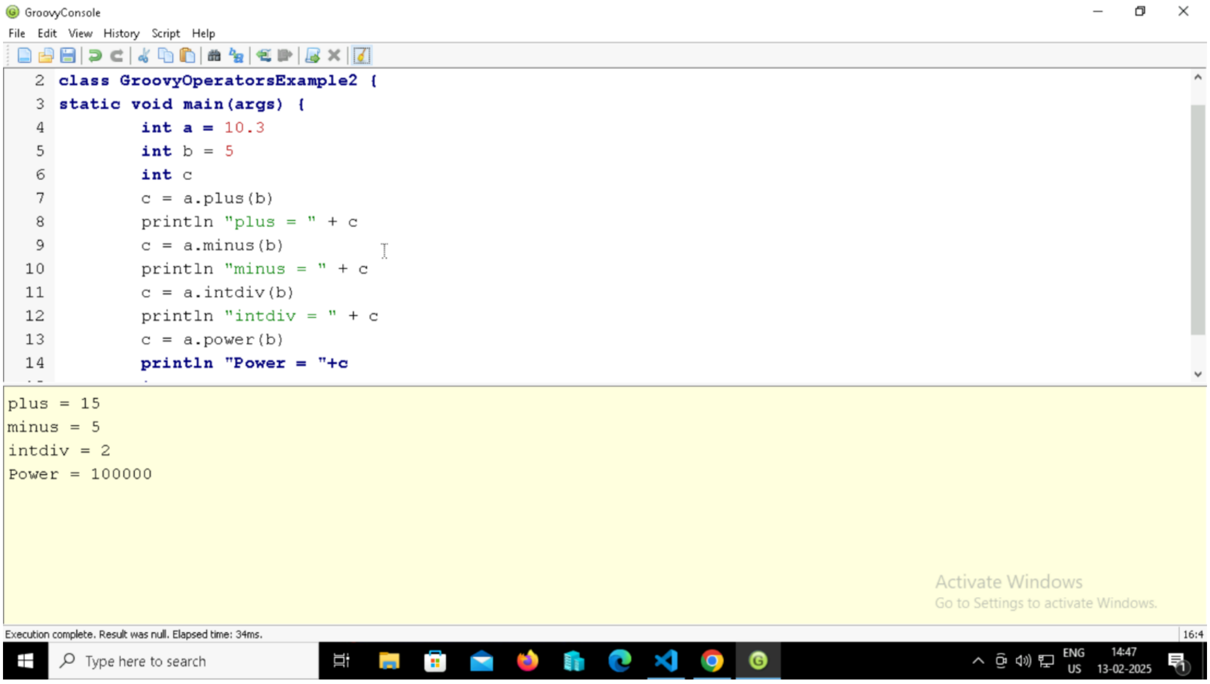


```
        println "Division = " + c
        c = a % b
        println "Remainder  = " + c
        c = a ** b
        println "Power = "+c
        }
}
```

```
Welcome to Javatpoint tutorial on Groovy... Addition = 15
Subtraction = 5
Multiplication = 50
Division = 2
Remainder  = 0
Power = 100000
```

Execution complete. Result was null. Elapsed time: 93ms.

Groovy arthimetic operator example 2

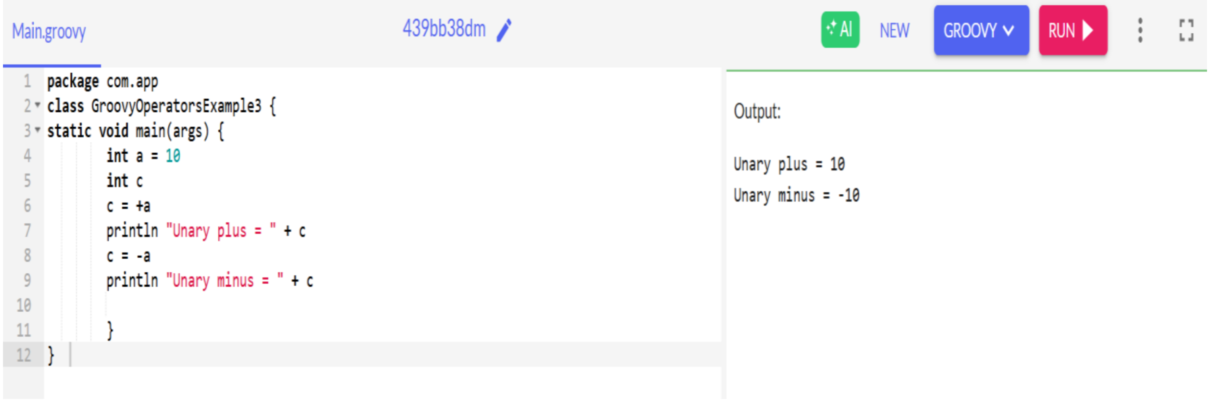


```
class GroovyOperatorsExample2 {
static void main(args) {
        int a = 10.3
        int b = 5
        int c
        c = a.plus(b)
        println "plus = " + c
        c = a.minus(b)
        println "minus = " + c
        c = a.intdiv(b)
        println "intdiv = " + c
        c = a.power(b)
        println "Power = "+c
```

```
plus = 15
minus = 5
intdiv = 2
Power = 100000
```

