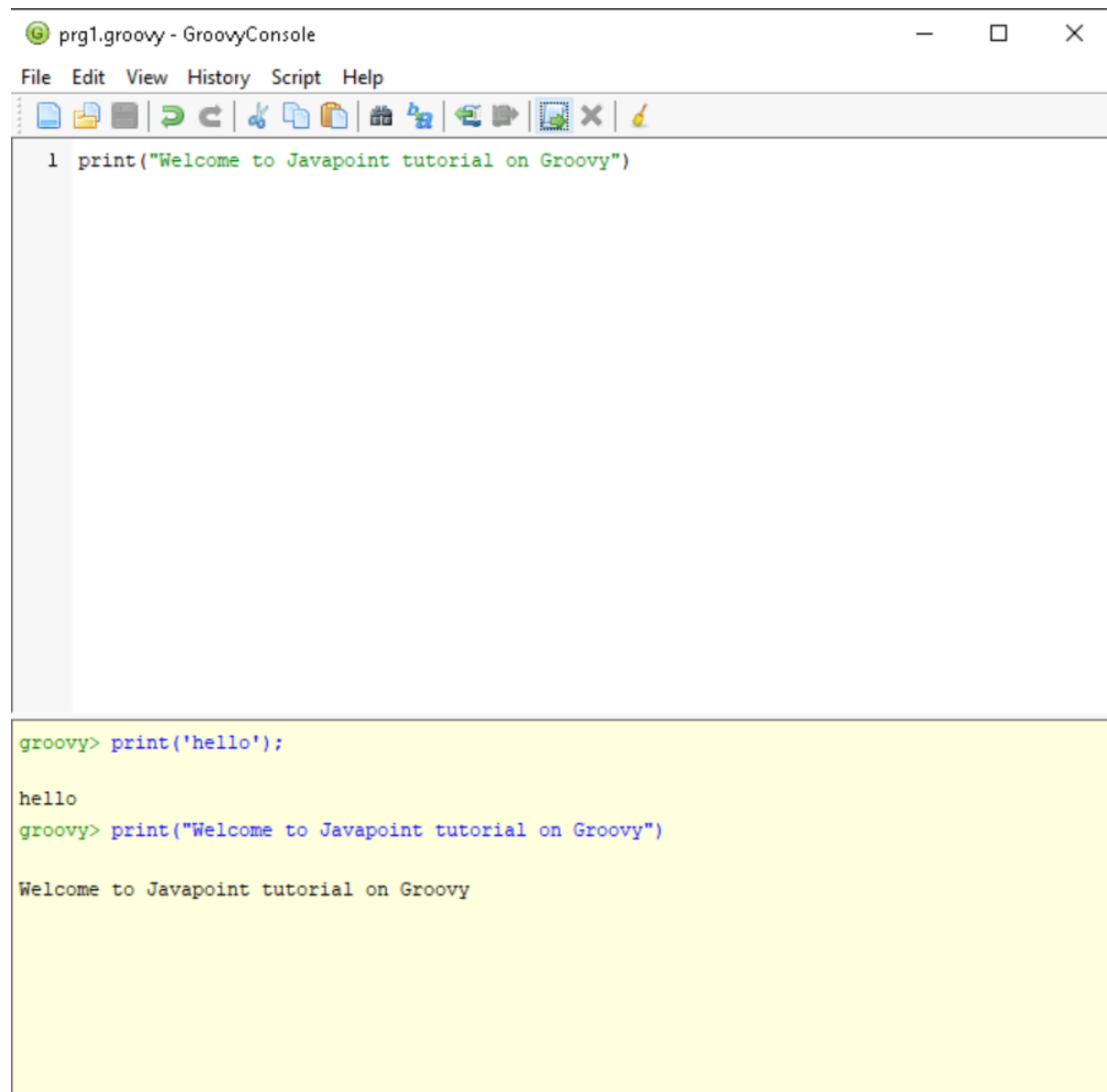


## Basics of Groovy

Q.



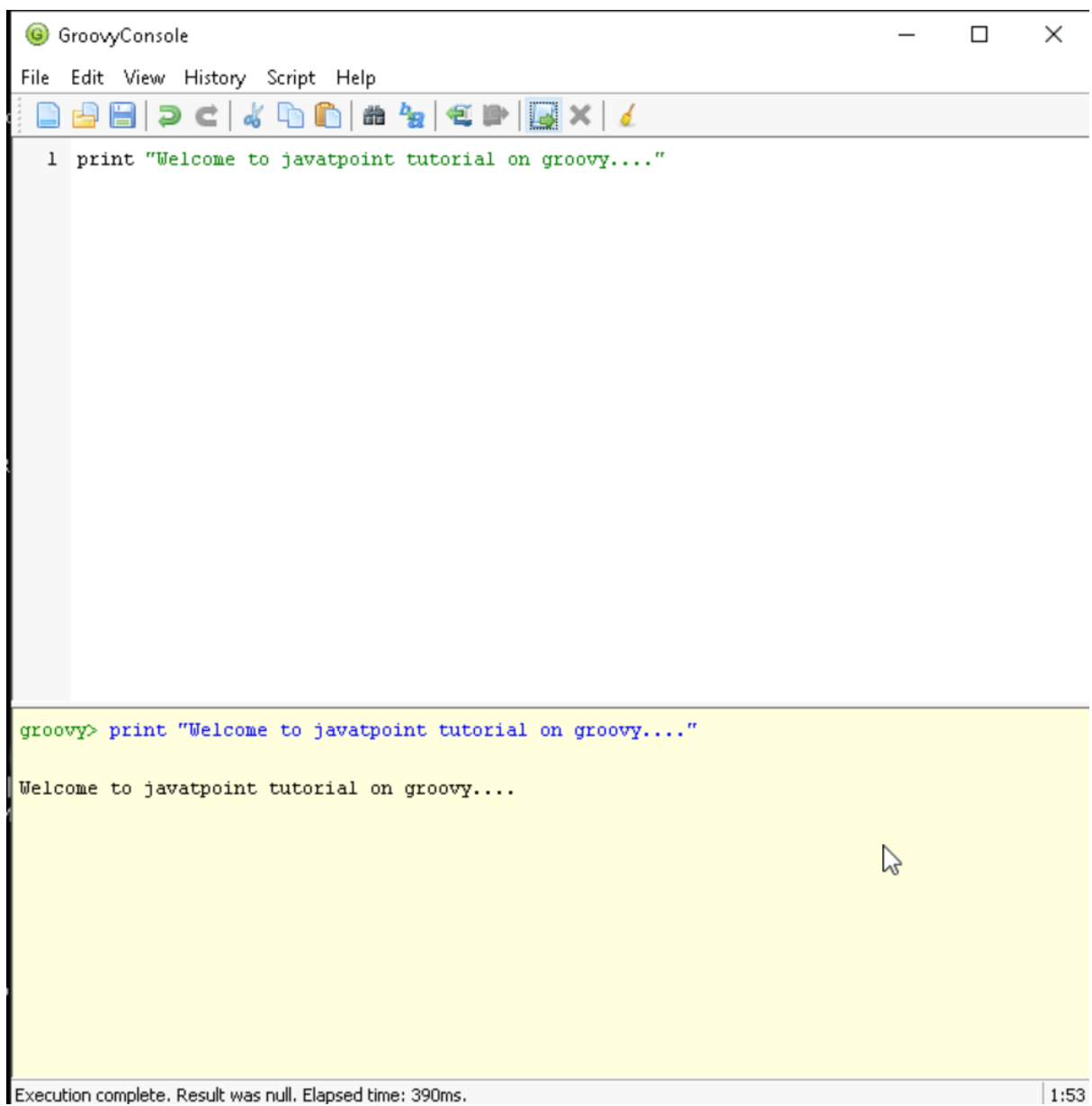
The screenshot shows a window titled "prg1.groovy - GroovyConsole". The menu bar includes "File", "Edit", "View", "History", "Script", and "Help". The toolbar contains icons for file operations (new, open, save, print), editing (undo, redo, cut, copy, paste), and execution (run, stop, clear). The script editor contains a single line of code: `1 print("Welcome to Javapoint tutorial on Groovy")`. The output console, which has a yellow background, shows the following sequence of commands and results: `groovy> print('hello');` followed by the output `hello`; `groovy> print("Welcome to Javapoint tutorial on Groovy")` followed by the output `Welcome to Javapoint tutorial on Groovy`.

```
prg1.groovy - GroovyConsole
File Edit View History Script Help

1 print("Welcome to Javapoint tutorial on Groovy")

groovy> print('hello');
hello
groovy> print("Welcome to Javapoint tutorial on Groovy")
Welcome to Javapoint tutorial on Groovy
```

Q.



The screenshot shows the GroovyConsole application window. The title bar reads "GroovyConsole" with standard window controls. The menu bar includes "File", "Edit", "View", "History", "Script", and "Help". The toolbar contains icons for file operations (new, open, save, print), editing (undo, redo, cut, copy, paste), and execution (run, stop, clear). The script editor contains a single line of Groovy code: `1 print "Welcome to javatpoint tutorial on groovy...."`. The console output area, which has a yellow background, shows the command `groovy> print "Welcome to javatpoint tutorial on groovy...."` and the output `Welcome to javatpoint tutorial on groovy....`. A mouse cursor is visible in the console area. The status bar at the bottom indicates "Execution complete. Result was null. Elapsed time: 390ms." and the time "1:53".