Execution complete. Result was null. Elapsed time: 34ms.

Groovy unary operator 2

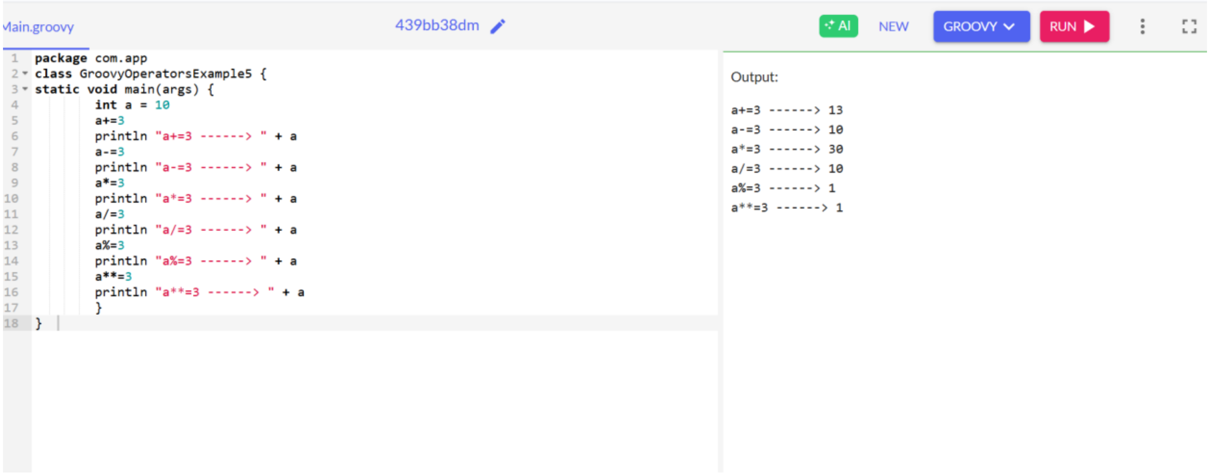


```
package com.app
class GroovyOperatorsExample3 {
static void main(args) {
        int a = 10
        int c
        c = +a
        println "Unary plus = " + c
        c = -a
        println "Unary minus = " + c

        }
}
```

Output:

```
Unary plus = 10
Unary minus = -10
```

Groovy Assignment Operator

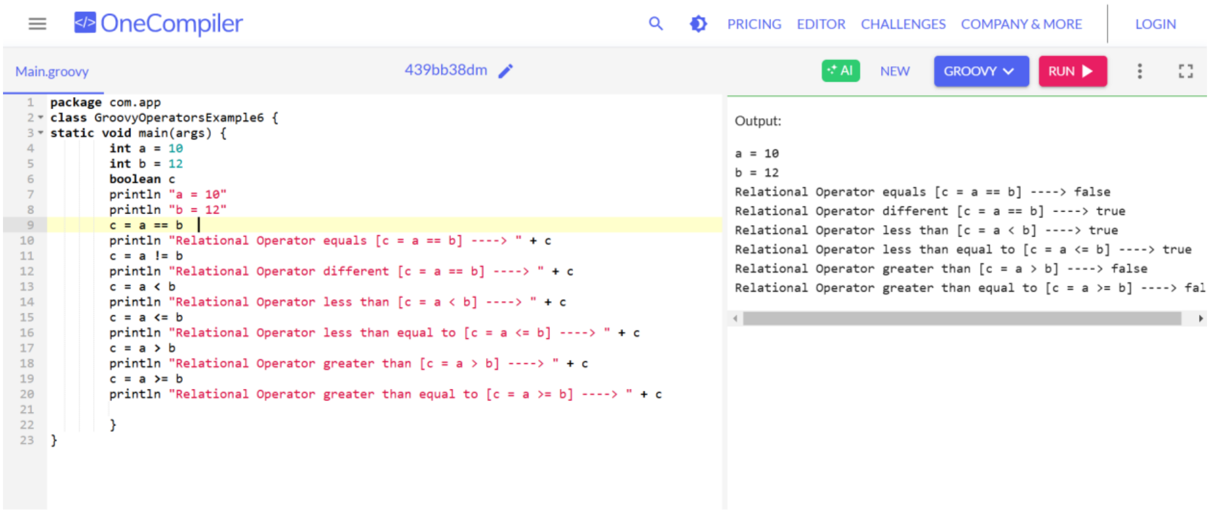


```
package com.app
class GroovyOperatorsExample5 {
static void main(args) {
        int a = 10
        a+=3
        println "a+=3 ------> " + a
        a-=3
        println "a-=3 ------> " + a
        a*=3
        println "a*=3 ------> " + a
        a/=3
        println "a/=3 ------> " + a
        a%=3
        println "a%=3 ------> " + a
        a**=3
        println "a**=3 ------> " + a
        }
}
```