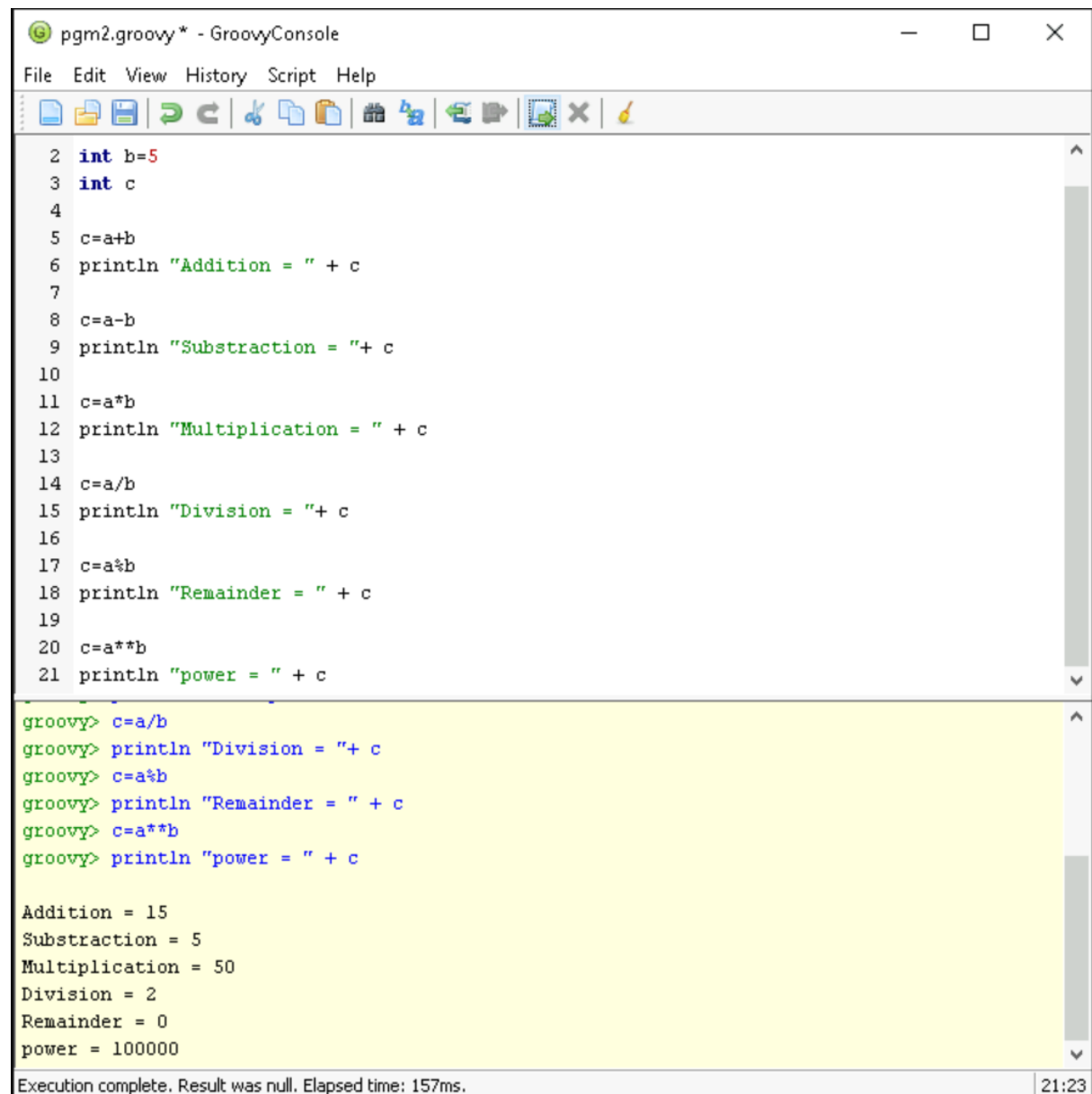
```
1 print "Welcome to javatpoint tutorial on groovy...."
```

```
groovy> print "Welcome to javatpoint tutorial on groovy...."
Welcome to javatpoint tutorial on groovy....
```

Execution complete. Result was null. Elapsed time: 390ms. 1:53

## Operators in groovy

### Q. Arithmetic Operator



The screenshot shows a GroovyConsole window titled "pgm2.groovy\* - GroovyConsole". The window has a menu bar (File, Edit, View, History, Script, Help) and a toolbar with icons for file operations and execution. The main text area contains Groovy code for arithmetic operations. Below the code, the console output shows the results of these operations. At the bottom, a status bar indicates "Execution complete. Result was null. Elapsed time: 157ms." and the time "21:23".

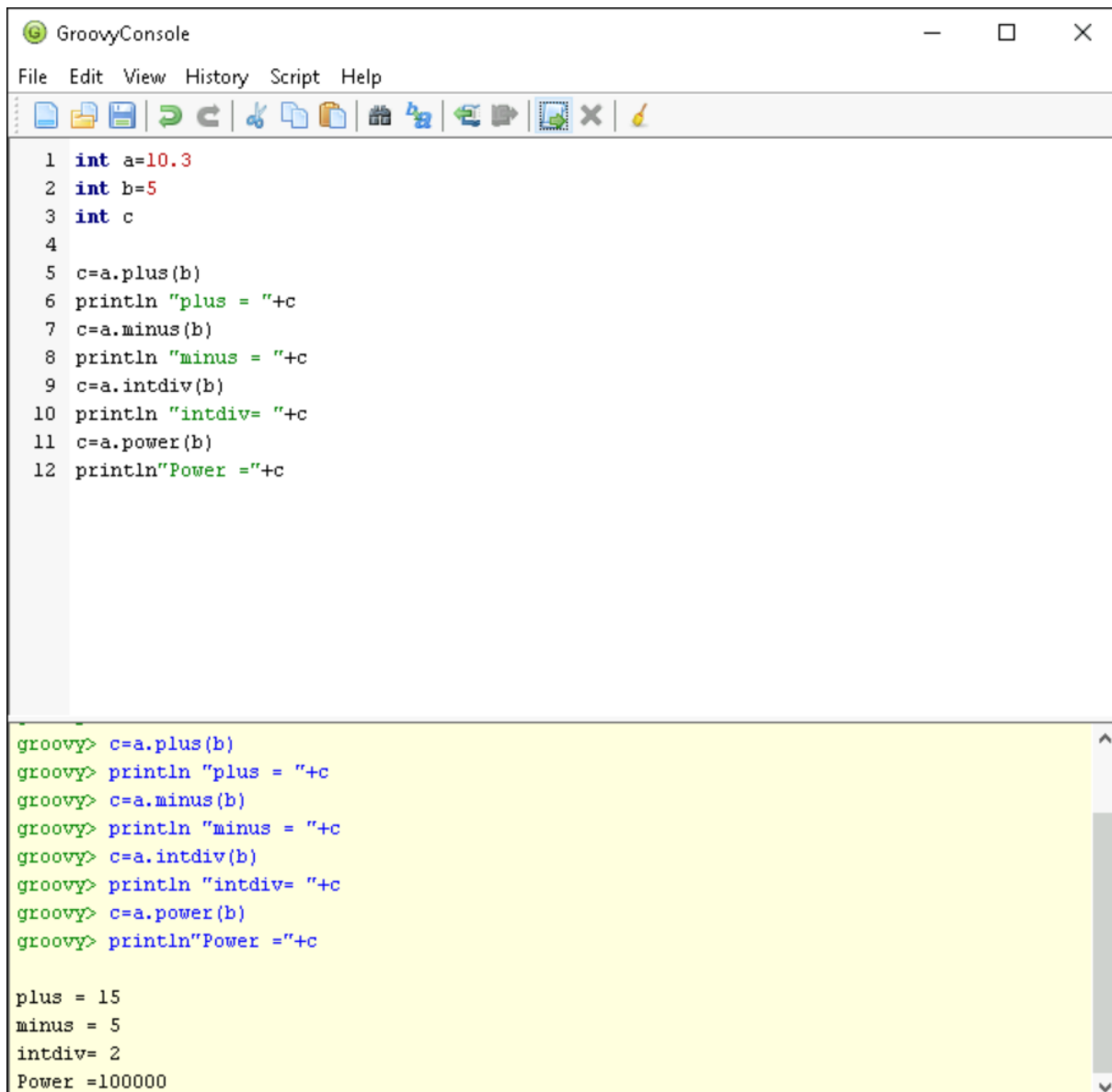
```
2 int b=5
3 int c
4
5 c=a+b
6 println "Addition = " + c
7
8 c=a-b
9 println "Substraction = "+ c
10
11 c=a*b
12 println "Multiplication = " + c
13
14 c=a/b
15 println "Division = "+ c
16
17 c=a%b
18 println "Remainder = " + c
19
20 c=a**b
21 println "power = " + c

groovy> c=a/b
groovy> println "Division = " + c
groovy> c=a%b
groovy> println "Remainder = " + c
groovy> c=a**b
groovy> println "power = " + c

Addition = 15
Substraction = 5
Multiplication = 50
Division = 2
Remainder = 0
power = 100000

Execution complete. Result was null. Elapsed time: 157ms. 21:23
```

Q.



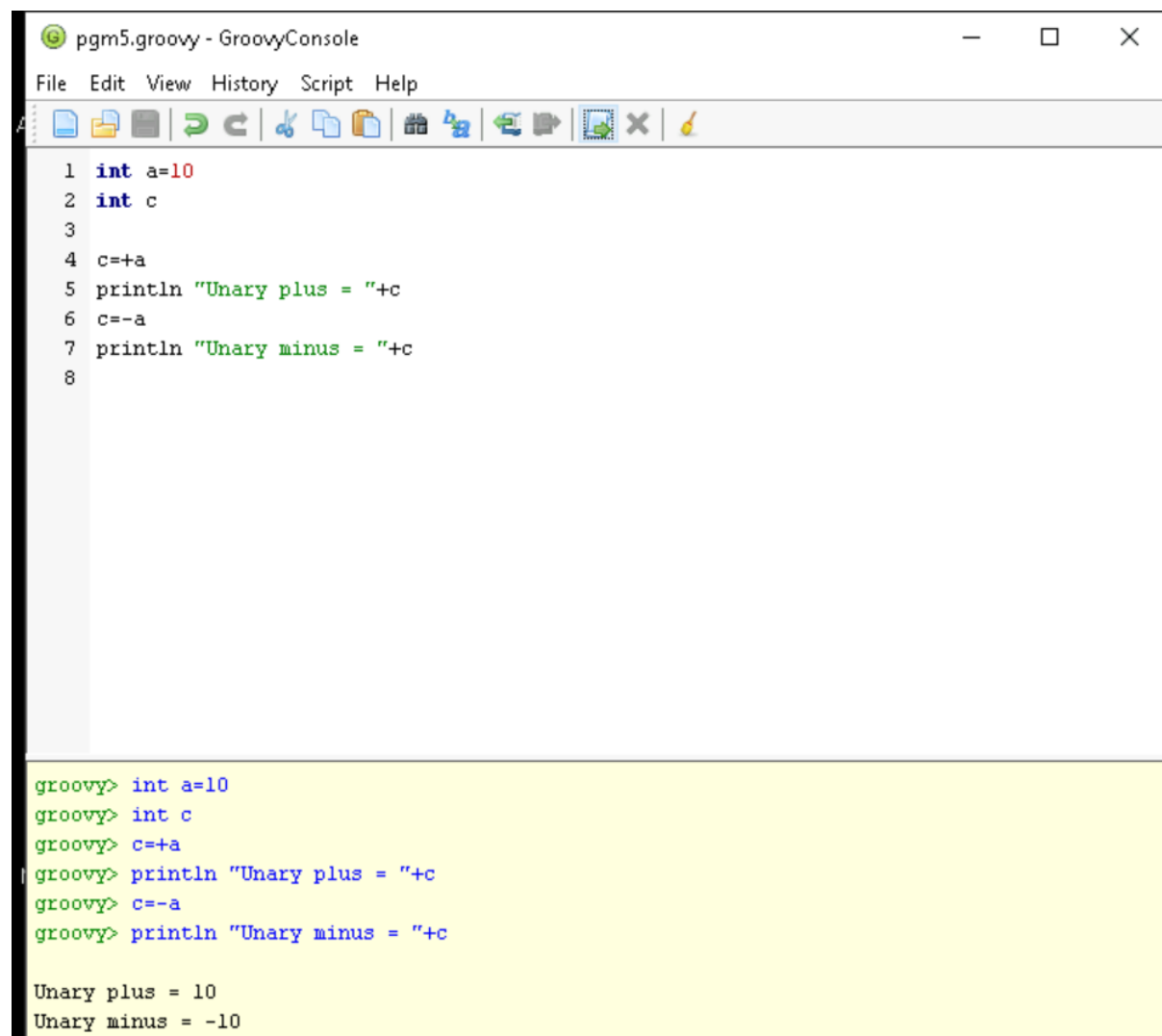
The screenshot shows a window titled "GroovyConsole" with a standard menu bar (File, Edit, View, History, Script, Help) and a toolbar. The main text area contains 12 lines of Groovy code. Below the code is a yellow-shaded output area showing the execution results of the code. The code defines variables a, b, and c, and then performs arithmetic operations (plus, minus, intdiv, power) on them, printing the results.

```
1 int a=10.3
2 int b=5
3 int c
4
5 c=a.plus(b)
6 println "plus = "+c
7 c=a.minus(b)
8 println "minus = "+c
9 c=a.intdiv(b)
10 println "intdiv= "+c
11 c=a.power(b)
12 println "Power = "+c
```

groovy> c=a.plus(b)  
groovy> println "plus = "+c  
groovy> c=a.minus(b)  
groovy> println "minus = "+c  
groovy> c=a.intdiv(b)  
groovy> println "intdiv= "+c  
groovy> c=a.power(b)  
groovy> println "Power = "+c

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

## Q.Unary Operator



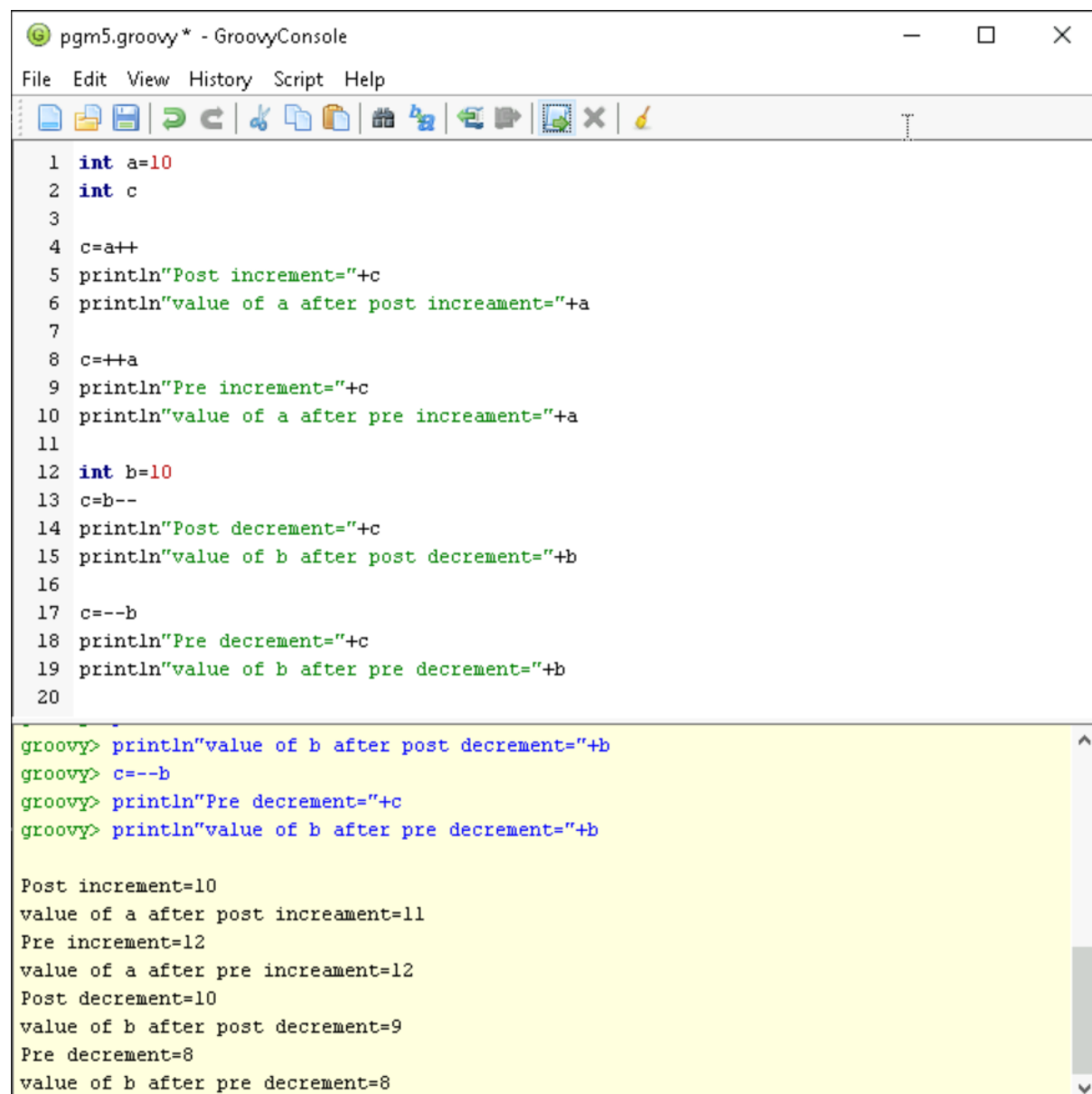
The screenshot shows a window titled "pgm5.groovy - GroovyConsole". The window has a menu bar with "File", "Edit", "View", "History", "Script", and "Help". Below the menu bar is a toolbar with various icons for file operations and execution. The main area of the window contains a script with the following code:

```
1 int a=10
2 int c
3
4 c+=a
5 println "Unary plus = "+c
6 c=-a
7 println "Unary minus = "+c
8
```

Below the script, the output of the execution is displayed on a yellow background:

```
groovy> int a=10
groovy> int c
groovy> c+=a
groovy> println "Unary plus = "+c
Unary plus = 10
groovy> c=-a
groovy> println "Unary minus = "+c
Unary minus = -10
```

Q.