Output:

```
a+=3 ------> 13
a-=3 ------> 10
a*=3 ------> 30
a/=3 ------> 10
a%=3 ------> 1
a**=3 ------> 1
```

Relational operators



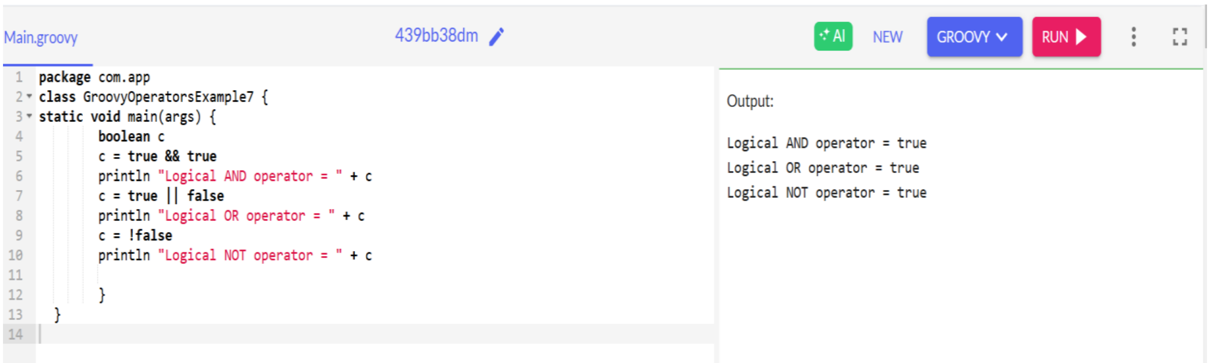
```
package com.app
class GroovyOperatorsExample6 {
static void main(args) {
        int a = 10
        int b = 12
        boolean c
        println "a = 10"
        println "b = 12"
        c = a == b
        println "Relational Operator equals [c = a == b] ----> " + c
        c = a != b
        println "Relational Operator different [c = a == b] ----> " + c
        c = a < b
        println "Relational Operator less than [c = a < b] ----> " + c
        c = a <= b
        println "Relational Operator less than equal to [c = a <= b] ----> " + c
        c = a > b
        println "Relational Operator greater than [c = a > b] ----> " + c
        c = a >= b
        println "Relational Operator greater than equal to [c = a >= b] ----> " + c

        }
}
```

Output:

```
a = 10
b = 12
Relational Operator equals [c = a == b] ----> false
Relational Operator different [c = a == b] ----> true
Relational Operator less than [c = a < b] ----> true
Relational Operator less than equal to [c = a <= b] ----> true
Relational Operator greater than [c = a > b] ----> false
Relational Operator greater than equal to [c = a >= b] ----> fal
```