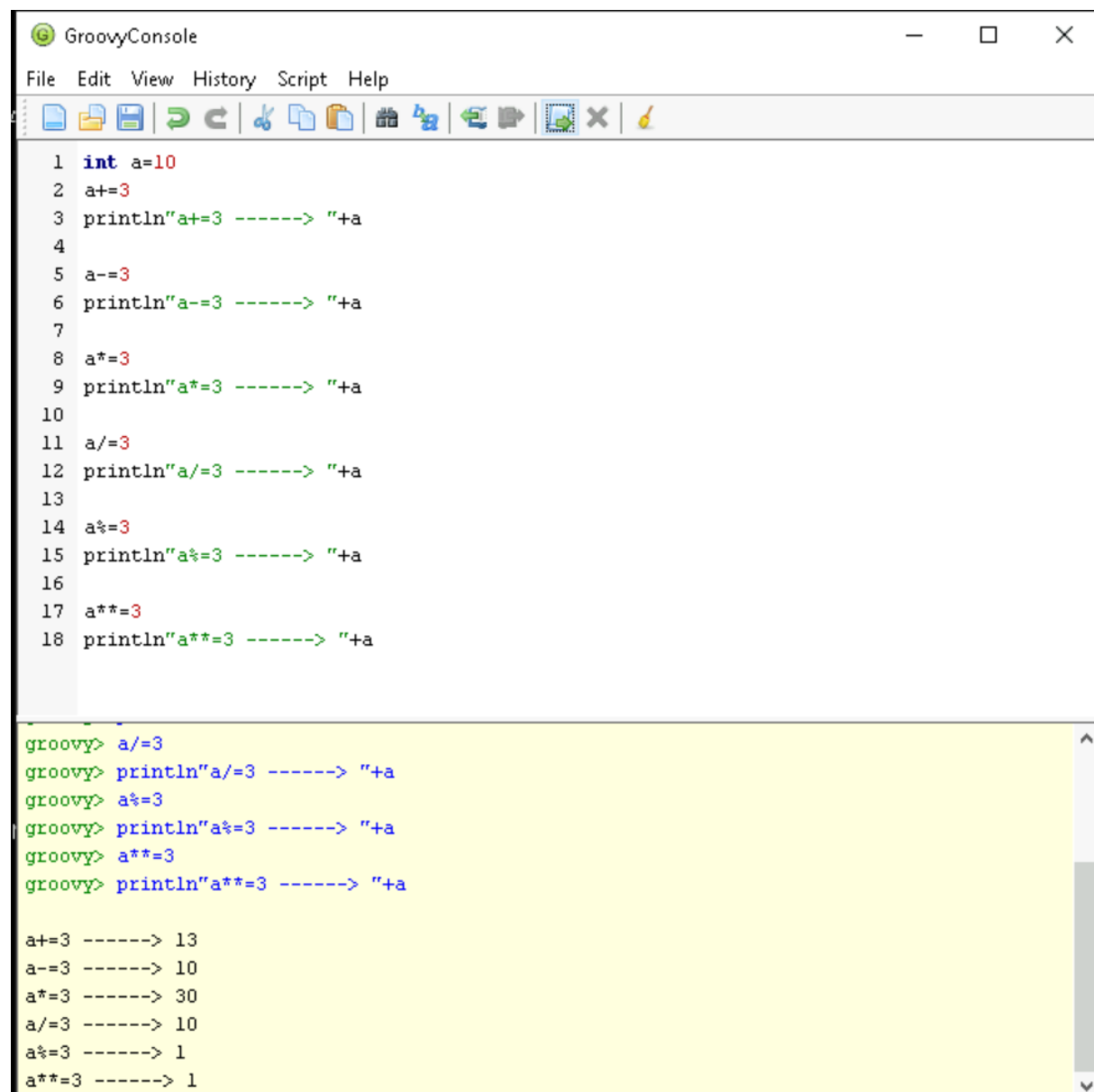
The screenshot shows a window titled "pgm5.groovy\* - GroovyConsole". The window has a menu bar with "File", "Edit", "View", "History", "Script", and "Help". Below the menu bar is a toolbar with various icons for file operations and execution. The main area contains Groovy code with line numbers 1 through 20. The code defines two integer variables, `a` and `b`, and performs post-increment, post-decrement, pre-increment, and pre-decrement operations on them, printing the results at each step. The output of the code is displayed in a separate pane at the bottom, showing the sequence of printed values.

```
1 int a=10
2 int c
3
4 c=a++
5 println"Post increment="+c
6 println"value of a after post increament="+a
7
8 c=++a
9 println"Pre increment="+c
10 println"value of a after pre increament="+a
11
12 int b=10
13 c=b--
14 println"Post decrement="+c
15 println"value of b after post decrement="+b
16
17 c=--b
18 println"Pre decrement="+c
19 println"value of b after pre decrement="+b
20
```

groovy> println"value of b after post decrement="+b  
groovy> c=--b  
groovy> println"Pre decrement="+c  
groovy> println"value of b after pre decrement="+b

Post increment=10  
value of a after post increament=11  
Pre increment=12  
value of a after pre increament=12  
Post decrement=10  
value of b after post decrement=9  
Pre decrement=8  
value of b after pre decrement=8

## Q. Assignment arithmetic operations



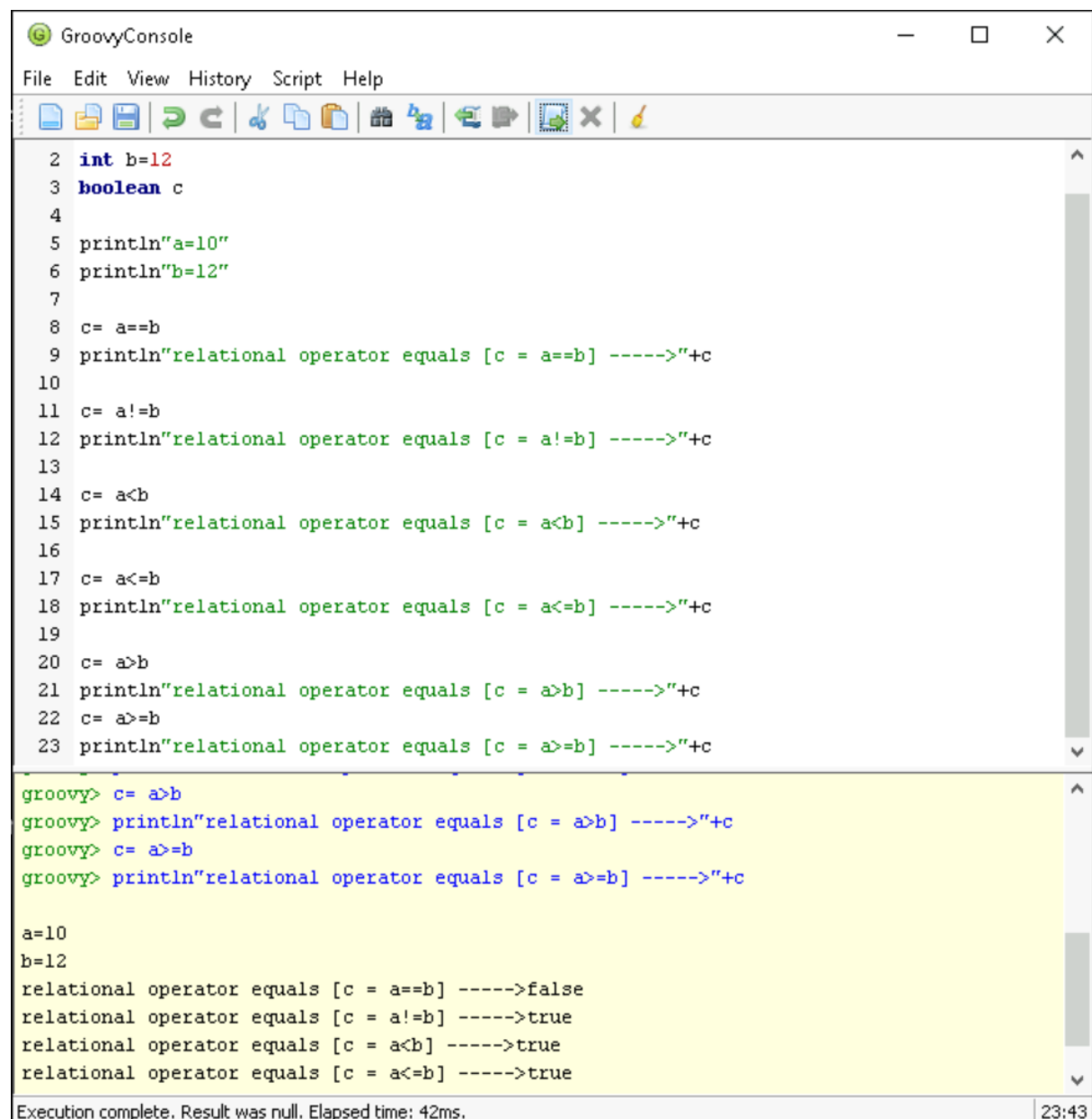
The screenshot shows a GroovyConsole window with a menu bar (File, Edit, View, History, Script, Help) and a toolbar. The script area contains 18 lines of code performing arithmetic operations on a variable 'a'. The output area shows the results of these operations, with the first five lines of output highlighted in yellow.

```
1  int a=10
2  a+=3
3  println"a+=3 -----> "+a
4
5  a-=3
6  println"a-=3 -----> "+a
7
8  a*=3
9  println"a*=3 -----> "+a
10
11 a/=3
12 println"a/=3 -----> "+a
13
14 a%=3
15 println"a%=3 -----> "+a
16
17 a**=3
18 println"a**=3 -----> "+a
```

```
groovy> a/=3
groovy> println"a/=3 -----> "+a
groovy> a%=3
groovy> println"a%=3 -----> "+a
groovy> a**=3
groovy> println"a**=3 -----> "+a

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

## Q. Relational operator



The screenshot shows a GroovyConsole window with a menu bar (File, Edit, View, History, Script, Help) and a toolbar. The main area contains Groovy code for testing relational operators. The code defines variables `a=10` and `b=12`, and then tests various relational operators (`==`, `!=`, `<`, `<=`, `>`, `>=`) by assigning the result to a variable `c` and printing it. The output shows the results of these operations: `a==b` is false, while the others are true. The console also shows interactive commands entered at the prompt, such as `c = a > b` and `println` statements, and their corresponding outputs.

```
2 int b=12
3 boolean c
4
5 println"a=10"
6 println"b=12"
7
8 c= a==b
9 println"relational operator equals [c = a==b] ----->" +c
10
11 c= a!=b
12 println"relational operator equals [c = a!=b] ----->" +c
13
14 c= a<b
15 println"relational operator equals [c = a<b] ----->" +c
16
17 c= a<=b
18 println"relational operator equals [c = a<=b] ----->" +c
19
20 c= a>b
21 println"relational operator equals [c = a>b] ----->" +c
22 c= a>=b
23 println"relational operator equals [c = a>=b] ----->" +c

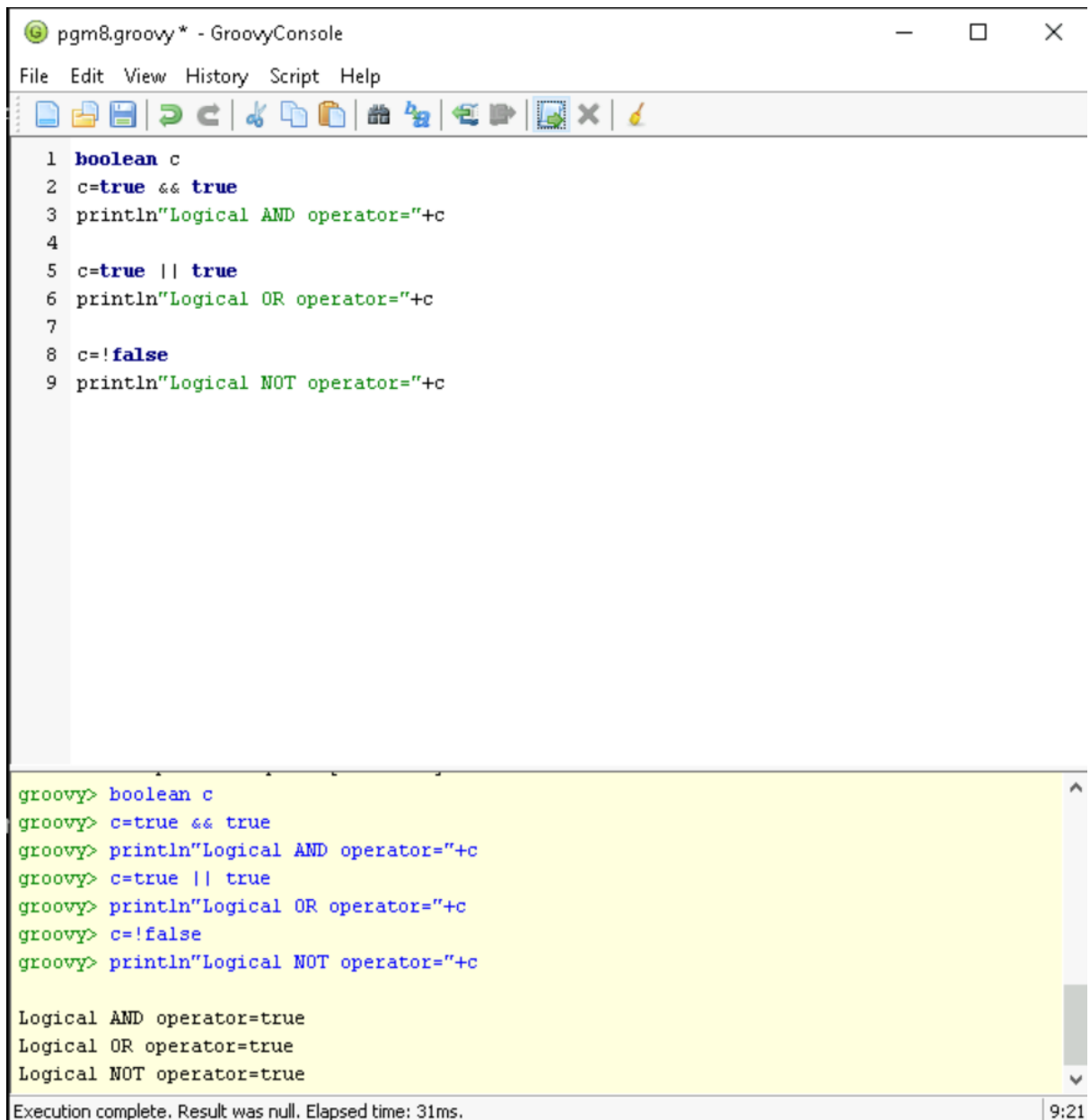
groovy> c= a>b
groovy> println"relational operator equals [c = a>b] ----->" +c
groovy> c= a>=b
groovy> println"relational operator equals [c = a>=b] ----->" +c

a=10
b=12
relational operator equals [c = a==b] ----->>false
relational operator equals [c = a!=b] ----->>true
relational operator equals [c = a<b] ----->>true
relational operator equals [c = a<=b] ----->>true

Execution complete. Result was null. Elapsed time: 42ms.
```



## Q.Logical operator



The screenshot shows a GroovyConsole window titled "pgm8.groovy\* - GroovyConsole". The window has a menu bar (File, Edit, View, History, Script, Help) and a toolbar with various icons. The main text area contains the following Groovy code:

```
1 boolean c
2 c=true && true
3 println"Logical AND operator="+c
4
5 c=true || true
6 println"Logical OR operator="+c
7
8 c=!false
9 println"Logical NOT operator="+c
```

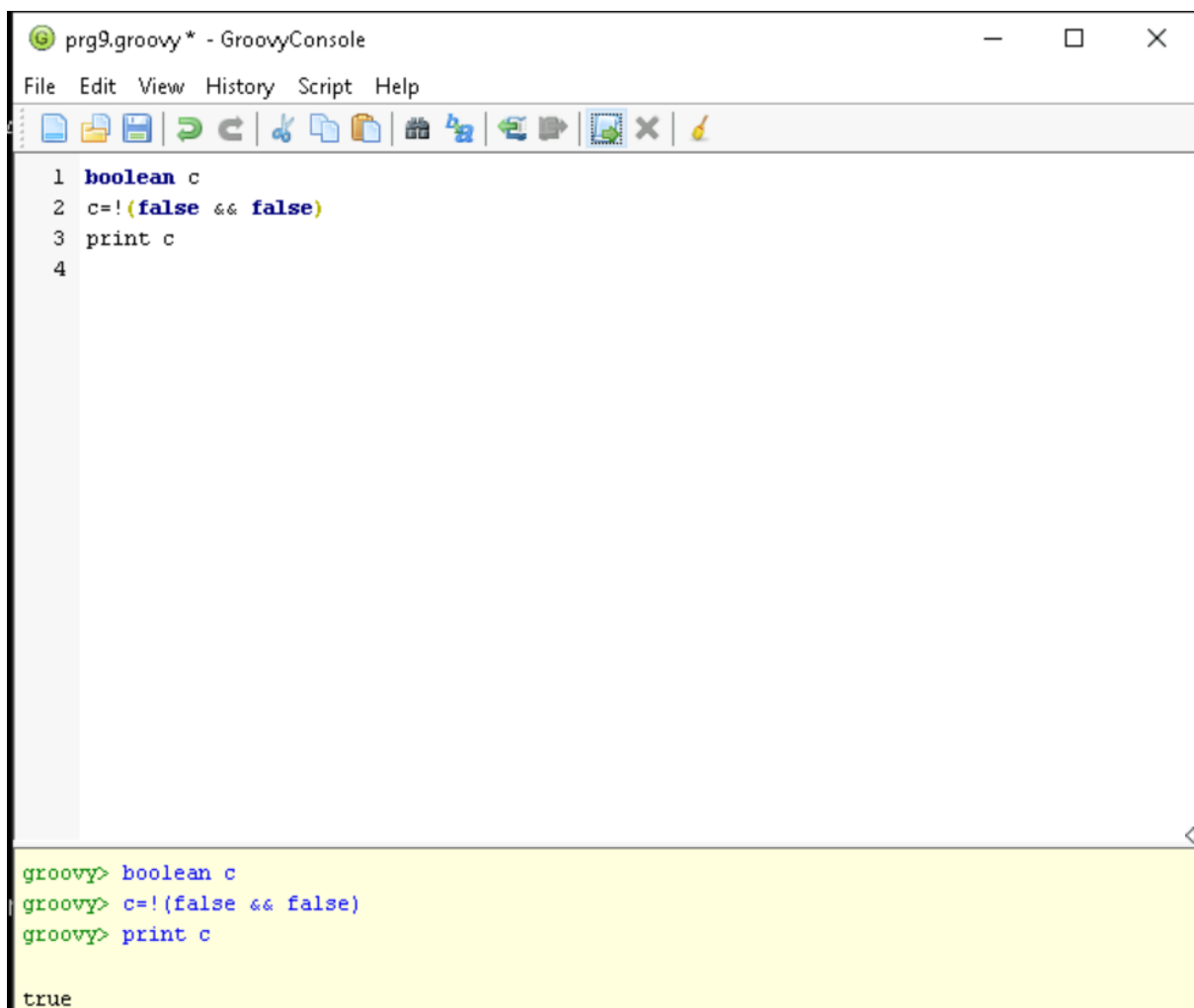
Below the code, the execution output is displayed on a yellow background. It shows the same code being executed line by line, followed by the results of the println statements:

```
groovy> boolean c
groovy> c=true && true
groovy> println"Logical AND operator="+c
groovy> c=true || true
groovy> println"Logical OR operator="+c
groovy> c=!false
groovy> println"Logical NOT operator="+c

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

At the bottom of the window, a status bar indicates "Execution complete. Result was null. Elapsed time: 31ms." and the time "9:21" is shown on the right.

Q.