Logical operators

Example 1

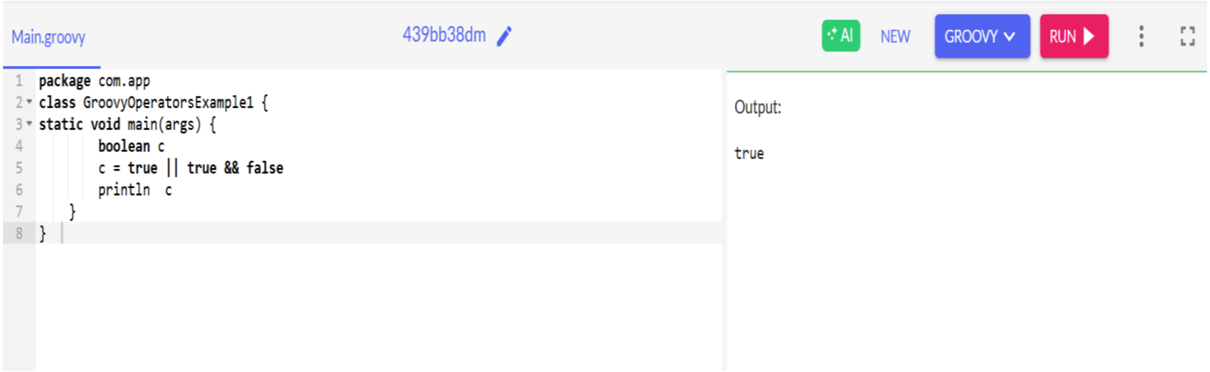


```
package com.app
class GroovyOperatorsExample7 {
static void main(args) {
        boolean c
        c = true && true
        println "Logical AND operator = " + c
        c = true || false
        println "Logical OR operator = " + c
        c = !false
        println "Logical NOT operator = " + c

        }
  }

```

Output:

```
Logical AND operator = true
Logical OR operator = true
Logical NOT operator = true
```

Example 2

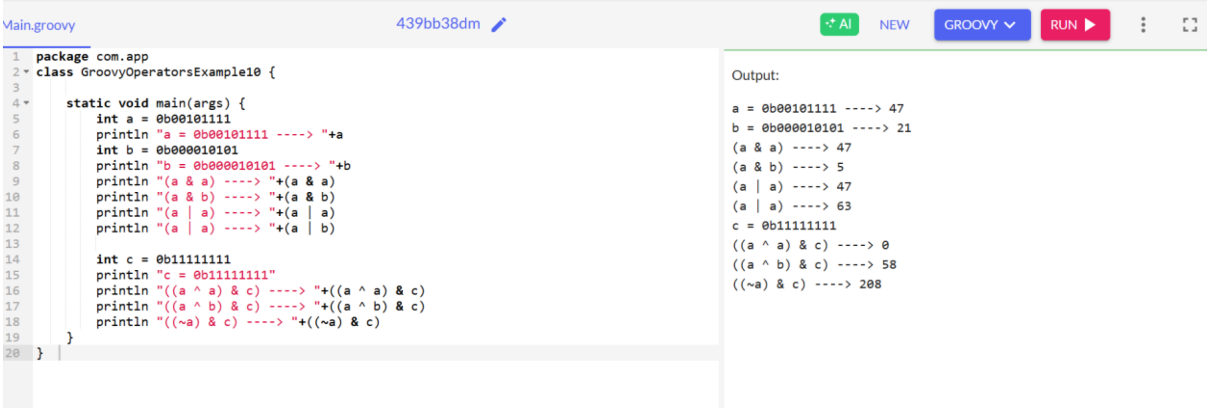


```
package com.app
class GroovyOperatorsExample1 {
static void main(args) {
        boolean c
        c = true || true && false
        println  c
    }
}
```

Output:

```
true
```

Bitwise operators

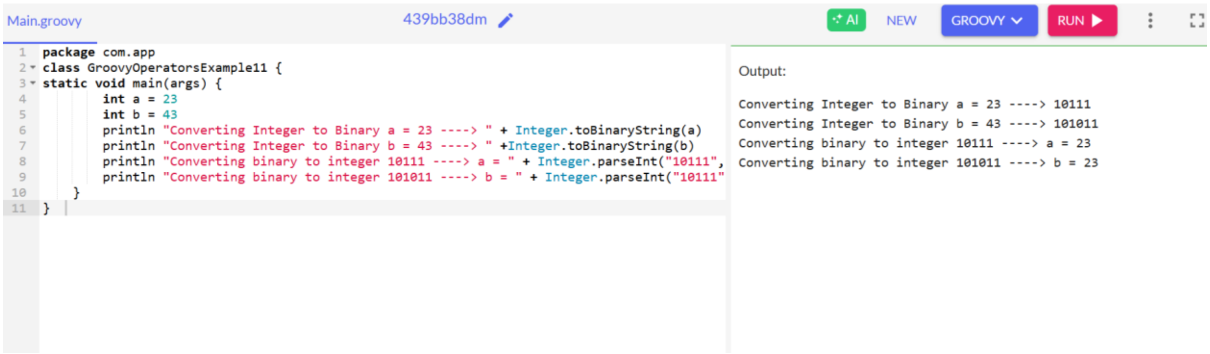


```
package com.app
class GroovyOperatorsExample10 {

    static void main(args) {
        int a = 0b00101111
        println "a = 0b00101111 ----> "+a
        int b = 0b000010101
        println "b = 0b000010101 ----> "+b
        println "(a & a) ----> "+(a & a)
        println "(a & b) ----> "+(a & b)
        println "(a | a) ----> "+(a | a)
        println "(a | a) ----> "+(a | b)

        int c = 0b11111111
        println "c = 0b11111111"
        println "((a ^ a) & c) ----> "+((a ^ a) & c)
        println "((a ^ b) & c) ----> "+((a ^ b) & c)
        println "((~a) & c) ----> "+((~a) & c)
    }
}
```