The image shows a screenshot of a GroovyConsole window titled "prg9.groovy\* - GroovyConsole". The window has a menu bar with "File", "Edit", "View", "History", "Script", and "Help". Below the menu bar is a toolbar with various icons for file operations, editing, and execution. The main area of the window contains a Groovy script with four lines:

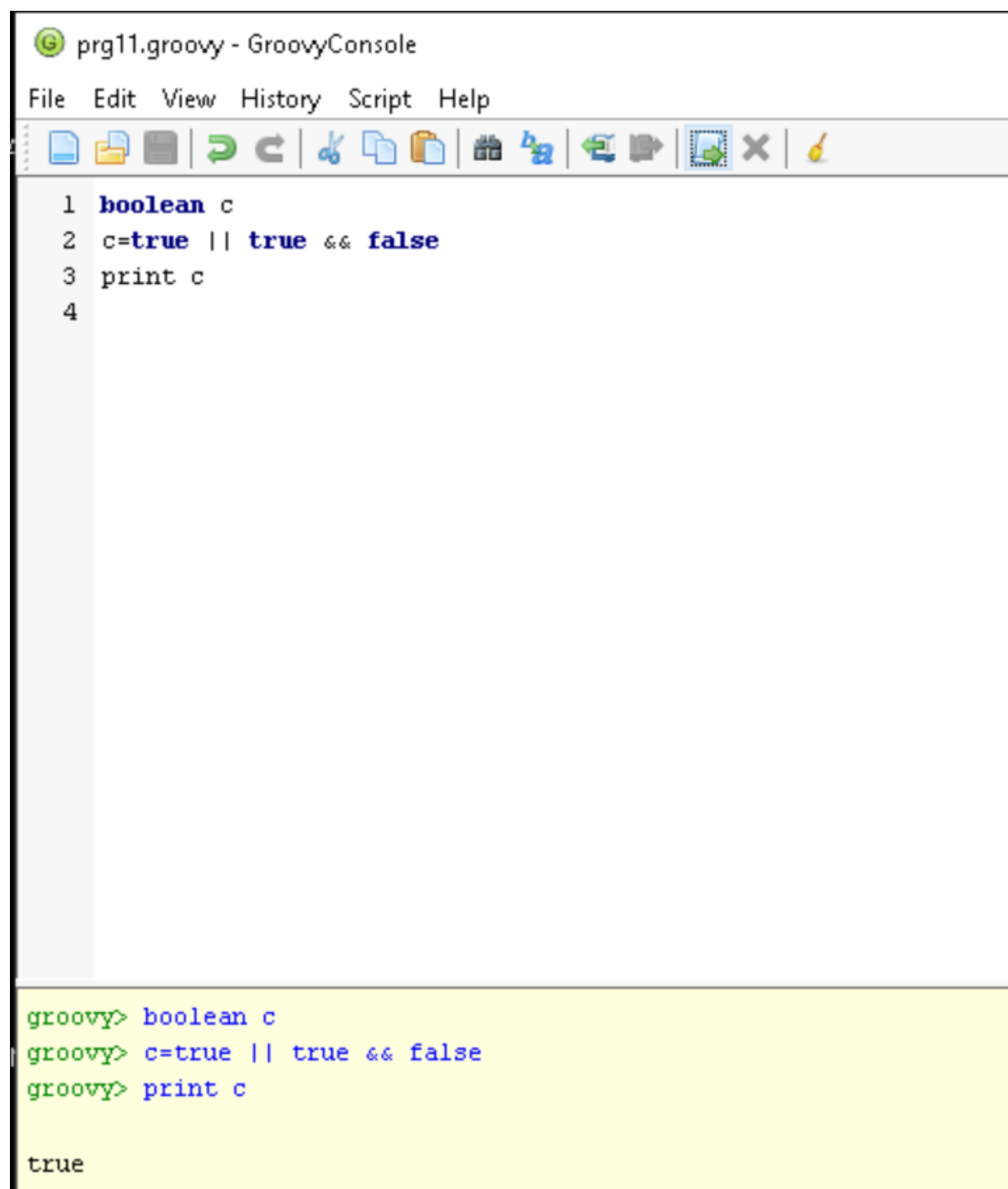
```
1 boolean c
2 c=! (false && false)
3 print c
4
```

Below the script, the execution output is displayed on a yellow background. It shows the commands entered in the console and the resulting output:

```
groovy> boolean c
groovy> c=! (false && false)
groovy> print c

true
```

Q.



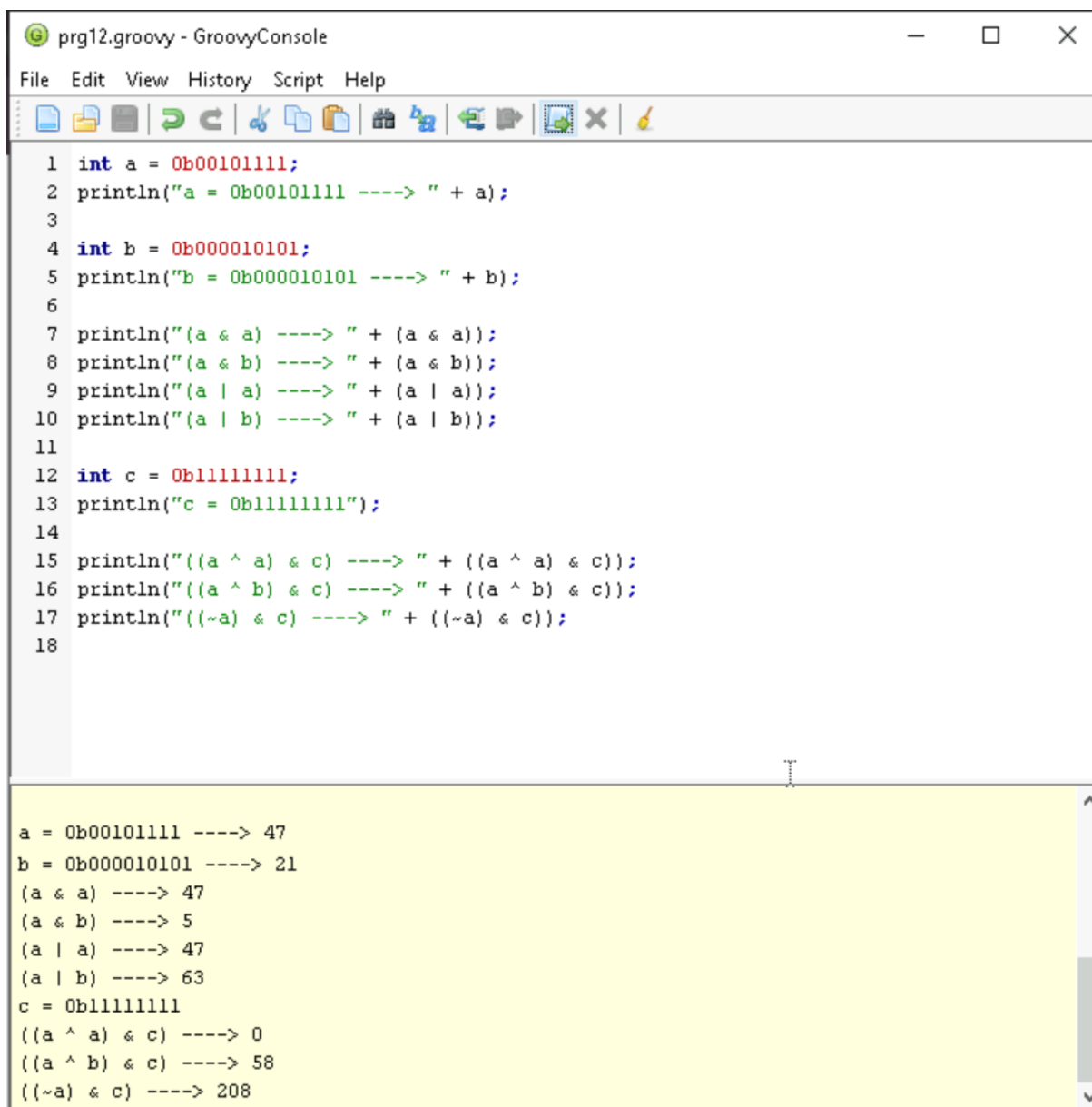
```
prg11.groovy - GroovyConsole
File Edit View History Script Help

1 boolean c
2 c=true || true && false
3 print c
4

groovy> boolean c
groovy> c=true || true && false
groovy> print c

true
```

## Q. Bitwise operators

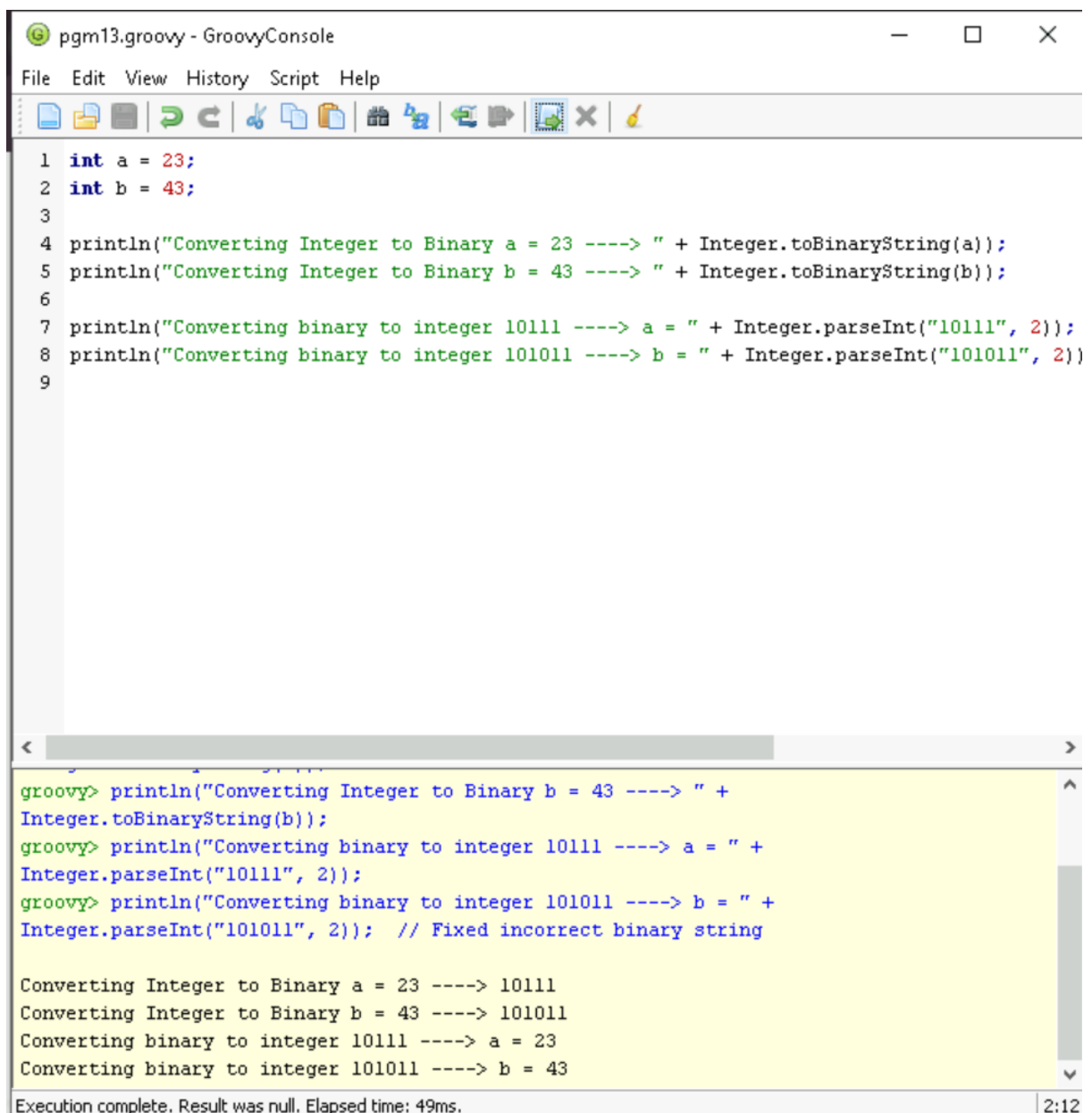


The screenshot shows a GroovyConsole window titled "prg12.groovy - GroovyConsole". The window has a menu bar with "File", "Edit", "View", "History", "Script", and "Help". Below the menu bar is a toolbar with various icons for file operations and execution. The main area contains Groovy code that defines three integers: `a = 0b00101111`, `b = 0b000010101`, and `c = 0b11111111`. It then performs several bitwise operations and prints the results. The output at the bottom shows the decimal values of these operations.

```
1 int a = 0b00101111;
2 println("a = 0b00101111 ----> " + a);
3
4 int b = 0b000010101;
5 println("b = 0b000010101 ----> " + b);
6
7 println("(a & a) ----> " + (a & a));
8 println("(a & b) ----> " + (a & b));
9 println("(a | a) ----> " + (a | a));
10 println("(a | b) ----> " + (a | b));
11
12 int c = 0b11111111;
13 println("c = 0b11111111");
14
15 println("((a ^ a) & c) ----> " + ((a ^ a) & c));
16 println("((a ^ b) & c) ----> " + ((a ^ b) & c));
17 println("((~a) & c) ----> " + ((~a) & c));
18
```

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

Q.



The screenshot shows a GroovyConsole window titled "pgm13.groovy - GroovyConsole". The window has a menu bar (File, Edit, View, History, Script, Help) and a toolbar with icons for file operations and execution. The main text area contains the following Groovy code:

```
1 int a = 23;
2 int b = 43;
3
4 println("Converting Integer to Binary a = 23 ----> " + Integer.toBinaryString(a));
5 println("Converting Integer to Binary b = 43 ----> " + Integer.toBinaryString(b));
6
7 println("Converting binary to integer 10111 ----> a = " + Integer.parseInt("10111", 2));
8 println("Converting binary to integer 101011 ----> b = " + Integer.parseInt("101011", 2));
9
```

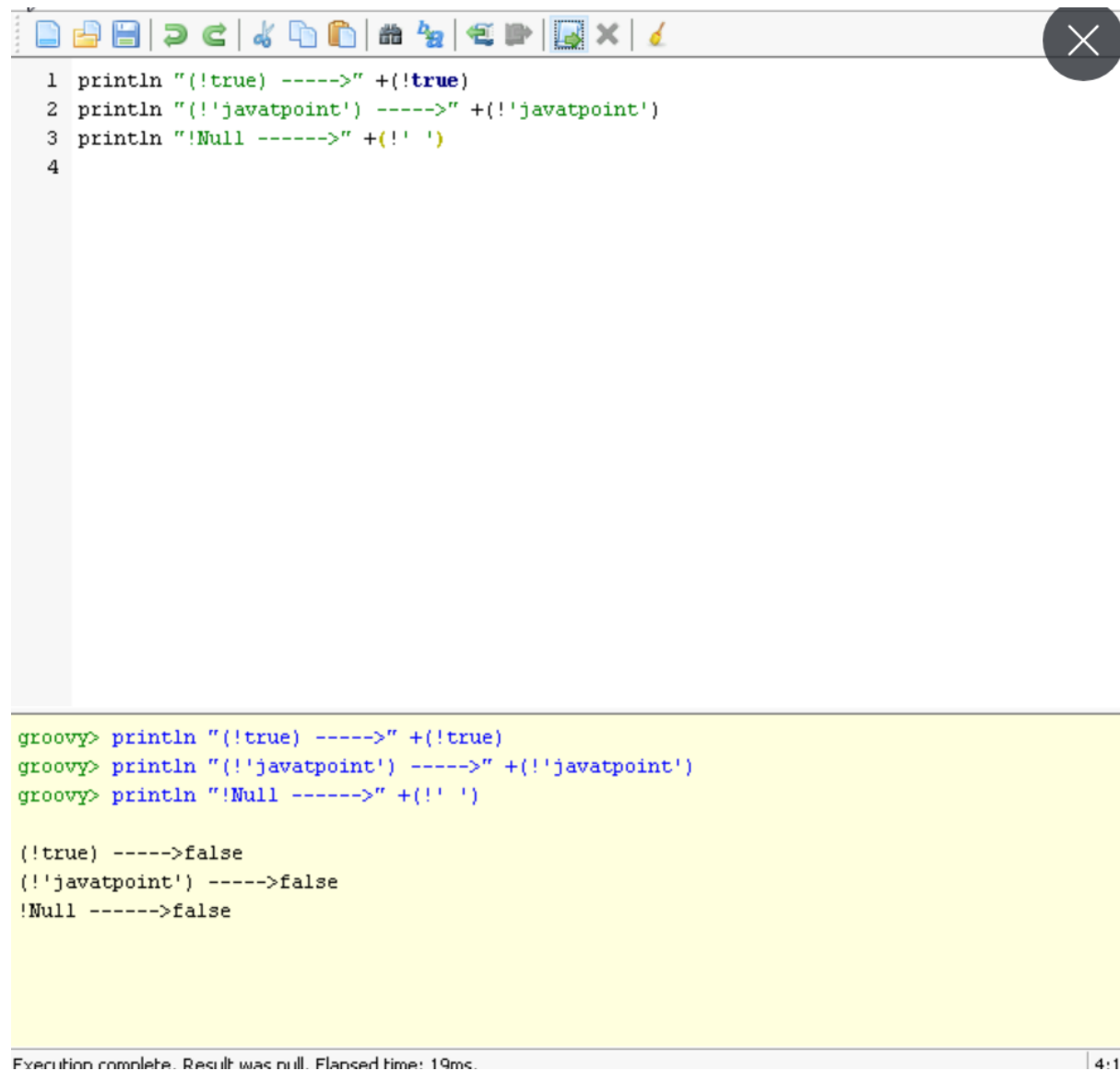
Below the code, the execution output is displayed on a yellow background. It shows the same code being executed line by line, with the output of each println statement. The output is:

```
groovy> println("Converting Integer to Binary b = 43 ----> " +
Integer.toBinaryString(b));
groovy> println("Converting binary to integer 10111 ----> a = " +
Integer.parseInt("10111", 2));
groovy> println("Converting binary to integer 101011 ----> b = " +
Integer.parseInt("101011", 2)); // Fixed incorrect binary string

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 = 43
```

At the bottom of the window, a status bar indicates "Execution complete. Result was null. Elapsed time: 49ms." and the time "2:12".