Output:

```
a = 0b00101111 ----> 47
b = 0b000010101 ----> 21
(a & a) ----> 47
(a & b) ----> 5
(a | a) ----> 47
(a | a) ----> 63
c = 0b11111111
((a ^ a) & c) ----> 0
((a ^ b) & c) ----> 58
((~a) & c) ----> 208
```

Example 2



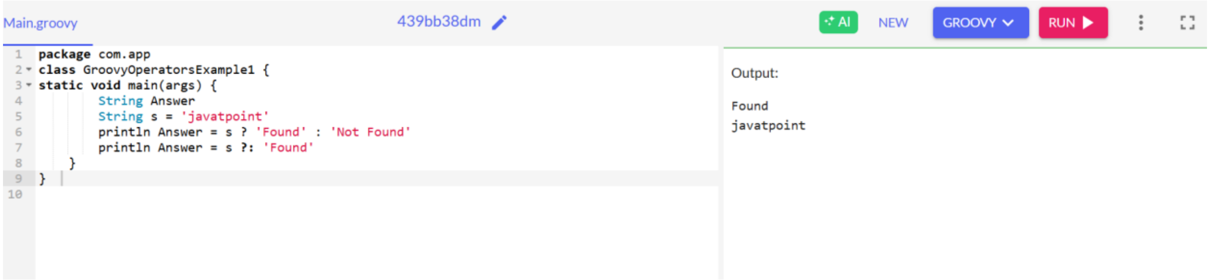
```groovy
package com.app
class GroovyOperatorsExample11 {
static void main(args) {
        int a = 23
        int b = 43
        println "Converting Integer to Binary a = 23 ----> " + Integer.toBinaryString(a)
        println "Converting Integer to Binary b = 43 ----> " +Integer.toBinaryString(b)
        println "Converting binary to integer 10111 ----> a = " + Integer.parseInt("10111",
        println "Converting binary to integer 101011 ----> b = " + Integer.parseInt("10111"
    }
}
```

Output:

```
Converting Integer to Binary a = 23 ----> 10111
Converting Integer to Binary b = 43 ----> 101011
Converting binary to integer 10111 ----> a = 23
Converting binary to integer 101011 ----> b = 23
```

Conditional operators

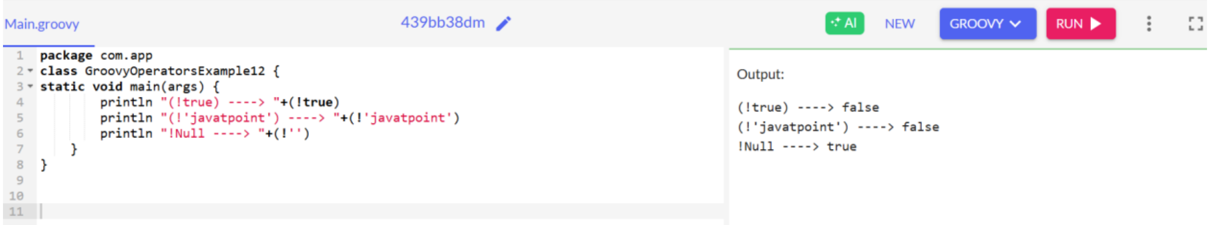
Example programs



```groovy
package com.app
class GroovyOperatorsExample1 {
static void main(args) {
        String Answer
        String s = 'javatpoint'
        println Answer = s ? 'Found' : 'Not Found'
        println Answer = s ?: 'Found'
    }
}
```

Output:

```
Found
javatpoint
```



```groovy
package com.app
class GroovyOperatorsExample12 {
static void main(args) {
        println "(!true) ----> "+(!true)
        println "(!'javatpoint') ----> "+(!'javatpoint')
        println "!Null ----> "+(!'')
    }
}
```

Output:

```
(!true) ----> false
(!'javatpoint') ----> false
!Null ----> true
```