## Q. Conditional operators



The screenshot shows a code editor window with a toolbar at the top. The code is written in Groovy and uses the `println` method to output the results of logical negations. The code is as follows:

```
1 println "(!true) ---->" +(!true)
2 println "(!'javatpoint') ---->" +(!'javatpoint')
3 println "(!Null ---->" +(!' ')
4
```

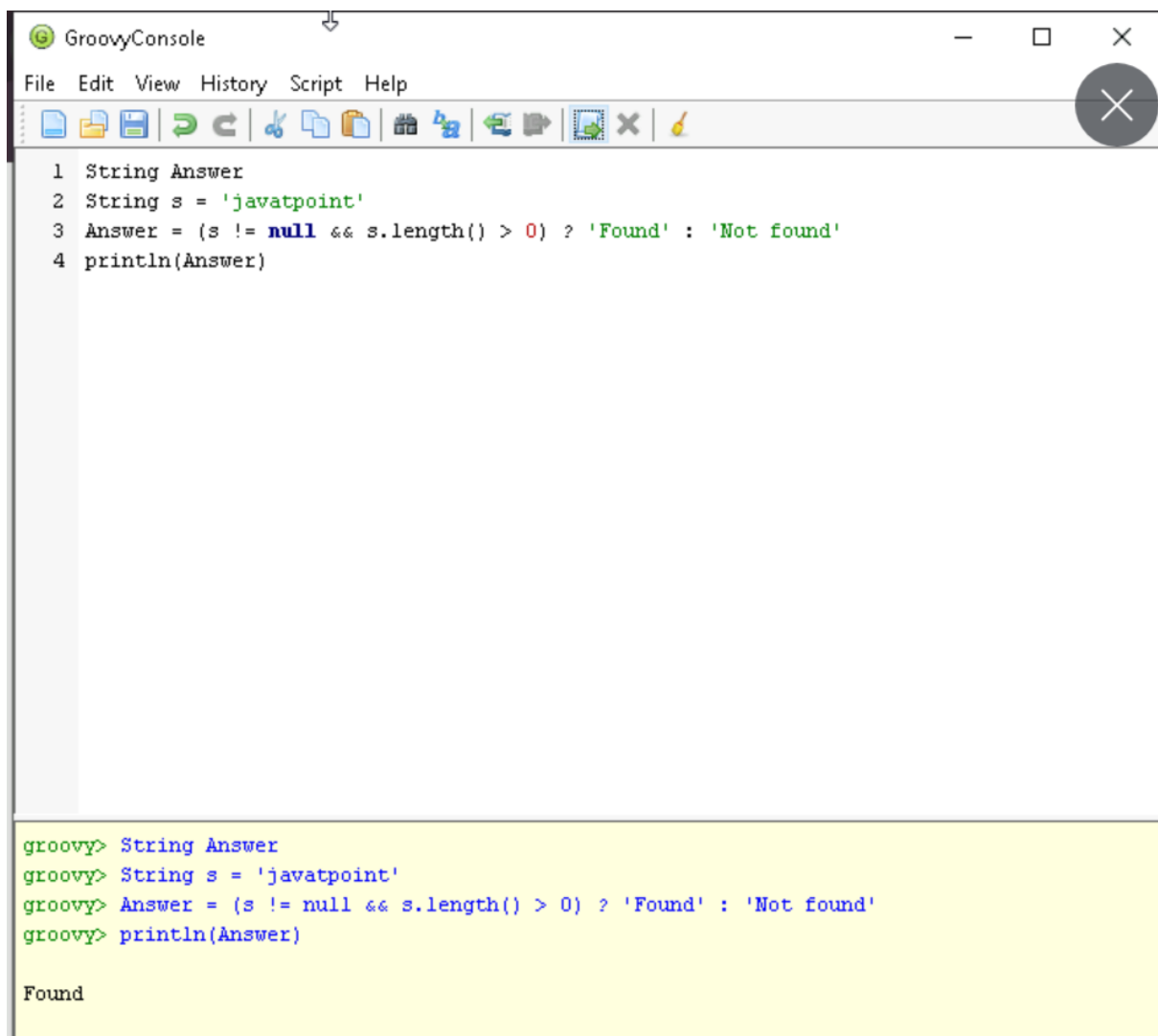
Below the code editor, the execution output is displayed on a yellow background. It shows the same three lines of code being executed, followed by the results of the logical negations:

```
groovy> println "(!true) ---->" +(!true)
groovy> println "(!'javatpoint') ---->" +(!'javatpoint')
groovy> println "(!Null ---->" +(!' ')

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

At the bottom of the window, a status bar indicates: "Execution complete. Result was null. Flashed time: 19ms." and a page number "4/1".

## Q. Ternary operator



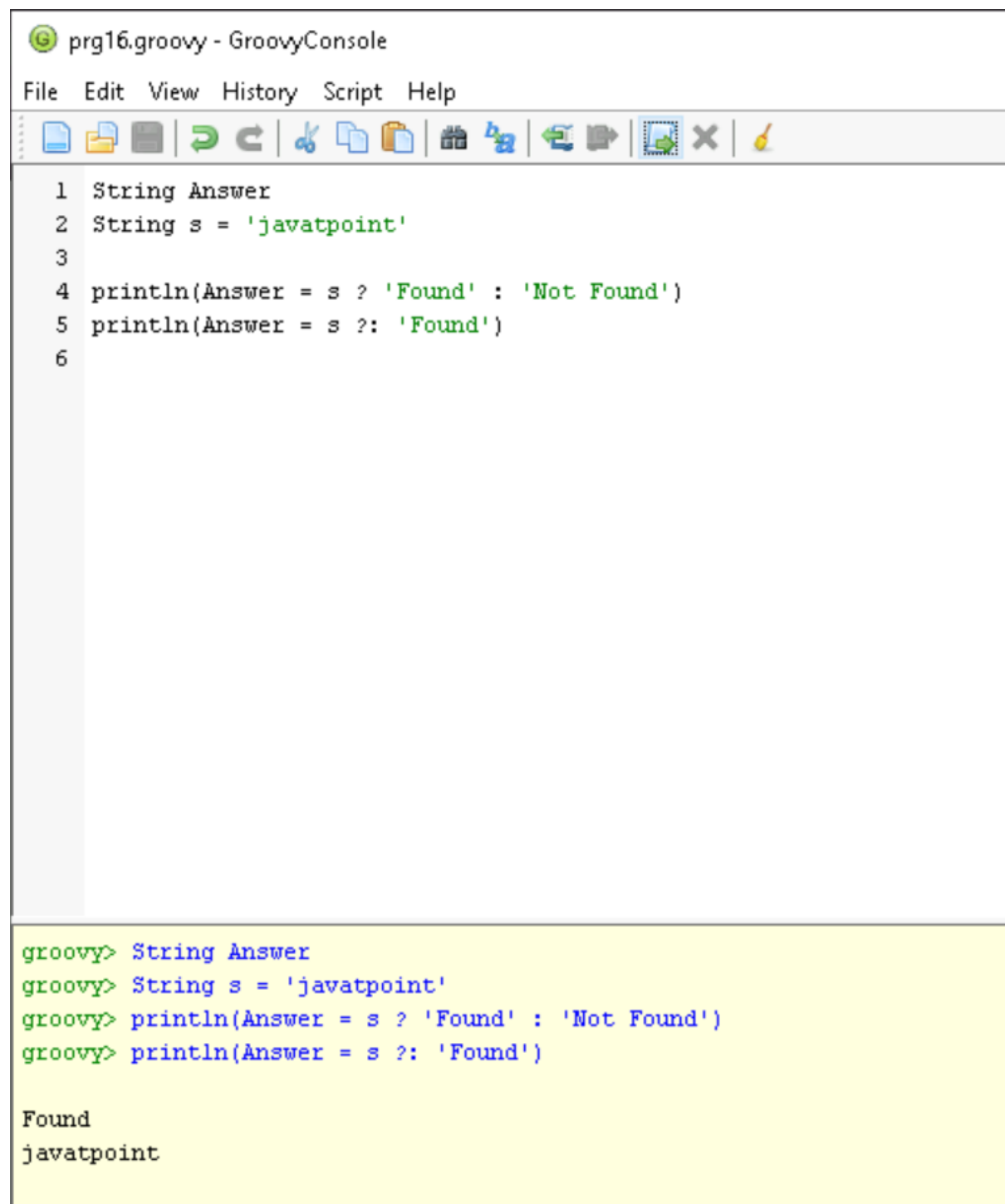
The screenshot shows a window titled "GroovyConsole" with a menu bar (File, Edit, View, History, Script, Help) and a toolbar. The main text area contains a Groovy script using the ternary operator. Below the script, the execution output is displayed on a yellow background.

```
1 String Answer
2 String s = 'javatpoint'
3 Answer = (s != null && s.length() > 0) ? 'Found' : 'Not found'
4 println(Answer)
```

```
groovy> String Answer
groovy> String s = 'javatpoint'
groovy> Answer = (s != null && s.length() > 0) ? 'Found' : 'Not found'
groovy> println(Answer)

Found
```

## Q.Elvis Operator



The screenshot shows a GroovyConsole window titled "prg16.groovy - GroovyConsole". The window has a menu bar with "File", "Edit", "View", "History", "Script", and "Help". Below the menu bar is a toolbar with various icons for file operations and execution. The main area of the window contains a script with six lines of code. The first line declares a variable "Answer" of type "String". The second line declares a variable "s" of type "String" and assigns it the value "javatpoint". The third line is empty. The fourth line uses the Elvis operator to assign the value of "s" to "Answer" if "s" is not null, otherwise it assigns the value "Found". The fifth line uses the Elvis operator to assign the value of "s" to "Answer" if "s" is not null, otherwise it assigns the value "Found". The sixth line is empty. The output of the script is displayed in a yellow background area at the bottom of the window. The output shows the value of "Answer" after each assignment, which is "Found" for the first assignment and "javatpoint" for the second assignment.

```
1 String Answer
2 String s = 'javatpoint'
3
4 println(Answer = s ? 'Found' : 'Not Found')
5 println(Answer = s ?: 'Found')
6
```

```
groovy> String Answer
groovy> String s = 'javatpoint'
groovy> println(Answer = s ? 'Found' : 'Not Found')
groovy> println(Answer = s ?: 'Found')

Found
javatpoint
```